Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

ŠTÁTNY PEDAGOGICKÝ ÚSTAV
NATIONAL INSTITUTE FOR EDUCATION

Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

Robotické stavebnice vo vzdelávaní

Predmet: Robotické stavebnice vo vzdelávaní

Línia: Digitálna gramotnosť učiteľa

Operačný program VZDELÁVANIE

EURÓPSKA ÚNIA
Európsky sociálny fond

Európska únia
Európsky sociálny fond

Moderné vzdelávanie pre vedomostnú spoločnosť/Projekt je spolufinancovaný zo zdrojov ES

Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

Robotické stavebnice vo vzdelávaní

Garant predmetu:

RNDr. Peter Tomcsányi
KZVI FMFI UK, Bratislava
tomcsanyi@fmph.uniba.sk

Autori modulu:

RNDr. Peter Tomcsányi
KZVI FMFI UK, Bratislava
PaedDr. Martina Kabátová
KZVI FMFI UK, Bratislava
PaedDr. Janka Pekárová
KZVI FMFI UK, Bratislava
Mgr. Ľubomír Janovický
KZVI FMFI UK, Bratislava

Identifikácia modulu

Aktivita projektu:	1.2 Vzdelávanie nekvalifikovaných učiteľov informatiky na 2. stupni ZŠ a na SŠ
Línia aktivity:	Digitálna gramotnosť učiteľa
Predmet:	Robotické stavebnice vo vzdelávaní

Zaradenie modulu

Predmet je zaradený do línie digitálnej gramotnosti učiteľa ako praktický úvod, v ktorom sa účastníci zoznámia s konkrétnou robotickou stavebnicou a jej programovaním. Na tento predmet ďalej nadviaže predmet Didaktika robotických stavebníc, ktorý prehĺbi schopnosti účastníkov používať nadobudnuté vedomosti so svojimi žiakmi.

Abstrakt modulu

V module sa po krátkom úvode a prehľade problematiky robotických stavebníc dostanú účastníci k praktickému programovaniu modernej a vo svete rozšírenej stavebnice LEGO MINDSTORMS NXT. Základným kameňom stavebnice je programovateľná kocka NXT, ktorú sa najprv naučia zapojiť, potom nastavovať a programovať pomocou jej vstavaného programu.

Väčšinu času modulu je venovaná programovaniu v ikonickom programovacom prostredí LEGO MINDSTORMS Education NXT, ktoré beží na počítačoch s operačným systémom Microsoft Windows. Naučia sa používať prostredie programu, programovať základné pohyby robotov, využívať senzory, programové konštrukcie cyklus a vetvenie. Na záver sa budú venovať pokročilejším témam ako je používanie údajov a premenných a posielanie správ medzi niekoľkým NXT kockami.

Po absolvovaní by mali účastníci ovládať programovanie konkrétnej stavebnice na dostatočnej úrovni na to, aby mohli jednak takúto stavebnicu použiť vo výučbe jednak aby pochopili na tomto konkrétnom príklade osobitosti programovania robotických stavebníc tak, aby zvládli ľahšie aj používanie iných stavebníc založených na podobných princípoch.

Modul má e-learningovú podporu

Obsah

Cieľ modulu

Účastníci vzdelávania

- sa podrobnejšie zoznámia s programovateľnými stavebnicami LEGO NXT,
- vytvoria sadu jednoduchých programov pre riadenie robotického modelu,
- získajú prehľad o jestvujúcich robotických platformách pre vzdelávanie.

Zabezpečenie modulu

Softvérové a hardvérové požiadavky a odporúčania

- PC so zvukovou kartou a reproduktormi alebo slúchadlami.
- Windows XP alebo Windows Vista.
- Nainštalovaný Browser a prezerač PDF súborov.
- Nainštalovaný Program LEGO MINDSTORMS Education NXT.
- Základné stavebnice Lego NXT v počte jedna na nie viac než 5 účastníkov.
- Postavený základný robot podľa návodu zo stavebnice **bez zapojenia káblikov pripájajúcich jednotlivé senzory.**
- Čierna lepiaca páska, prípadne veľké hárky papiera a fixky na vytváranie značiek pre svetelný senzor.

V tejto časti budeme používať základný model robota používaný v časti Robot Educator. Lektor by mal zaistiť poskladanie týchto robotov podľa návodu v stavebnici. Senzory a motory pripoja účastníci sami, je však potrebné mať časti na ich pripevnenie postavené vopred.

E-learning

V E-learningovej podpore bude účastníkom kurzu k dispozícii text materiálu, prípadne niektoré doplňujúce materiály.

Vstupné vedomosti

Požadované prerekvizity

Účastník vzdelávania absolvoval tieto moduly:

- Základná digitálna gramotnosť
- Digitálna gramotnosť učiteľa
- Programovanie 0
- Programovanie 1

Predpokladané vstupné vedomosti

Účastník vzdelávania

- ovláda základy práce so súbormi (otvorenie, ukladanie, kopírovanie, odstraňovanie súborov)
- efektívne spravuje súbory,
- pracuje so základnými aplikáciami operačného systému (spustenie, ukončenie konkrétnej aplikácie)
- vie obsluhovať USB zariadenie pripojené k počítaču,
- vytvára jednoduché programy využívajúce základné koncepty programovania (premenná, cyklus, podmienka),
- dokáže postaviť robotický model podľa obrázkového návodu a zodpovedajúceho popisu.

Edukačná robotika

Miesto robotiky vo vzdelávaní

Keďže robotika si našla svoje miesto v osnovách informatiky, na mnohých školách sa stavebnice nachádzajú a aj robotické súťaže sa stali populárnou mimoškolskou aktivitou, považujeme za prínosné oboznámiť účastníkov vzdelávania s robotickými stavebnicami a princípmi učenia sa, ktoré sa v robotike uplatňujú.

Prieskum

Používate roboty vo svojej výučbe? Ak áno, aké a ako sa vám takéto vyučovanie osvedčuje?

Edukačná robotika môže podporovať **medzipredmetové vzťahy** a **myslenie v súvislostiach**, **tímovú prácu**, ale aj schopnosť **plánovať**, **testovať** svoje postupy a odstraňovať chyby vo svojom riešení.

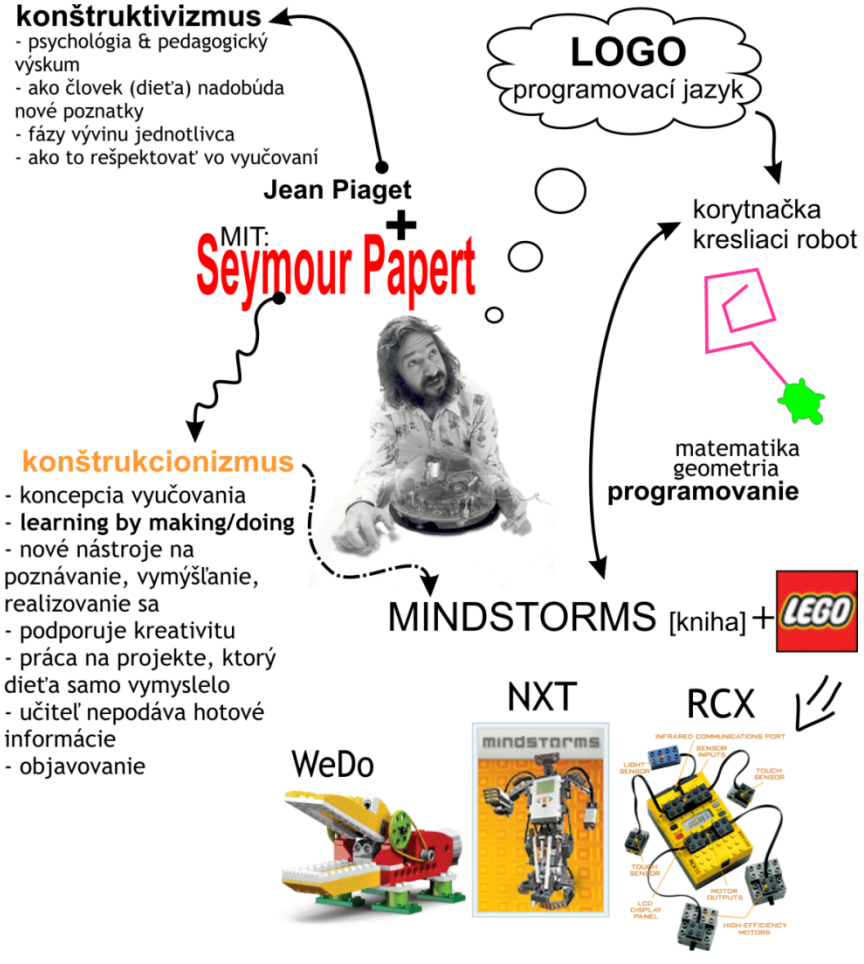
Programovanie skutočného alebo virtuálneho robota prináša viaceré pozitíva, ktoré ho robia jedinečným nástrojom pre posilnenie schopnosti plánovať (voľne podľa [2]):

- vedie k **presnosti v myslení** – nie sú dovolené nejednoznačné formulácie, robot vykonáva presne to, čo určuje jeho program;
- je činnosťou, v ktorej sa používajú všeobecné pojmy ako zobrazenie, funkcia a premenná a môžeme pritom pozorovať dôsledky ich použitia; prostredníctvom ovládania robota či cez grafickú plochu umožňuje **zobraziť myslenie**;
- podporuje riešenie problémov ich **rozdelením na menšie časti**, ale aj **hľadaním analógie**; pri robotike sa navyše mnohokrát využije **heuristický prístup k riešeniu problému**;
- učí nás **nebáť sa chýb** – sú štartovacím prvkom pre zlepšenie nášho riešenia;
- rozvíja naše **metakognitívne schopnosti** – ide o jeden z mála procesov, ktorý je založený na popise nášho spôsobu myslenia a riešenia problému. Výber najlepšieho riešenia závisí od analýzy alternatív a hodnotenia ich kvality, čo od nás vyžaduje hlboké pochopenie problému pri rôznych vstupoch.

Heuristika je hľadanie riešenia na základe skúseností, ktoré je dostatočne dobré, ale nie nutne najlepšie.

Metakognitívne schopnosti sú také, ktoré sa týkajú premýšľania o našom vlastnom procese učenia sa a poznávania.

Mindstorms je názov slávnej knižky Seymoura Paperta, viď modul *Vízie a inšpirácie (2MŠ2)*, v ktorej sa zdôrazňuje učenie sa robením a úloha programovania pre rozvoj detského myslenia. Na týchto princípoch by mala byť postavená aj edukačná robotika na našich vyučovacích hodinách.



Robotické stavebnice a programovateľné hračky

Pre tento kurz sme vybrali robotickú stavebnicu LEGO MINDSTORMS NXT, ktorá je na našich školách v mnohých prípadoch už zakúpená. Domnievame sa, že je dobrou pomôckou pre robotické vyučovanie a je vhodná ako pre druhý stupeň ZŠ, tak aj pre SŠ. Okrem tejto stavebnice však existuje aj celý rad ďalších platforiem vhodných pre edukačnú robotiku.

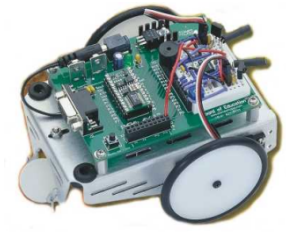
Programovateľné hračky

Takéto hračky môžu vykonávať sekvenciu jednoduchých príkazov (pohyb rôznymi smermi, otočenie). Rozvíjajú algoritmické myslenie a plánovanie už v predškolskej príprave.

LEGO WeDo

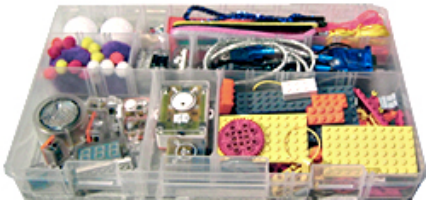
Stavebnica je určená deťom od 7 do 11 rokov. Okrem kocky obsahuje dva senzory a jeden motor. Programovací jazyk je obrázkový a má iba základné príkazy.

BoeBot



Táto stavebnica je určená nadšencom pre elektrotechniku. Programuje sa v jazyku podobnom BASICu a diely pripomínajú stavebnicu Merkúr.

Pico Crickets



Trochu iný prístup k robotike ponúka stavebnica, ktoré obsahuje veľa materiálu na tvorivé a umelecké ozvláštnenie robotických modelov. Programuje sa v blokovom jazyku Scratch.

Tvorivá robotika a robotické súťaže

Na MIT (Massachusetts Institute of Technology) sa skupina vedcov venuje tvorivej robotike a skúma rôzne výučbové metódy, ktoré by mohli prispieť k lepšiemu učeniu sa prostredníctvom programovateľných modelov. Práve stavebnica Pico Cricket vznikla v súvislosti s týmto výskumom. Detský programovací jazyk Scratch, ktorý rovnako vznikol na tejto akademickej pôde, je ďalším dielikom, ktorý dopĺňa myšlienku zábavného programovania a tvorivej robotiky pre deti. Autori sa zameriavajú aj na zapojenie dievčat do aktivít, ktoré bývajú často považované za chlapčenské. Zásady podporujúce tvorivú robotiku podľa M. Resnicka a jeho kolegov [3]: **orientujte sa na tému, kombinujte umenie a inžiniersky prístup, vytvorte priestor pre rozprávanie príbehov, organizujte výstavy.**

Robotika je populárnou súčasťou voľnočasových aktivít. Mnoho detí sa stretáva počas detstva s rôznymi stavebnicami a vnímajú ich ako hračky. **Robotické súťaže** môžu byť vhodným rozšírením školskej robotiky. Svojou lákavou formou dokážu pritiahnuť mnoho nadšených súťažiacich aj divákov.

Zhrnutie

Dozvedeli sme sa o tom, čo je to robotika a aké je jej miesto vo vzdelávaní. Vysvetlili sme si prečo budeme používať vybranú stavebnicu LEGO a dozvedeli sme sa aj o iných robotických platformách.

Stavebnice svojou podstatou nabádajú k netradičnej organizácií vyučovania a využívaniu výučbových metód, ktoré vedú žiakov k

- rozvoju tímovej práce,
- porozumeniu a navrhovaniu zložitých funkčných modelov, ktoré často simulujú zariadenia alebo javy z reálneho sveta,
- zberu, spracovaniu a prezentácií dát z prostredia,
- riešeniu problémov, ktoré sú neznáme a vyplývajú z konkrétnych požiadaviek na funkčnosť modelu v reálnom svete,
- aktívnemu manažmentu svojej práce a zadávaniu si vlastných cieľov,
- ku kreativite a prezentácií svojich nápadov a riešení.

Úloha

Nájdite na internete videá, v ktorých sú spomínané stavebnice a modely z nich predvedené v akcii.

Robotickým súťažiam a rôznym prístupom k edukačnej robotike sa budeme venovať v module *Didaktika robotických stavebníc (2DidRobo)*.

Prvé kroky s LEGO MINDSTORMS NXT

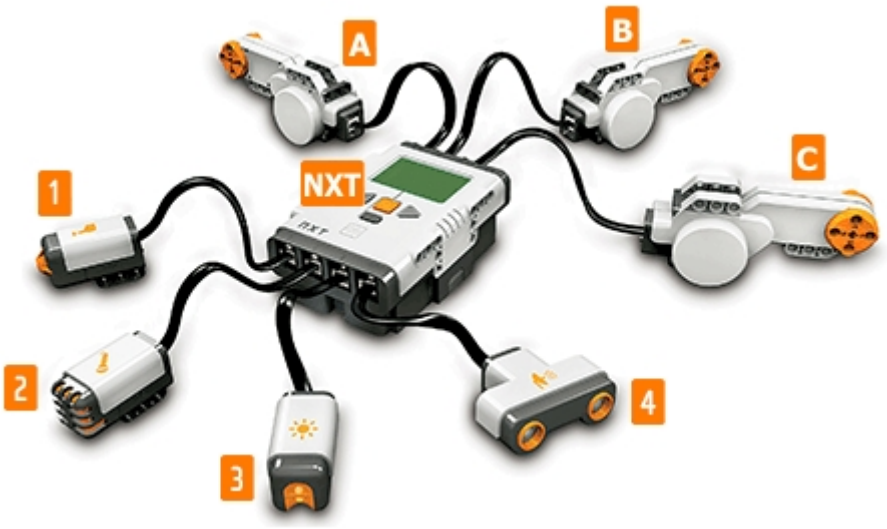
O stavebnici

Slovenský manuál k stavebnici Lego MINDSTORMS NXT nájdete na stránke: http://eduxe.cz/download/files/9797_lme_manual_sk.pdf

Stavebnicu tvoria rozmanité LEGO dieliky – rôzne osi, hriadele, kolieska a kocky. Od klasických stavebníc sa však líši ďalšími, **aktívnymi prvkami**:

- mozog stavebnice tvorí **programovateľná kocka** označovaná ako NXT,
- pohyblivého robota získame pripojením **motorov** ku kocke,
- robot „vníma“ svoje prostredie pomocou rôznych **senzorov.**

Pripojenie senzorov a motorov podľa obrázku sa používa pri programovaní vo voľbe NXT program, s ktorou sa účastníci oboznámia v tejto kapitole.



Programovateľná kocka s pripojenými aktívnymi prvkami

Vstupné porty 1 – 4 slúžia pre zapojenie senzorov, výstupné označené A – C na pripojenie motorov alebo lampičky.

Motory

Motor vie počítať svoje otáčky. Presvedčte sa o tom pomocou programu ***Try Motor***. Motor však musí byť pripojený na port A.

Zadanie 1

Pripojte ku kocke, ktorá je použitá v základnom robote, senzory a motory podľa obrázka.

Dajte pozor na to, aby sa vám prepojovacie káble nezaplietli do kolies.

Pri programovaní budeme využívať len dva motory na pohyb robota. Zapojte ich do portov **B** a **C.**

Senzory

Robotická stavebnica obsahuje 4 základné senzory. Dotykový senzor, zvukový senzor, svetelný senzor a ultrasonický senzor.

Dotykový senzor dokáže rozpoznať prekážky v prostredí. Má dva stavy – môže byť zapnutý a vypnutý, resp. zatlačený a voľný.

Zadanie 2

Použite voľbu ***View*** v kocke a zistite, aké hodnoty sa zobrazia na displeji kocky pri zapínaní a vypínaní senzora.

Zvukovým senzorom dokáže robot rozpoznať intenzitu zvuku. Tú meria v percentách za použitia dB (decibelov – všetky zvuky vrátane zvukov, ktoré nie je počuť ľudským uchom), alebo dBA (len zvuky ktoré zachytí aj ľudské ucho).

Zadanie 3

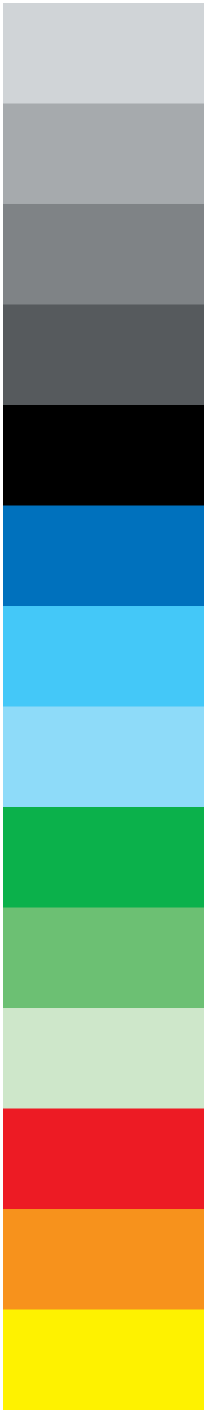
Použitím voľby ***View*** v kocke zistite úrovne zvuku v učebni pri tichu, tlmenej vrave, ako aj pri hlasnom rozprávaní sa, tlieskaní a pod.

Skúste fúknuť do zvukového senzora.

Zamyslite sa	Akú hodnotu nameria? Prečo?

Svetelný senzor umožňuje robotovi rozpoznať kvality osvetlenia a farby, tie však rozlišuje len ako odtiene šedej. Takisto ako zvukový senzor, aj svetelný senzor meria hodnoty v percentách.

Využite farebnú paletu pri práci so svetelným senzorom.



Zadanie 4	Pomocou palety farieb a voľby ***View*** v kocke zistite, pre ktoré farby vráti senzor aké hodnoty. Takisto zistite pre ktorú farbu vráti senzor najvyššiu hodnotu a pre ktorú najnižšiu.

Pomocou **ultrasonického senzora** môže robot merať vzdialenosť k objektom a rozpoznávať pohyb. Senzor pracuje na princípe odrazeného ultrazvuku. Z času, ktorý uplynie medzi vyslaním zvukového signálu a jeho prijatím určí vzdialenosť prekážky s presnosťou na 3 cm. Namerané hodnoty si môžeme nechať zobrazovať v centimetroch (cm), ale aj v palcoch (inch).

Zadanie 5	Použite voľbu ***View*** v kocke a zistite, akú najmenšiu a najväčšiu vzdialenosť od prekážky dokáže robot zaznamenať pomocou ultrasonického senzora. Skúste zmerať vzdialenosť nejakého malého predmetu

Senzory a programy

Ako môžeme využiť senzory pri definovaní správania robota? Vyskúšajme si demoprogramy pripravené v kocke.

Zadanie 6	Použite voľbu ***Try me*** v kocke. Postupne vyskúšajte jednotlivé programy. Ako sa mení správanie robota v závislosti od rôznych vstupov senzora? Zapnite si napríklad program ***Try Light*** a skúste prechádzať svetelným senzorom nad paletou farieb.

Vytvárame malé programy v kocke

Jednoduché programy, ktoré riadia motory a reagujú na senzory, možno programovať priamo v kocke pomocou jej klávesnice a displeja. Použijeme ponuku kocky ***NXT Program***.

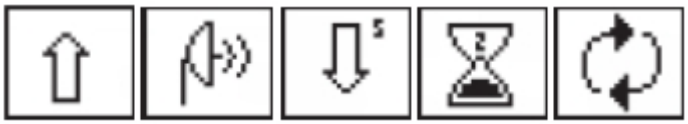
Zadanie 7	V ponuke kocky zvoľte časť ***NXT Program*** a zadajte nasledujúci program:  Zistite, čo program vykonáva.

Všimnite si, že pri zadávaní programu do kocky nemôžete zadať ľubovoľnú postupnosť príkazov, ani ľubovoľne dlhý program. Programové obmedzenie je na 5 blokov, pričom na každom mieste v programe môžu byť len určité príkazy. Na prvej a tretej pozícii môžeme použiť príkazy pohybu a hrania melódie. Na druhej a štvrtej pozícii zas len príkazy čakaj na čas alebo na hodnotu senzora. Na poslednej piatej pozícii v programe máme na výber príkazy stop a opakuj, ktoré program buď zastavia alebo ho budú vykonávať donekonečna.

Všimnime si: rozsah programu v kocke je značne obmedzený. Môžeme vytvárať iba krátke, maximálne 5-príkazové programy.

Zadanie 8

Zadajte do NXT kocky nasledujúci program:



Zistite, čo program vykonáva.

Všimnite si, že program je takmer identický z predchádzajúcim zadaním. Vedeli by ste teraz popísať ako fungujú príkazy pohybu?

Zistite, čo vykonáva tento program:

Zadanie 9

Zadajte do NXT kocky nasledujúci program a spustite ho.

Ako treba program upraviť, aby robot šiel dopredu stále, pokiaľ nenarazí na tmavý podklad?

V programe sme použili ďalšie dva programové bloky, ktoré využívajú svetelný senzor. V akých ďalších programoch by ste ich vedeli využiť?

Vytvárame vlastné programy

Zadanie 10

Vytvorte takýto program:

Robot bude viesť nevidiacich ľudí po bielom chodníku. Keď pred sebou nájde čiernu cestu, zapípa a zastaví sa. Nevidiaci človek teraz vie, že bude prechádzať cez cestu.

Zadanie 11

V horúcom letnom dni každý milovník leta s úsmevom na tvári mieri k stánku so zmrzlinou. Avšak hneď ako sa na oblohe objaví mráčik, náš zmrzlinový fanúšik stráca sebadôveru a vracia sa späť domov. Nič si z toho netreba robiť – milovníci leta sú bezstarostné tvory – slnko na oblohe ich vždy nájde na ceste k zmrzlinovému stánku.

Úloha: kým robot vidí svetlo, pohybuje sa rovno dopredu. Keď však narazí na tmu, začne cúvať. Situácia sa opakuje, keď na robota znovu zasvieti svetlo.

Zadanie 12

Náš robot sa stratil v bludisku a nevie ako má ísť von. Už je z toho nešťastný, ale nakoniec si spomenie na to, čo sa naučil v škole: stačí, keď sa bude držať jednej steny a nakoniec východ nájde.

Úloha: vytvorte program, v ktorom bude robot chodiť rovno dopredu, pokiaľ nezbadá pred sebou prekážku. Potom sa pootočí doprava a znova skúsi ísť dopredu.

Zadanie 13

Robot športovec sa pripravuje na letnú olympiádu v bežeckej disciplíne. Trénuje si hlavne presný štart.

Úloha: naprogramujte robota tak, aby čakal na zvukový signál. Po jeho zaznení sa začne pohybovať smerom dopredu až kým nepríde do cieľa (tmavý podklad).

Zadanie 14	Vytvorte skupiny po dvoch robotoch. Naprogramujte štafetu robotov nasledovne: Prvý robot vyrazí na zvukový signál dopredu. Oproti nemu v istej vzdialenosti stojí druhý robot. Keď sa roboty k sebe dostatočne priblížia (zistia to pomocou ultrasonického senzora), tak prvý robot zastane a druhý robot sa začne pohybovať dozadu, pokiaľ neprejde cieľom (tmavý podklad).
Zadanie 15	Už poznáte základné princípy správania sa NXT robota. Vymyslite program, ktorý robot zrejme zvládne vykonať, no pomocou programovania priamo v kocke ho nedokážete vytvoriť. Aké možnosti vám pri programovaní v kocke chýbajú?

Programovacie jazyky pre LEGO NXT

V nasledujúcej kapitole sa budeme učiť programovať NXT kocku v jazyku LEGO MINDSTORMS Education NXT. Toto programovacie prostredie však nie je jedinou možnou voľbou.

Okrem jazykov podobných C sú na internete na stiahnutie aj iné programovacie jazyky a prostredia, ktoré dokážu spolupracovať s NXT kockou.
Prehľad je napríklad na stránke:
en.wikipedia.org/wiki/Lego_Mindstorms

Bricx Command Center (BricxCC)

je programovacie prostredie, v ktorom sa dajú tvoriť programy pre množstvo robotických platforiem. Jazyk je založený na jazyku C a nazýva sa NXC ("Not eXactly C", nie presne C). Aktuálna verzia je 3.3. Toto prostredie je vytvárané ako otvorený softvér a je voľne dostupné na stiahnutie na stránkach bricxcc.sourceforge.net.

ROBOTC 2.0

RobotC je komerčné programovacie prostredie pre NXT aj RCX kocky vyvíjané v spolupráci s Carnegie Mellon University. Programovací jazyk je opäť založený na C. Skúšobnú verziu si môžete siahnuť na stránke www.robotc.net/download/nxt/.

Okrem samotného programu je pre toto prostredie pripravené množstvo materiálov pre učiteľa aj pre študentov (zadania, pracovné listy, riešené príklady a pod.).

Zhrnutie

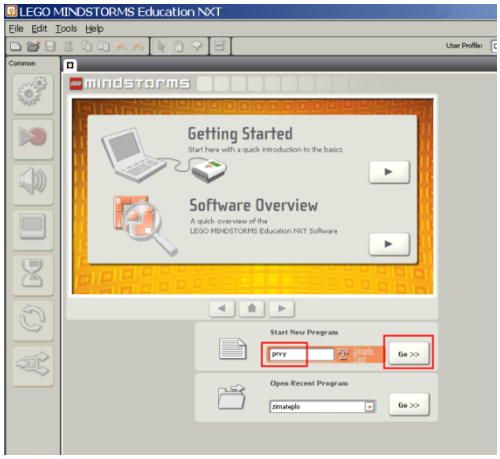
Poznáme aktívne prvky stavebnice Lego Mindstorms a vieme, ako ich pripojiť k programovateľnej kocke. Pomocou senzorov dokážeme namerať rôzne vlastnosti prostredia. Vytvárame jednoduché programy priamo v programovateľnej kocke.

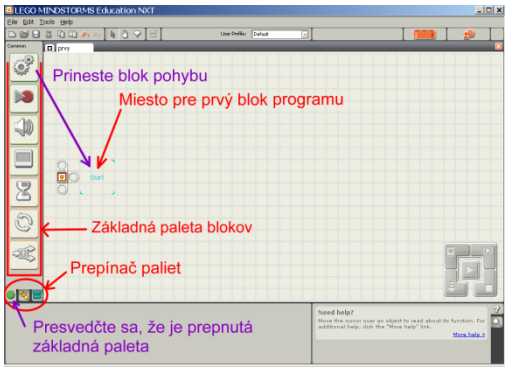
Programovanie v ikonickom prostredí LEGO MINDSTORMS Education NXT

V tejto kapitole sa budeme venovať programovaniu v prostredí programu LEGO MINDSTORMS Education, ktoré umožňuje vytvárať oveľa zložitejšie a zaujímavejšie programy pre robotov riadených kockou NXT.

Prvé zoznámenie s programom

Program je súčasťou dodávky robotov a dá sa jednoducho nainštalovať z príslušného CD nosiča. Po jeho spustení uvidíme úvodnú obrazovku programu (obrázok vľavo).



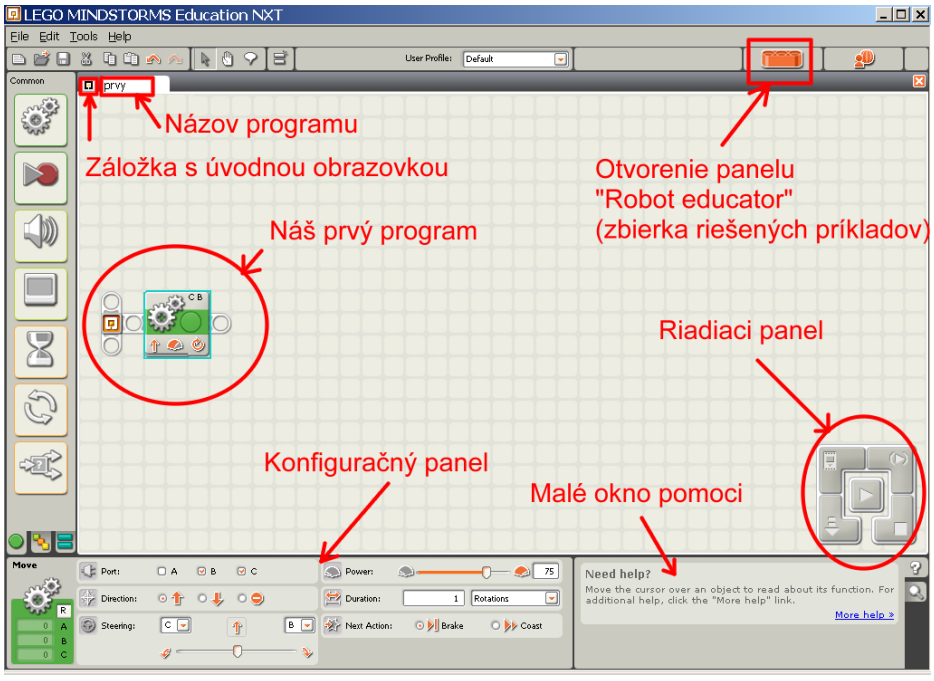


V ikonickom prostredí sa programuje pomocou blokov, ktoré sa kladú na obrazovku v istom poradí a istým spôsobom. Bloky majú význam príkazov programovacieho jazyka, preto budeme v materiáli používať slová blok aj príkaz podľa toho, či myslíme konkrétny grafický objekt na obrazovke alebo skôr to, čo vyjadruje.

Na začiatku kapitoly budeme používať len bloky zo základnej palety, preto sa presvedčte, či máte prepnutú základnú paletu.

S programom začneme pracovať tak, že do textového políčka v pod Start New Program napíšeme meno nového programu (napíšme **prvy**, keďže je to náš prvý program v tomto module) a stlačíme tlačidlo Go (označené červeným rámikom na našom obrázku vľavo). Tým sa dostaneme do základnej pracovnej obrazovky programu (obrázok vpravo). Najväčšiu plochu v nej zaberá štvorčekovaná **pracovná oblasť**. V nej budeme vytvárať program z grafických blokov, ktoré budeme do pracovnej plochy ťahať z **palety**. Ako prvý skúsme potiahnuť blok pohybu (prvý v palete) na miesto označené ***Start***.

Bloky sa dajú ťahať z palety dvomi spôsobmi. Buď „dospeláckym“ spôsobom: na bloku stlačím ľavé tlačidlo myši, ťahám blok pričom mám stále stlačené tlačidlo a pustím ho až na mieste, kam chcem blok položiť. Alebo „detským“ spôsobom: kliknem na blok, tým sa mi prilepí na kurzor myši, potom ťahám myš bez držania tlačidla až na miesto určenia a tam znovu kliknem ľavým tlačidlom, čím blok položím.

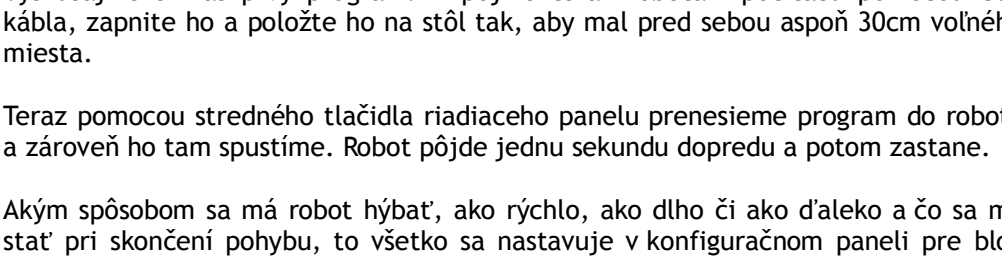


V malom okne pomoci je krátka anglická pomôcka o tom bloku alebo inom prvku okna, nad ktorým sme postáli myšou (nemusí to byť ten blok, ktorý je označený. Pre stránku s kompletným popisom bloku (v angličtine) kliknime potom na modrý nápis ***More help***.

Tým sme vytvorili náš prvý program. Blok pohybu prikazuje robotu zapnúť motory tak, aby sa pohol nejakým smerom a nejakou rýchlosťou.

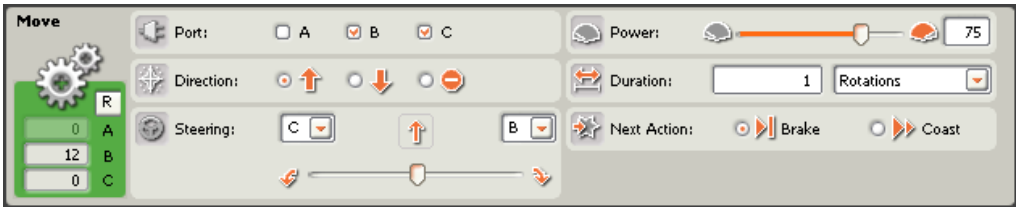
Vyskúšajme si náš prvý program. Pripojme teraz robota k počítaču pomocou USB kábla, zapnite ho a položte ho na stôl tak, aby mal pred sebou aspoň 30cm voľného miesta.

Teraz pomocou stredného tlačidla riadiaceho panelu prenesieme program do robota a zároveň ho tam spustíme. Robot pôjde jednu sekundu dopredu a potom zastane.



Tlačidlo na prenesenie a spustenie programu.

Akým spôsobom sa má robot hýbať, ako rýchlo, ako dlho či ako ďaleko a čo sa má stať pri skončení pohybu, to všetko sa nastavuje v konfiguračnom paneli pre blok pohybu:



Konfiguračný panel pre daný blok sa ukáže keď je ten blok označený (klikli sme naň).

Ak dĺžka USB kábla nedovoľuje robotovi dostatočný pohyb, alebo ide o program, kde sa robot musí voľne pohybovať a otáčať, použite na prenesenie programu pravé dolné tlačidlo riadiaceho panelu.

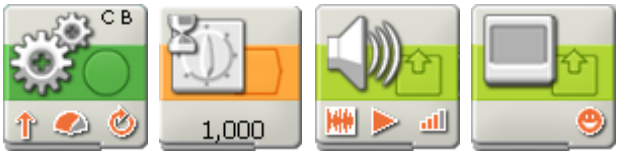
Zadanie
Pohyby

Experimentujte s nastaveniami pohybu. Skúste dosiahnuť aby robot:

- išiel dopredu veľmi pomaly,
- zatočil doľava,
- cúval,
- točil sa na mieste (pozor na zamotanie kábla!).

Pohyby a čakanie

V tejto podkapitole naprogramujeme niekoľko programov, kde hlavnou témou budú rôzne druhy pohybu. Budeme však používať aj niektoré ďalšie bloky:



Tým sa program len prenesie, ale nespustí. Po odpojení kábla môžete robota odniesť na vhodné miesto a tam program spustiť Pomocou jeho menu. V hlavnom menu zvolíte ***My Files***, potom ***Software files*** a potom nájdete program podľa mena (v našom prípade to je **prvy**), zvolíte ho a ďalším stlačením oranžového tlačidla ho spustíte.
NXT sa Vám snaží pomôcť a v menu ***Software files*** je označený ten program, ktorý ste do neho posledne preniesli. Takže pri dlhšej práci s tým istým programom netreba ísť neustále cez veľa menu.

Naše ukážkové zadanie:

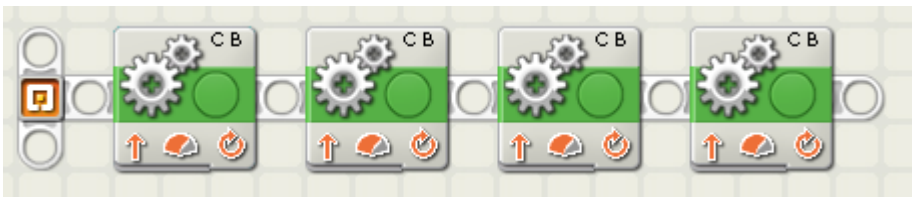
Zadanie
Málo paliva

Robot vyštartuje rýchlo, dochádza mu palivo, preto postupne spomaľuje až zastane, na displeji ukáže ikonu benzínovej pumpy a vyjadrí zvukom svoje zdesenie nad tým, že mu došlo palivo.

Najprv si ukážeme si viac verzií riešenia samotného spomaľovania, až potom dokončíme celý program.

1. Prvý pokus: Použime niekoľko blokov pohybu za sebou, prvý bude prikazovať ísť rýchlo, každý ďalší z nich bude prikazovať ísť pomalšie a pomalšie.

Začnime tým, že do programu naťaháme štyri bloky pohybu:



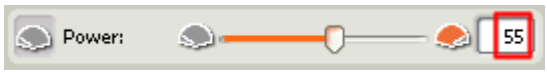
Ak máte vo Windows nastavené správne regionálne nastavenia pre Slovensko, tak musíte pri zápise desatinných čísiel používať čiarku, nie bodku ako je to bežné v iných programovacích jazykoch.

Ak máte na stole málo miesta, môžete zvoliť ešte menšie číslo pre trvanie pohybu – 0,3 alebo aj 0,2 otáčky.

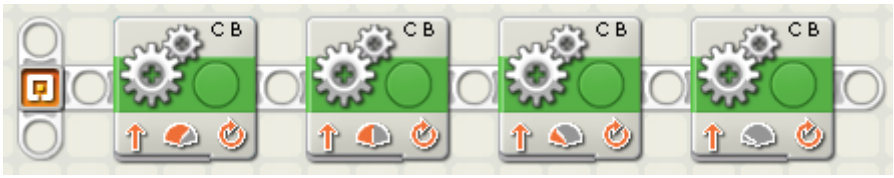
Všimnite si, že stredná ikona bloku znázorňuje nastavenú rýchlosť.

Aby robot nezašiel príliš ďaleko, zmenšime mu dráhu tak, že namiesto jedného otočenia kolesa každému bloku nastavíme len pol otáčky (nastavenie ***Duration***):

Druhému bloku potom znížime v nastavení ***Power*** rýchlosť pohybu na 55, ďalšiemu na 35 a poslednému 15.



Náš výsledný program bude takýto:



Prenesme ho do robota a spustime. Vidíme, že robot ide postupne pomalšie a pomalšie, ale po každom úseku pohybu zastane. Celá akcia je teda trhaná a nevyzerá ako spomaľovanie, skôr je to pohyb prískokmi vpred.

2. Jemnejšie prechody vďaka dobehnutiu motora

Prudké zabrzdenie na konci vykonávania každého bloku pohybu spôsobuje nastavenie ***Brake*** (zabrzdi).

Vďaka nemu motory na konci každého pohybu prudko zabrzdia tak, aby bola prejdená vzdialenosť čo najpresnejšia.

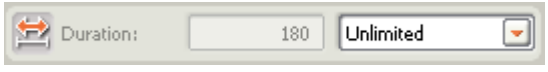
Skúsme ho prepnúť na ***Coast*** (dobehni). Ten na konci pohybu necháva motor postupne dobehnúť. Prepnime ho teda každému zo štyroch blokov na ***Coast***.

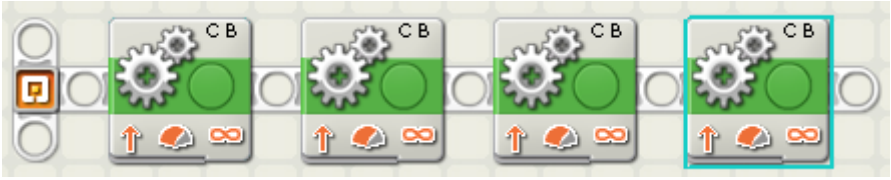
Skúsme, čo spraví robot. Vidíme, že vďaka spomaľovaniu pri funkcii ***Coast*** nadväzujú bloky na seba plynulejšie. Pretrváva však problém, že ďalší blok pre pohyb sa začne vykonávať až keď sa pohyb podľa predošlého bloku skončí. Preto aj tak po vykonaní každého bloku najprv robot úplne zastaví a až potom sa zase pohne.

3. Nadväzovanie pohybov bez zastavenia

Ako teda dosiahnuť plynulé nadväzovanie pohybov bez toho, aby robot zakaždým najprv zastavil? Všimnime si že jedna z možností určenia trvania pohybu je aj ***Unlimited*** (neobmedzená). Čo to znamená a ako nám to môže pomôcť?

Skúsme upraviť program tak, aby každý blok mal nastavené trvanie na ***Unlimited***:





Vidíme, že pravá ikona sa zmenila na symbol nekonečna. Bude sa teda náš robot pohybovať nekonečne dlho? Alebo dokonca štyrikrát nekonečne dlho?

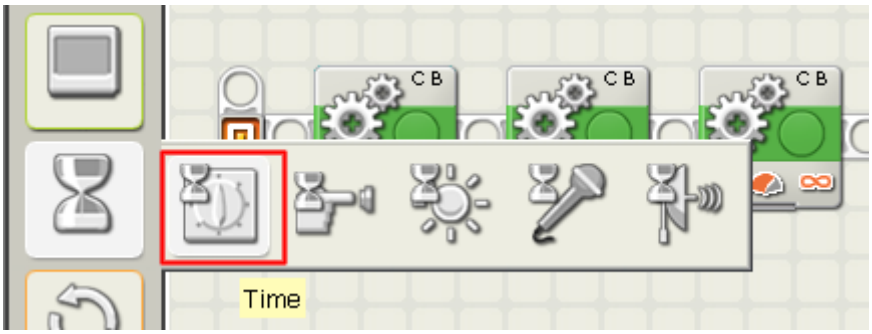
Blok pohybu s trvaním ***Unlimited*** by sa preto mal radšej nazývať „zapni motory“.

Skúsme teda spustiť program v robotovi a uvidíme, čo sa stane. Uvidíme, že robot sa pohne len o veľmi malý kúsok.

Je náš program chybný? Nie. To len ***Unlimited*** neznamená pohyb donekonečna. V skutočnosti blok len prikáže robotovi zapnúť motory na danú rýchlosť a smer. Potom ale na nič nečaká a v programe sa začne vykonávať nasledujúci blok.

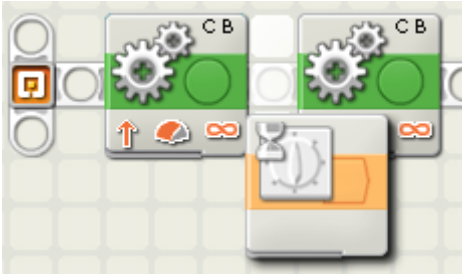
V našom programe boli použité za sebou štyri také bloky. Robot teda najprv nastavil rýchlosť 75, potom vzápätí rýchlosť 55. Ešte sa ani nestihol pohnúť a už nastavil rýchlosť 35 a hneď potom 15. Hneď za tým program skončil, a preto robot postupne dobehol z rýchlosti 15 na nulu. Keďže počítače sú rýchle, všetky štyri príkazy sa stihli vykonať skôr než sa motor mohol pohnúť a robot sa pohol len vďaka tomu, že sa všetky motory na konci programu zastavia metódou ***Coast*** (dobehni).

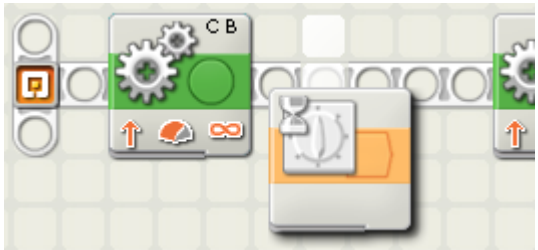
Aby sa robot medzi dvomi blokmi pohybu stihol aj pohnúť, musíme do programu medzi bloky pohybu pridať bloky čakania na uplynutie času. Bloky čakania nájdeme na palete keď v ponuke, ktorá sa rozbalí keď prídeme myšou na ikonu presýpacích hodín. Teraz chceme čakať na uplynutie času, preto zvolíme prvý blok v ponuke.



Pri nosení blokov do stredu programu je veľmi dôležité počkať, kým sa nám urobí v programe miesto, lebo ak blok pustíme predčasne, tak tým môžeme úplne pokaziť usporiadanie blokov na obrazovke. To platí predovšetkým pre dlhšie programy, ale naučiť sa to musíme už pri krátkych.

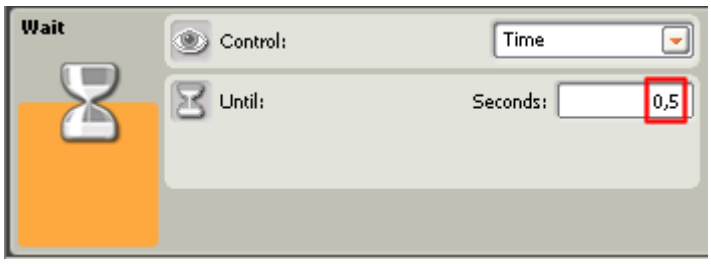
Blok potom položíme medzi prvý a druhý blok nášho programu tak, že ho tam pomaly nesieme až kým sa neukáže na správnom mieste svetlé miesto označujúce, kam bude blok položený (obrázok vľavo dole). Zastaneme s pohybom, ale blok nepúšťame. Počkáme, kým sa nám na tom mieste bloky program posunú tak, ako vidíme na obrázku vpravo dole. Až potom blok pustíme (alebo klikneme druhýkrát ak používame „detský“ spôsob ťahania blokov).





Novému bloku v jeho konfiguračnom paneli nastavme trvanie 0,5 sekundy:

Bloky v programe môžeme kopírovať tak, že stlačíme kláves ***Ctrl*** a potom stlačíme ľavé tlačidlo myši na bloku, ktorý chceme kopírovať a začneme ťahať myš bez pustenia jej tlačidla. Uvidíme, že ťaháme kópiu daného bloku. Tú potom položíme na želané miesto rovnako, ako keď ťaháme bloky z palety.



Rovnaký blok treba vložiť do programu ešte trikrát, tak aby bol za každým blokom pohybu. To môžeme spraviť nosením blokov z palety a nastavovaním času v každom z nich, alebo aj kopírovaním blokov v programe (viď postup v stĺpci vľavo), čím si ušetríme nastavovanie času pre každý blok.

Skúsme teraz preniesť program do robota a spustiť ho. Uvidíme, že pohyby už nadväzujú na seba bez toho, aby motor najprv zastavil.

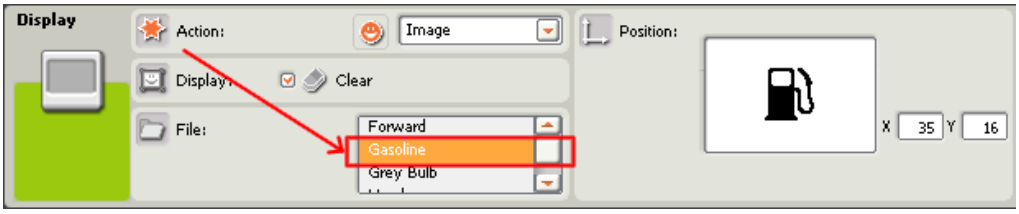
Pri vyberaní zvukov nám počítač každý z nich zahrá. Ak však v učebni nemajú počítače zvukové karty alebo nemajú pripojené reproduktory ani slúchadlá, môžeme si vypočuť zvuk tak, že ho necháme zahrať samotnej kocke NXT. Keď označíme v programe len blok zvuku, môžeme ho poslať na vykonanie do kocky NXT použitím pravého horného tlačidla riadiaceho panelu.

Našli sme teda dobrý spôsob ako naprogramovať postupné spomaľovanie robota. Pozrime sa ešte raz na úlohu blokov čakania v tomto programe. Z programovania počítačov sme zvyknutí, že keď program čaká, tak sa počas toho nemôže nič diať. Pri programovaní robotov to ale nemusí byť pravda. Keď sme zapli motory blokom s časom ***Unlimited*** a potom nasleduje blok čakania, tak síce program bude čakať, ale motory v tom čase stále pôjdu zadanou rýchlosťou až kým čakanie neskončí a oni nedostanú iný príkaz. Toto je typické pre programy, ktoré riadia nejaký hardvér (nielen robotov).

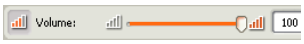
4. Dokončenie programu

Na tretí pokus sme teda našli dobré riešenie spomaľovania, teraz nám zostáva len dokončiť program podľa zadania. Na konci má robot ukázať na svojom displeji obrázok a niečo povedať.

Na ukázanie nejakého obrázku alebo aj textu či grafiky na obrazovke kocky používame blok zobrazenia. Keď ho prinesieme na koniec programu, tak v konfiguračnom paneli môžeme vybrať obrázok, ktorý sa má zobraziť na displeji. Vyberme si obrázok znázorňujúci stojan čerpacej stanice. Volá sa ***Gasoline***:



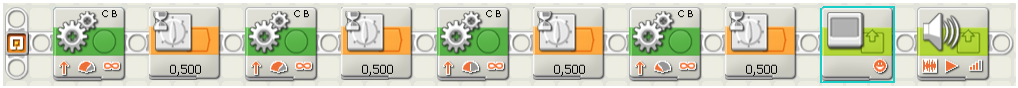
Ak sme v hlučnej triede, tak môžeme skúsiť pridať aj hlasitosť až na 100%.



Na zahranie zvuku slúži blok zvuku. Pridajme ho na koniec programu a v jeho konfiguračnom paneli si vyberme zvuk ***Ahnoo***:

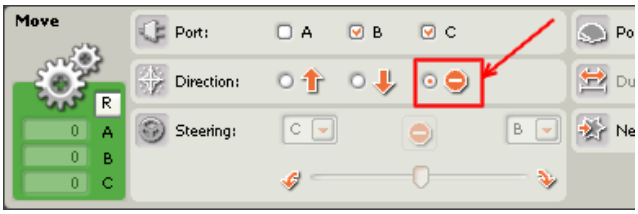
Aj obrázky z displeja zmiznú na konci programu. Preto ak ho chceme ukázať po dlhšiu dobu, než len počas trvania zvuku, pridajme na koniec programu ešte čakanie.

Tu je náš program po úpravách:

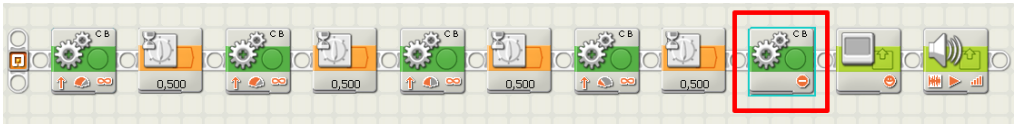


Vyskúšajme ho. Uvidíme, že ešte má malú chybičku: na posledný z blokov pohybu prikáže motorom ísť veľmi pomaly dopredu a oni stále pôjdu až do konca programu. Preto pre vrčanie motorov možno nebudeme vôbec počuť, čo robot povedal. A aj pôvodné zadanie hovorilo, že robot má najskôr zastaviť a až potom ukázať obrázok a zabedákať.

Motorom prikážeme zastaviť, ak do programu ešte pridáme blok pohybu, v ktorom zvolíme v časti ***Direction*** ikonu jednosmernej cesty.



Taký blok pohybu zastavuje motory a mohli by sme ho preto nazývať aj „vypni motory“.



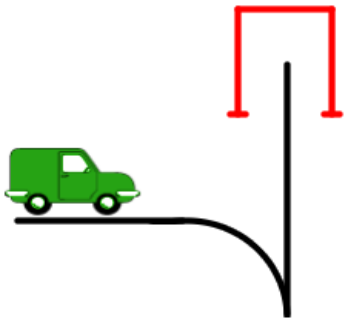
Teraz už zvuk motorov nebude rušiť hlas robota a tým sme naprogramovali náš prvý vzorový program.

Skúste samostatne naprogramovať nasledujúce zadania:

Zadanie

Zacúvaj 1

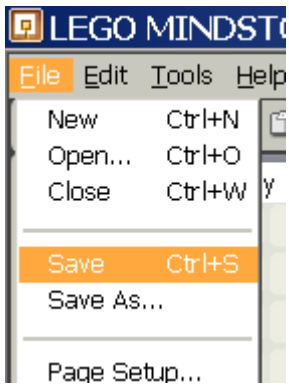
Robotické autíčko má zacúvať do parkovacieho miesta podľa obrázka:



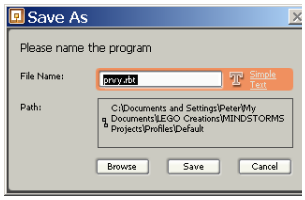
Teda má ísť najprv kúsok rovno, potom po oblúku doprava dopredu, až kým sa neotočí o 90 stupňov vzhľadom k pôvodnému smeru. Potom má na chvíľu zastaviť a nakoniec pomaly zacúvať rovno dozadu do boxu. Po zastavení má ukázať na displeji obrázok zatvoreného zámku na 3 sekundy.

Tip: musíte vyskúšať, ako dlho ísť dopredu a ako prudko doprava aby robot skončil približne 90 stupňov otočený od pôvodného smeru, tak, akoby naozaj zašiel správne do parkovacieho miesta.

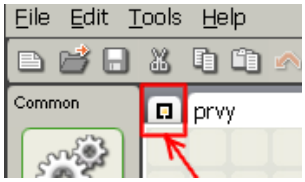
Ukončený program môžete zapísať na disk stlačením ***Ctrl+S*** alebo voľbou ***File/Save*** z menu.



Keď ukladáte program prvý raz, tak počítač žiada potvrdiť jeho meno. Je tam ale už predvolené meno, pod ktorým sme program vytvárali.



Nový program vytvoríte najpohodlnejšie tak, že prejdete na úvodnú obrazovku pomocou prvej záložky a tam napíšete meno programu a stlačíte ***Go***.



Stlačenie ***Ctrl+N*** je rýchly spôsob vytvorenia nového programu, ale dostane škaredý názov ***Untitled-1*** (pod ním ho potom budete hľadať v kocke).

Zadanie **Modelka**	Robot je modelka - má prejsť po móle dopredu istú vzdialenosť, potom sa pomaly otočiť dokola na mieste o 1 a pol otáčky (o 360 + 180 = 540 stupňov) teda tak, že zostane stáť otočený smerom späť odkiaľ prišiel, chvíľku počkať a vrátiť sa späť (ísť rovno späť do počiatočného miesta).
Pomôcka 1	Robot sa točí na mieste, keď posúvač zatáčania posuniete do krajnej polohy
Pomôcka 2	Koľko otáčok motora treba na otočenie vášho robota o 540 stupňov, to musíte vyskúšať. Môže sa to líšiť pre každého jedného robota. **Pozor:** stupne, ktoré sa dajú nastaviť v konfiguračnom paneli, definujú o koľko sa otočí oska motora, nie o koľko sa otočí celý robot.

Použitie senzorov

V tejto podkapitole sa naučíme ako môže náš robot jednoducho reagovať na podnety z prostredia pomocou svojich senzorov.

Začnime týmto zadaním:

Podobné programy sme programovali aj priamo v kocke NXT, ale tam sme nemohli definovať presnú hodnotu senzora, ktorá rozlišuje svetlo od tmy, hluk od ticha či ďaleko od blízka.

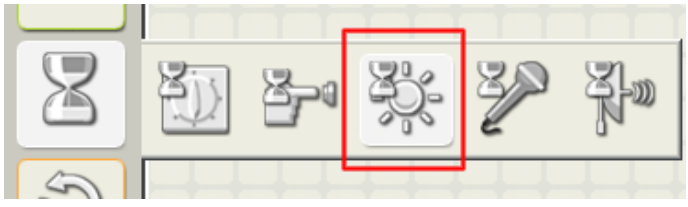
Zadanie **Chodník 1**	Robot má viesť nevidiaceho po chodníku svetlej farby a má zastaviť keď nájde cestu, ktorá je tmavšej farby a vydať varovný zvukový signál.

Najprv si pripravme pre robota testovacie prostredie. Nájdime rovnú plochu svetlej farby dlhú aspoň 30cm a na nej urobme čierny pásik (namaľujme alebo nalepme čiernu lepiacu pásku).

Teraz chceme robotovi povedať aby išiel kým neuvidí pred sebou tmavú farbu. To sa v jazyku robota povie najjednoduchšie takto:

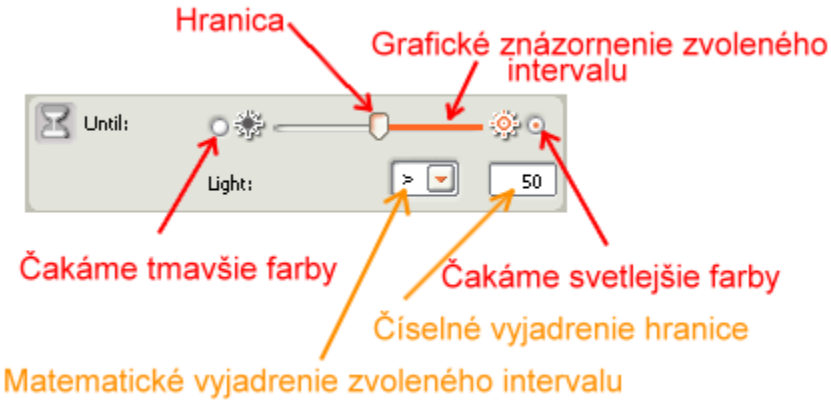
Zapni motory, čakaj kým bude svetelný senzor nad tmavou farbou, vypni motory.

Ako zapnúť a vypnúť motory, to už vieme. Medzi tieto dva príkazy dáme nový blok pre čakanie na stav svetelného senzora, ktorý nájdeme na palete v zozname čakacích blokov:



Takže náš program bude takýto:

V časti ***Until*** konfiguračného panelu svetelného senzora treba nastaviť, na akú hodnotu chceme čakať. Určíme hranicu a to, či čakáme na farby tmavšie než hranica alebo na farby svetlejšie:



Ako zistiť hraničnú hodnotu? Všimnime si, že v ľavom dolnom rohu konfiguračného panelu bloku čakania na svetelný senzor vidíme momentálnu hodnotu senzora pripojeného robota. Vďaka tomu vieme ľahko zistiť hraničnú hodnotu, ktorá rozlíši svetlý chodník od tmavej cesty. Postavme robota tak, aby bol jeho senzor nad svetlým chodníkom a zapíšme si hodnotu (napríklad 59). Potom ho postavme tak, aby mal senzor nad tmavou cestou a znova si zapíšme hodnotu (napríklad 27). Za hranicu zvoľme číslo niekde medzi týmito dvomi hodnotami, najlepšie priemer týchto dvoch čísiel (v našom prípade 43). Zistené číslo nastavme pomocou posúvača alebo zapíšme do textového okienka.

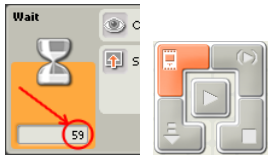
V našom prípade chceme čakať na farby tmavšie než zistená hranica, preto musíme kliknúť na bodku vedľa čierneho slniečka (alebo vybrať znamienko menší):

43 je naša hodnota, vaša hodnota, ktorú ste zistili, bude asi iná.

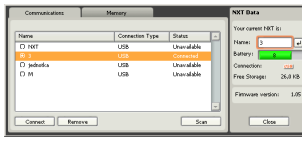
Takto má vyzerať kompletná konfigurácia bloku čakania na svetelný senzor:

Aby bol náš program úplný, pridajme ešte varovný zvukový signál. Umiestnime preto do programu blok zvuku a vyberieme niektorý zo zvukových signálov, napríklad ***! Error 02*** alebo ***! Startup***, prípadne iný.

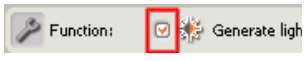
Aby však nezanikol vo zvuku motorov, počkajme pred jeho spustením krátky čas, napríklad pol sekundy. Takto má vyzerať hotový program.



Ak je políčko šedé, skontrolujte, či je pripojená a zapnutá kocka a či je na nastavenom porte pripojený senzor.
Ak políčko stále nereaguje, tak sa ešte počítač nespojil s kockou. Spojenie vytvoríte buď tak, že do kocky pošlete nejaký program, alebo stlačením ľavého horného tlačidla riadiaceho panelu vyvoláte okno kocky NXT, tam skontrolujete, či ste spojení a stlačíte jeho tlačidlo ***Close***.



Dôležité pre náš model je, aby svetelný senzor aj svietil, teda aby meral odrazené svetlo. Preto je zaškrtnuté ***Generate light***.

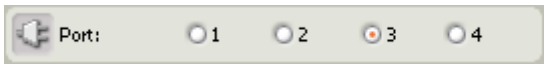


Zvuky, ktorých meno začína výkričníkom, sú technické zvuky, ostatné položky označujú vyslovené slová alebo zvuky vydávané človekom.

Vyskúšajte jeho správnu funkciu pre rôzne štartové pozície robota (rôzne vzdialenosti od cesty).

Podobne sa používajú aj ďalšie senzory. Príslušné bloky nájdeme všetky pod ikonou čakania na palete príkazov:

Všimnime si, že všetky tieto bloky majú pri ich položení do programu nastavené číslo vstupného portu (zobrazené v pravom hornom rohu bloku), na ktorý sa zvykne pripájať daný druh senzora podľa návodu na zapojenie z 2. kapitoly. Teda ak nepoužívame viac senzorov rovnakého druhu, je dobré ich zapájať na vstupné porty tak ako ukazuje obrázok v 2. kapitole. Senzor môžeme zapojiť aj na iný port, ale vtedy budeme musieť zmeniť číslo vstupného portu v nastaveniach bloku.



Zadanie **Vyhni sa**	Robot ide dopredu, kým nie je blízko prekážky (zistí to ultrasonickým senzorom), potom ide dozadu a doprava, kým nenarazí tlakovým senzorom na prekážku.

Kalibrácia senzorov

Svetelný aj zvukový senzor udáva nameranú hodnotu v percentách. Jeho hodnotou teda nie je absolútna hodnota osvetlenia alebo hluku, ale relatívna hodnota, kde 0 je najmenšia bežná hodnota (minimum) a 100 je najväčšia bežná hodnota (maximum). V základnom nastavení je minimum a maximum nastavené v zhode s najmenšou a najväčšou hodnotou, ktorú dokáže daný senzor fyzicky odmerať. To však niekedy nemusí vyhovovať, preto možno nastavenie minima a maxima zmeniť v procese, ktorý nazývame kalibrácia.

Kalibrácia má dve použitia. Niekedy potrebujeme zvýšiť citlivosť merania aj za cenu zmenšenia jeho rozsahu. To je napríklad vtedy, keď by nám pokus v zadaní Chodník dal hodnotu 45 pre chodník a 47 pre cestu (teda farby by sa v stupnici 0 až 100 líšili len veľmi málo). Keby sme za hranicu zvolili 46, tak by sa ľahko mohlo stať, že zašpinenú časť chodníka by robot vyhodnotil ako cestu alebo naopak svetlejšiu časť cesty by vyhodnotil ako chodník. Ak ale pomocou kalibrácie nastavíme minimum na úroveň cesty a maximum na úroveň chodníka, tak potom bude na chodníku senzor dávať hodnotu 100 a na ceste hodnotu 0. Hranicu môžeme potom nastaviť na 50 a budeme rozlišovať oveľa presnejšie cestu a chodník.

Kalibrácia sa dá použiť aj počas behu programu. Slúži na to blok kalibrácie, ktorý sa nachádza v rozšírenej palete, preto ho v tomto materiáli bližšie nepopisujeme.

Druhé použitie kalibrácie je vtedy, keď pomocou nej chceme zjednotiť chovanie robota v rôznych svetelných alebo hlukových podmienkach. Program teda vždy naprogramujeme tak, že bude predpokladať kalibrovanie na konkrétne podmienky tak, že 0 bude najmenšia hodnota (napr. hladina okolitého hluku alebo osvetlenia) a 100 bude najvyššia hodnota (najsilnejší zvuk alebo najsvetlejší objekt, s ktorým potrebuje pracovať). Pred spustením robota v iných podmienkach potom stačí kalibrovať senzory a netreba meniť program.

Postup kalibrácie si ukážeme na príklade zvukového senzora.

Zadanie

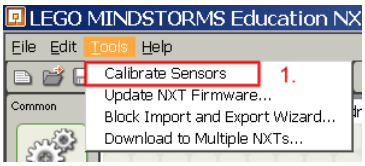
Kalibrácia zvukového senzora

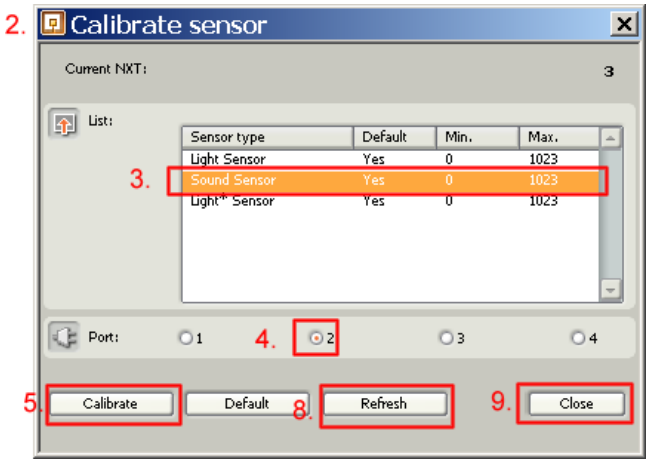
Skalibrujte zvukový senzor tak, aby pre bežnú hladinu hluku v triede dával hodnotu 0 a pre najsilnejší zvuk, ktorý dokážeme vydať, dával hodnotu 100.

Zvukový a svetelný senzor vie vnútorne rozlíšiť až 1024 rozdielnych úrovní

Postup kalibrácie:

1. Pri kalibrácii potrebujeme mať zapnutú a pripojenú kocku NXT s pripojeným senzorom. V hlavnom menu programu zvolíme položku ***Tools*** (nástroje) a v nej ***Calibrate Sensors***.





Ak je okno kalibrácie celé šedé, znamená to, že nemáme pripojenú kocku k počítaču.

Ak je v stĺpci ***Default*** hodnota ***Yes***, znamená to, že senzor ešte nebol kalibrovaný. No znamená, že už bol kalibrovaný. Tlačidlom ***Default*** môžeme kalibráciu senzora zrušiť.

V dialógu vidíme, že interné hodnoty senzorov sú čísla medzi 0 a 1023, citlivosť senzora je teda vyššia než 101 hodnôt, ktoré vidíme v programoch.

2. Objaví sa nám okno kalibrácie (vpravo). V ňom vidíme tri typy senzorov, ktoré sa dajú kalibrovať. ***Light Sensor*** označuje svetelný senzor NXT, ***Sound Sensor*** označuje zvukový senzor NXT a ***Light* Sensor*** označuje svetelný senzor zo staršej sady stavebníc, ktorý sa dá pripojiť cez redukciu aj k NXT.
3. V zozname si vyberme ***Sound Sensor***.
4. V časti Port zvolíme port, na ktorom je pripojený. Zvukový senzor má byť pripojený na porte 2.

Port potrebuje program vedieť len pre účely kalibrácie. Po jej ukončení si bude kocka pamätať kalibráciu pre daný druh senzora bez ohľadu na to, na akom porte bude neskôr pripojený.

5. Stlačme tlačidlo ***Calibrate***. Do kocky sa pošle kalibračný program pre daný senzor a spustí sa. Na displeji kocky budeme vidieť hodnotu senzora.
6. Potrebujeme namerať bežný hluk v triede, teda hlukové pozadie – hukot počítačov, prípadne iné stále prítomné zvuky. Preto sa musíme vyvarovať ostatných zvukov a stlačiť oranžové tlačidlo.
7. Potom musíme vyrobiť najsilnejší zvuk, aký chceme použiť v danom programe a práve vtedy, keď ho senzor registruje, stlačiť znova oranžové tlačidlo.
8. V kalibračnom dialógu na počítači teraz stlačme tlačidlo ***Refresh*** a mali by sme vidieť namerané hodnoty minima a maxima.
9. Stlačme tlačidlo ***Close***.

Tým je zvukový senzor skalibrovaný. Odteraz bude pre ďalšie merania používať nový rozsah.

S takto skalibrovaným senzorom riešte teraz nasledujúce zadanie:

Zadanie

Black

Robot ide a zastane na čiernej farbe, povie "Black", keď potom počuje silný zvuk (napríklad lusknutie prstami), tak sa kúsok vráti.

Skúste skalibrovať aj svetelný senzor na príklade chodníka:

Zadanie
Chodník 2

Zmeňte v príklade Chodník 1 hraničnú hodnotu pre svetelný senzor na 50 a skalibrujte svetelný senzor tak, aby pre chodník dával hodnotu 100 a pre cestu hodnotu 0.

Skúste zmeniť farbu chodníka a cesty (ale cesta musí byť vždy tmavšia) napríklad nakreslením inou farbou alebo použitím inej farby podložky. Skalibrujte senzor na nové farby a vyskúšajte, či bude fungovať nezmenený program aj v novom prostredí.

Cyklus

Už pri programovaní priamo pomocou tlačidiel kocky, sme mali možnosť opakovať postupnosť príkazov donekonečna. Túto možnosť nám v ikonickom jazyku dáva blok ***Loop*** (Opakuj) v jeho základnom tvare – nekonečný cyklus.

Nekonečný cyklus predvedieme rozvinutím posledného zadania predošlej podkapitoly:

Zadanie
Vyhýbaj sa 1

Robot ide dopredu kým nie je blízko prekážky (zistí to ultrasonickým senzorom vzdialenosti), potom zatáča dozadu a doprava kým nenarazí tlakovým senzorom na prekážku. Túto činnosť opakuje donekonečna a tým preskúmava celú miestnosť a snaží sa pritom vyhnúť prekážkam.

Skúsme najprv telo programu bez cyklu:

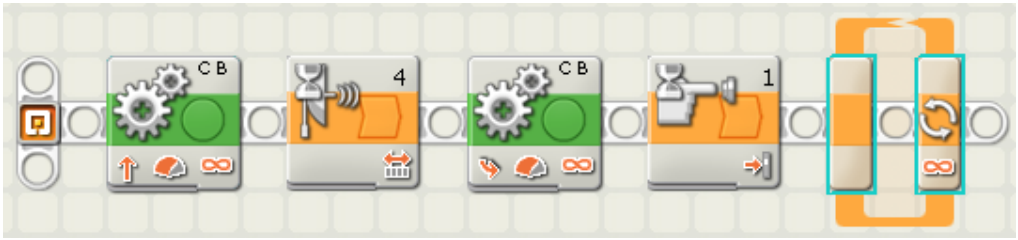
Pri programovaní cyklov potrebujeme často najprv vyskúšať len jeden prechod cyklom a až potom ho dať vykonávať viackrát v cykle. Náš príklad predvádza, ako to docielime v ikonickom jazyku robotov NXT.

Konfigurácia ultrasonického senzora definuje hranicu 30cm a čakáme na menšie hodnoty.

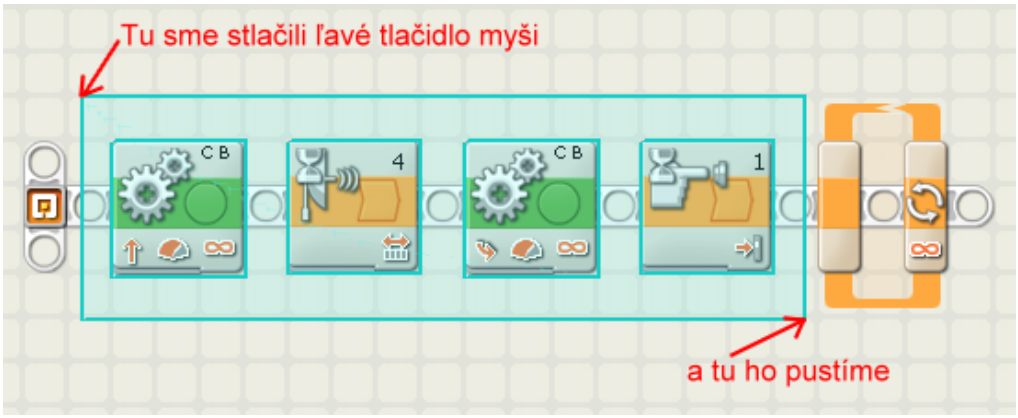
Program si vyskúšajte, či robí to, čo má.

Aby sme splnili naše zadanie, teraz musíme okolo štyroch blokov urobiť cyklus. V ikonickom jazyku to nie je také jednoduché ako v textovom programe, avšak nie je to ani veľmi zložité. Precvičíme si pritom prenášanie blokov v programe.

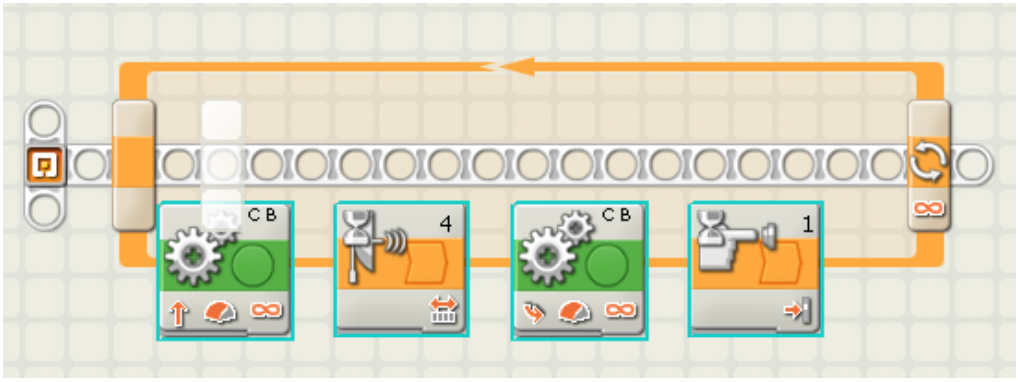
Najprv pridajme blok cyklu na koniec programu:



Teraz musíme presunúť štyri bloky dovnútra cyklu. Preto ich najprv označme tak, že myšou okolo nich nakreslíme obdĺžnik:



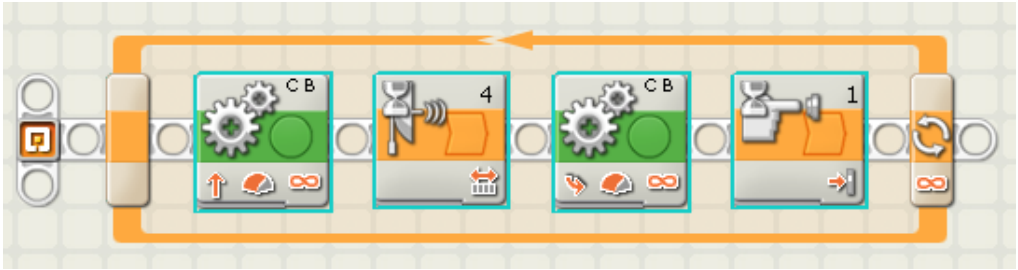
Po pustení tlačidla myši obdĺžnik zmizne, ale zostanú označené štyri bloky. Hociktorý z nich potom chytíme a ťaháme do vnútra cyklu (budú sa hýbať všetky štyri bloky):



Pri ťahaní viacerých blokov nezabúdajme na to, že musíme počkať, kým nám program jasne ukáže správne miesto vloženia blokov a aj správne zväčší či posunie ostatné bloky.

Bloky pustíme až keď sa nám blok cyklu správne zväčšil tak, ako to vidíme na obrázku hore.

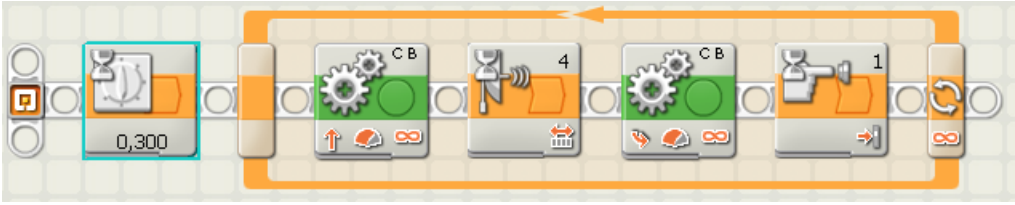
Po pustení blokov je program takýto:



Vidíme, že blok cyklu obklopuje naše štyri bloky, ktoré sa preto budú vykonávať v nekonečnom cykle.

Vyskúšajme si program. Možno sa aj vám stane, že robot sa hneď po tom, ako stlačíme jeho oranžové tlačidlo, začne pohybovať dozadu, hoci podľa programu mal ísť najprv dopredu až po prekážku. Je to preto, lebo sme mu pri stláčaní tlačidla zakryli rukou senzor vzdialenosti a program preto reagoval ako keby bol robot blízko steny.

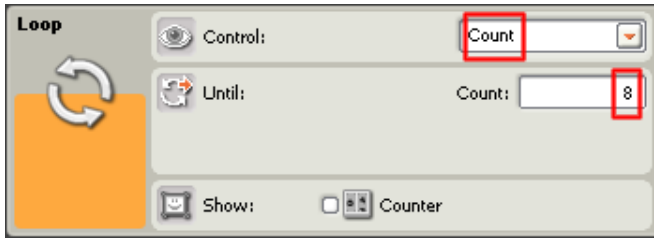
Tento problém môžeme odstrániť zaradením krátkeho čakania na začiatok programu:



Modifikácia zadania umožňuje riadiť robota aj zvukovými "povelmi":

Zadanie **Vyhýbaj sa 2**	Zmeňte program tak, aby robot išiel dozadu až kým nepočuje silný zvuk (tlesknutie, lusknutie prstami, výkrik, ...).

Ďalšie druhy cyklov umožňujú vykonať obsiahnuté príkazy istý počet krát alebo vykonávať ich po istú dobu. Skúste ich použiť v nasledujúcich príkladoch.

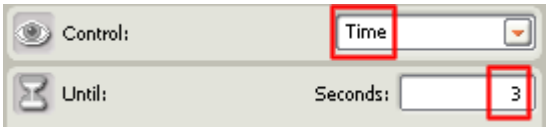
Zadanie **Tancuj 1**	Robot sa pomaly točí na mieste a vydáva zvuky a to tak, že sa vždy otočí o štvrťkruh, zastane a vydá tón. To opakuje niekoľko krát (napríklad 4 alebo 8) a potom zastane.
Pomôcka 1 **Počet opakovaní**	Cyklus s pevným počtom opakovaní dosiahneme tak, že najprv položíme do programu nekonečný cyklus, a potom zmeníme jeho typ v konfiguračnom paneli na ***Count*** a nastavíme počet opakovaní v časti ***Until***.  Všimnite si aj zmenenú ikonu bloku.
Pomôcka 2 **Tóny**	Blok zvuku vie hrať aj hudobné zvuky. Treba len v jeho konfiguračnom paneli prepnúť ***Action*** na ***Tone*** a potom v pravej časti nastaviť trvanie (v okienku) a výšku (kliknutím na klávesnicu klavíra alebo napísaním mena tónu do okienka).  **Dôležité:** Jeden blok zvuku zahrá len jeden tón. Na melódiu z viacerých tónov musíte použiť viac blokov za sebou.

Zadanie

Tancuj 2

Zmeňte program riešiaci predošlé zadanie tak, aby sa točenie a hranie neopakovalo istý počet krát, ale po istý čas, niekoľko sekúnd.

Tip:



Zadanie

Tancuj 2a

Skúste program zmeniť tak, aby hral počas točenia (aby nezastavoval).

Tip:

Zadanie

Modelka 2

Zmeňte program Modelka 1 z podkapitoly o pohyboch tak, aby pri točení aj občas niečo zahral (pri hraní nemusí zastaviť, ale dokážete potom program vyladiť tak, aby modelka skončila s otáčaním otočená presne smerom nazad po móle?)

Paralelné vykonávanie

Keď sme v predošlej kapitole chceli naraz robot hýbať aj hrať tóny, tak sme museli striedať tieto činnosti v jednom programe (Zadanie Tancuj 2a a Modelka 2).

Robot však dokáže vykonávať aj niekoľko činností naraz a to môže podstatne zjednodušiť logiku programu. Nemusíme sa starať o striedanie akcií, jednoducho naprogramujeme, aby sa isté veci diali zároveň.

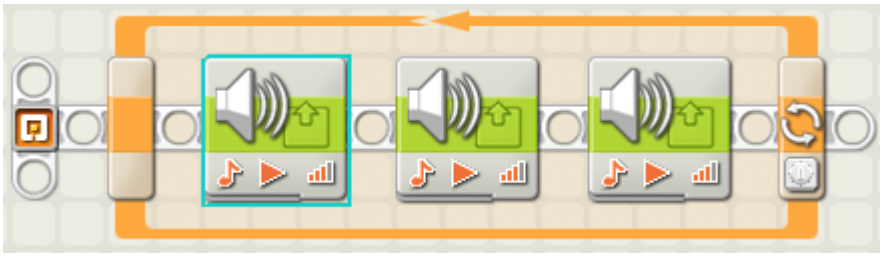
Ukážme si postup na jednoduchom zadaní:

Zadanie **Tancuj 2b**	Robot sa pomaly točí na mieste niekoľko sekúnd a zároveň hrá dokola jednoduchú znelku či melódiu z 3 až 5 tónov.

Naprogramujme najprv druhú časť zadania: znelku z troch tónov, ktorá sa opakuje istý čas, napríklad 10 sekúnd.

Tentokrát najprv umiestnime blok cyklu:

Do neho potom uložme niekoľko blokov zvuku a nastavme im hranie troch rôznych tónov:

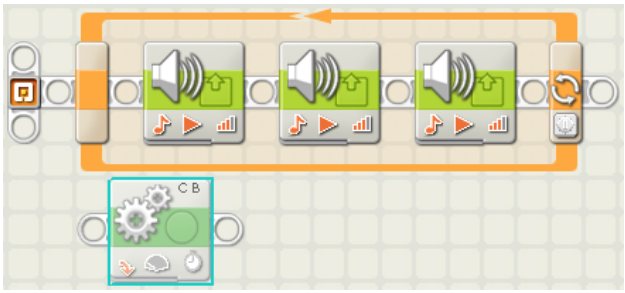


Všimnite si, že program vždy dohrá melódiu aj keby mal trvať viac než zadaných 10 sekúnd. Je to preto, lebo uplynutie času sa kontroluje vždy len na konci postupnosti príkazov v tele cyklu.

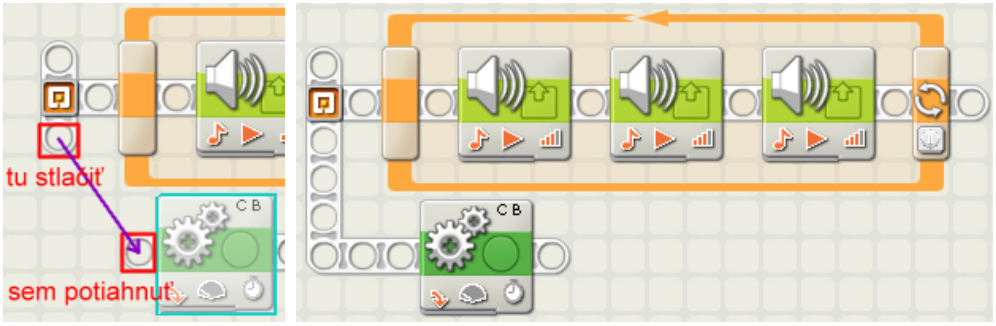
Dĺžku a výšku tónov nastavte podľa svojho hudobného cítenia, prípadne ich pridajte aj viac aby hrali krátku melódiu.

Program si vyskúšajte.

Točenie na mieste po dobu 10 sekúnd je ľahké naprogramovať, stačí na to jeden blok. Teraz ho ale nepoložme na šedú tyč za melódiu, ale pod ňu:

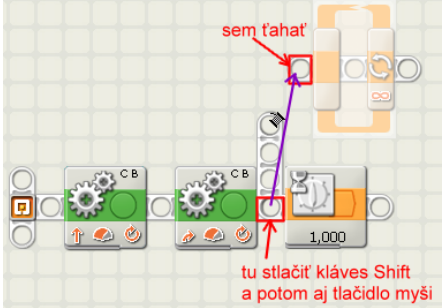


Vidíme, že nespojený príkaz je zobrazený ako šedý. Tým nám počítač dáva najavo, že ten príkaz zatiaľ nie je súčasťou programu. Teraz ho ale spojíme a to tak, že prídeme myšou k oku v dolnej vetve štartovacieho „téčka“ (viď obrázok vľavo), kurzor myši sa zmení na vretienko s drôtom, stlačíme ľavé tlačidlo myši a ťaháme ju k oku k tyči, na ktorej je blok pohybu. Tam tlačidlo myši pustíme a príkaz bude spojený so zvyškom programu (viď obrázok vpravo).



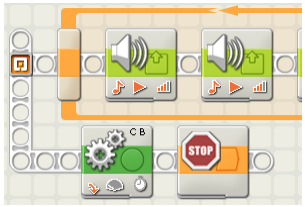
Obrázok nám jasne napovedá, že sa obe vetvy budú vykonávať naraz. Celý program skončí, keď skončia obe vetvy.

Program si vyskúšajme. Robot sa točí a zároveň hrá dookola znelku. Na konci však znelka znie trochu dlhšie, lebo sa dohrá ešte poslednýkrát kompletná a až potom program skončí. Ak nám toto vadí, môžeme skrátiť čas hrania v cykle hornej vetvy o 2 až 3 sekundy (o čas, ktorý trvá jedna znelka). Potom hudba dohrá trochu skôr než sa robot dotočí. Ak chceme zastaviť hranie presne vtedy keď aj točenie, môžeme použiť tip z pravého stĺpca na tejto strane.

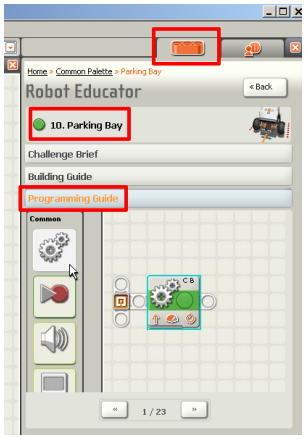
Zadanie **Zacúvaj 2**	Upravte program zo zadania Zacúvaj 1 tak, aby počas celého manévru cúvania (už od chvíle keď zastane po opísaní štvrťkruhu a až do chvíle keď prestane ukazovať zámok) vydával opakovane zvuk ***! Hydraulic 3***.
Pomôcka 1	Cúvajte s robotom istý čas (nie istý počet obrátok), tak budete presne vedieť koľko sekúnd treba v paralelnej vetve hrať zvuk.
Pomôcka 2 **Rozvetvenie v strede programu**	V tomto programe budete potrebovať vytvoriť paralelnú vetvu inde než hneď na začiatku programu. V takom prípade treba príkazy druhej vetvy napojiť na hlavnú vetvu tak, že myšou prídete na miesto, kde sa majú vetvy rozdeliť (medzi dvomi blokmi v hlavnej vetve), stlačíte kláves **Shift.** Až potom sa kurzor myši zmení na vretienko drôtu a môžete spojiť dané miesto so začiatkom druhej vetvy. 

Ak chceme aby melódia zastavila ihneď ako sa robot prestane točiť, môžeme použiť pokročilý blok ***Stop*** z kompletnej palety. Ten zastaví celý program, teda všetky jeho vetvy.
Prepnime paletu na kompletnú:

A vo štvrtej skupine blokov zhora nájdeme blok Stop, ktorý potiahneme na koniec dolnej vetvy nášho programu.



Tip: Zadanie úlohy Zacúvaj 2 sa nachádza aj v samotnom programe v jeho časti Robot Educator (Aktivita 10 v časti Common Palette). Môžete si ju teda naprogramovať podľa návodu.



Vetvenie programu (blok Switch)

Program môžeme vetviť podobne ako v iných programovacích jazykoch pomocou bloku pre podmienený príkaz (***Switch***). Jeho jednoduchý tvar má podmienku a dve vetvy. Jedna sa vykoná, keď je podmienka pravdivá, druhá, keď je nepravdivá.

Predvedieme si ho na zadaní:

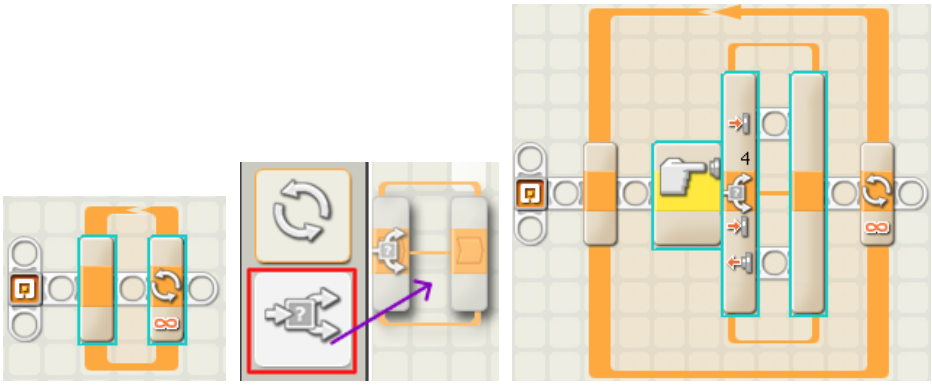
Zadanie

Diaľkový ovládač 1

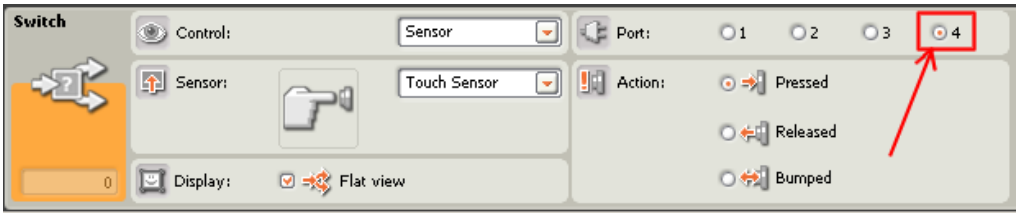
Odpojte ultrasonický senzor zo vstupného portu 4 a na dlhší káblik tam pripojte druhý tlakový senzor. Ten bude plniť úlohu jednoduchého ovládača. Naprogramujte robota tak, aby išiel stále dopredu, ale tak, že keď je stlačené tlačidlo, tak zatáča doprava, inak zatáča doľava.

Skúste riadiť robota tak, aby prešiel nejakú dráhu.

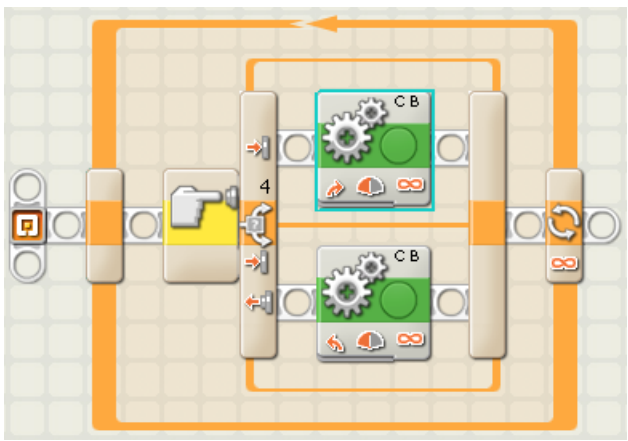
Telo nášho programu bude nekonečný cyklus, v ktorom použijeme podmienený príkaz na zistenie, či je stlačený tlakový senzor alebo nie. Začneme teda cyklom (obrázok vľavo) a do neho dáme podmienený príkaz – vezmeme ho z poslednej pozície na palete (obrázok v strede) a položíme ho do vnútra cyklu (obrázok vpravo):



V konfiguračnom paneli podmieneného príkazu nezabudnime prepnúť ***Port*** na 4 lebo tlakový senzor je pripojený na iný než zvyčajný port. Ostatné nastavenia netreba meniť.



Potom do každej vetvy prinesme blok zapnutia motorov (teda blok pohybu nastavený na ***Unlimited***). Obom znížme rýchlosť na 50, jeden z nich nech zatáča doprava a druhý doľava.



Keďže sa bloky pohybu v hornej a dolnej vetve líšia len v jednom nastavení, môžeme najprv doniesť a nastaviť horný blok, potom ho skopírovať (***Ctrl + ťahanie myšou***) aj do dolnej vetvy a tam mu zmeniť len smer zatáčania.

Konfigurácia horného bloku:

.

Konfigurácia dolného bloku sa líši len smerom zatáčania:

Podmienený príkaz funguje aj s inými typmi senzorov. Vyskúšame si to v nasledujúcich zadaniach.

Zadanie

Zima, teplo, horíš

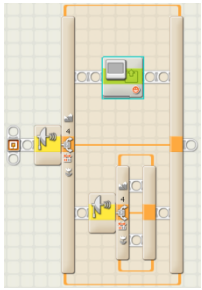
Naučme robota časť hry zima, teplo, horíš. Budeme pred ním pohybovať rukou a on má na svoj displej napísať ZIMA, ak je naša ruka ďaleko, TEPLO, keď je bližšie a HORIS, keď je ruka veľmi blízko.

Pomôcka 1

V konfiguračnom paneli podmieneného príkazu zvoľte ultrasonický senzor, a potom v jeho pravej časti nastavte hraničnú hodnotu a typ intervalu podobne ako v bloku čakania.

Pomôcka 2

Budete potrebovať dva vnorené podmienené príkazy.



Pomôcka 3

Text na displej napíšeme pomocou bloku zobrazenia tak, že v jeho konfiguračnom paneli zvolíme ***Action: Text***, potom napíšeme text. V pravej časti môžeme zadať aj presnú pozíciu textu na displeji.

Dôležité: Pre skúšanie ďalších programov nezabudnite znovu pripojiť ultrasonický senzor do portu 4.

Mohol by robot aj povedať „zima“? Áno, ale nie je to také jednoduché. Museli by sme nahrať slovo „zima“ ako zvukový súbor typu WAV, potom ho previesť do špeciálneho tvaru zvukov pre NXT (typ RSO) a potom ho umiestniť do správneho priečinka na disku, aby ho program poznal. Skúsený používateľ nájde návod napríklad na http://thenxtzoo.com/sound/sound_windows.pdf

Keď je program príliš rozvetvený, môže sa stať, že už nevojde na výšku obrazovky. Vtedy prepnite kurzor myši zo šípky na ruku (v palety nástrojov):

Pomocou ruky môžete posunúť plochu s programom tak, aby ste ho mohli celý prezrieť. Pre ďalšiu prácu si nezabudnite prepnúť kurzor späť na šípku.

Zadanie

Hudobný nástroj

Všimnite si, že na zozname senzorov, podľa ktorých sa môže vetviť podmienený príkaz, sú aj tri tlačidlá samotnej kocky pod názvom ***NXT buttons***. S použitím tejto voľby naprogramujte robota tak, aby stlačenie každého tlačidla vydávalo tón (každé tlačidlo iný).

Dáta a konektory

Kompletná paleta blokov:

Prvá položka kompletnej palety obsahuje všetky prvky základnej palety, preto sa už nemusíme prepínať nazad do základnej palety pre niektoré jej príkazy.

Výstupné konektory smerujú doprava a poskytujú údaje ďalším blokom v programe. Vstupné konektory smerujú doľava a nimi prichádzajú do bloku dáta z iných blokov, ktoré sú v programe pred nimi. Niekedy je na jednom mieste vstupný aj výstupný konektor.

Aj tento príklad môžete nájsť v časti Robot Educator, je to prvý príklad rozšírenej palety (má číslo 21).

Doposiaľ sme v programe používali len príkazy - pre motory, zvuky, displej, čakanie, cykly a vetvenie.

V ikonickom programovacom jazyku však možno použiť aj dáta a premenné.

Na to, aby sme ich mohli používať, musíme teraz opustiť základnú paletu, ktorú už poznáme, a začať používať kompletnú paletu. Preto si ju teraz prepnime a po zvyšok tohto učebného textu ju budeme používať.

Najzákladnejšiu prácu s dátami si ukážeme v nasledujúcom zadaní:

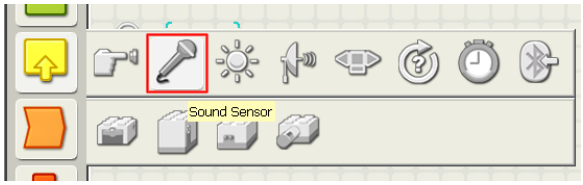
Zadanie

Bojazlivý robot

Bojazlivý robot uteká preč, keď začuje hluk. A beží tým rýchlejšie, čím je hluk silnejší.

Naprogramujme teda robota tak, aby išiel stále dopredu, ale jeho rýchlosť aby závisela od momentálnej sily zvuku, ktorý zachytáva jeho zvukový senzor.

Program bude opakovane vyhodnocovať hodnotu zo zvukového senzora a podľa nej nastavovať rýchlosť pohybu. Preto začneme nekonečným cyklom a do neho dáme nový blok – svetložltý blok zvukového senzora:

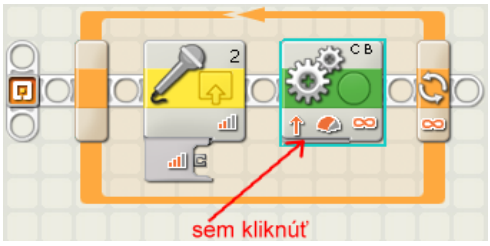


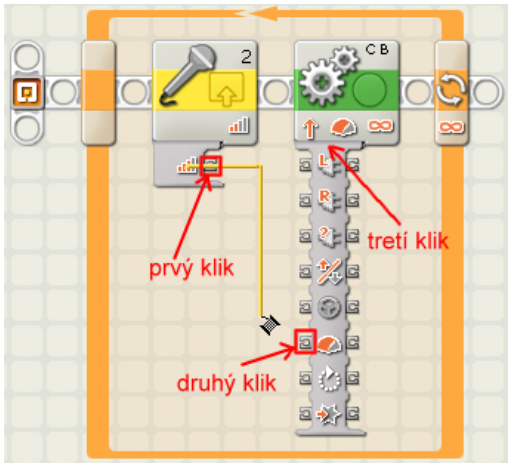
Vidíme, že má v spodnej časti čosi nové – dátový konektor (alebo len konektor). Účelom tohto bloku je totiž zmerať úroveň zvuku a výsledok dať k dispozícii cez svoj dátový konektor niektorému inému bloku, ktorý za ním nasleduje v programe.

V našom prípade je ďalším blokom blok pohybu. Ten už poznáme zo základnej palety a v kompletnej palete ho nájdeme v prvej položke.

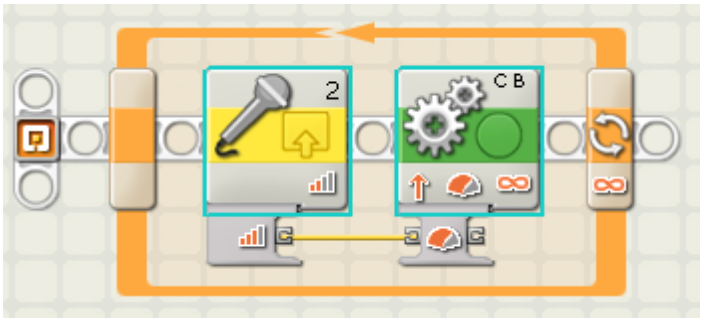
Keďže ho potrebujeme vo funkcii „zapni motory“, v jeho konfiguračnom paneli nezabudnime nastaviť Unlimited:

Aby mohol prijať dáta od bloku senzora, musí mať aj on svoj dátový konektor. Ale on ho v skutočnosti aj má (a to hneď niekoľko), len sme ho doteraz nepotrebovali, a preto bol schovaný. Dátové konektory bloku sa ukážu, keď klikneme do spodnej časti bloku, ako ukazuje šípka.





Už nám zostáva len spojiť výstupný konektor bloku zvukového senzora na ten vstupný konektor bloku pohybu, ktorý je označený ikonou rýchlosti. Prídeme myšou nad výstupný konektor bloku senzora, klikneme, ťaháme káblik k vstupnému konektoru bloku pohybu a tam zase klikneme (podrobnejší návod je v pravom/ľavom stĺpci). Keď sa nám konektory spoja žltým vodičom, tak klikneme ešte raz na miesto, kde sa otvárajú a zatvárajú konektory. Tým zatvoríme všetky nepoužité konektory aby bol program prehľadnejší, čím dostaneme tento výsledný program.



Vyskúšajte si program. Postavte robota na zem, spustite program. Robot bude možno stáť na mieste, možno sa bude pomaly hýbať dopredu, ale keď za ním začneme dupať, tak sa zľakne a rozbehne sa rýchlejšie.

To bolo jednoduché zadanie. Skúste niečo trocha zložitejšie:

Zadanie

Nasleduj pána

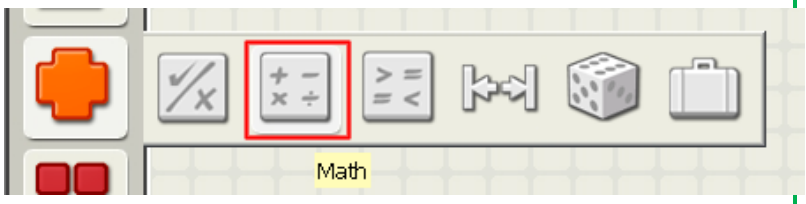
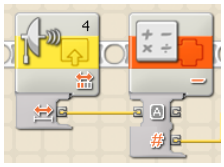
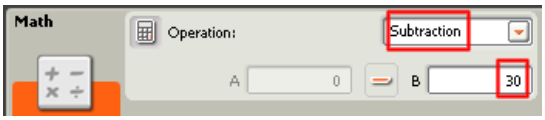
Robotický sluha má sledovať svojho pána vo vzdialenosti asi 30 cm. Keď mu pán ujde priďaleko, tak sa ho má snažiť čo najrýchlejšie dohoniť. Čím je pán bližšie, tým má ísť pomalšie. Keď je vzdialený 30cm alebo bližšie, tak má zostať stáť.

Pre úspešné použitie káblikov musíme postupovať takto (podľa [1]):

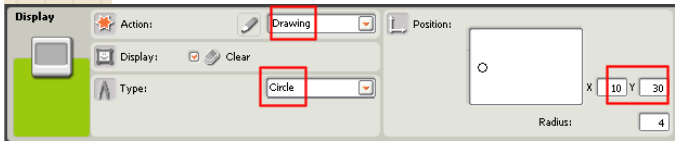
1. Prísť myšou nad výstupný konektor a počkať kým sa zmení kurzor na vretienko s drôtom.
2. Kliknúť ľavým tlačidlom myši.
3. Nenáhlivo presunúť myš (bez držania jej tlačidla!) nad vstupný konektor, kde chceme káblik ukončiť, skontrolovať presnú polohu.
4. Kliknúť ľavým tlačidlom myši.

Pri nedodržaní týchto pravidiel, hlavne ak sa budete pokúšať káblik ťahať so stlačeným ľavým tlačidlom myši alebo ak budete myš hýbať príliš rýchlo, káblik sa budú správať veľmi nepredvídateľne a dojem z celej činnosti môže byť veľmi frustrujúci.

Robot sa bude asi hýbať aj keď na neho nebudeme kričať ani dupať. Je to preto, lebo v miestnosti je vždy nejaká úroveň hluku. Ak chceme dosiahnuť, aby robot stál na mieste, keď sme ticho, tak skalibrujme zvukový senzor tak, aby pre bežnú úroveň hluku v miestnosti bola jeho hodnota 0. Potom bude v danej situácii aj rýchlosť robota nulová, teda bude stáť na mieste.

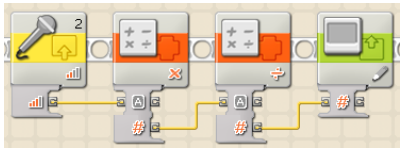
Pomôcka	Výstupom z ultrasonického senzora je vzdialenosť v centimetroch. Z tohto čísla môžeme pomocou aritmetickej operácie získať číslo udávajúce rýchlosť tak, aby pri vzdialenosti 30cm bola rýchlosť 0, pre väčšie vzdialenosti než 30cm rýchlosť rástla a pre menšie vzdialenosti bola záporná (blok pohybu sa pre záporný vstup pre rýchlosť chová akoby to bola nula, teda motory zostanú stáť). Operácie potom naprogramujete do robota pomocou aritmetického bloku, ktorý nájdete v položke Dáta a premenné (***Data***).   

Pomocou konektorov a dátových spojení medzi blokmi možno naprogramovať aj zložité úlohy, ako napríklad takúto:

Zadanie **Grafické znázornenie hluku**	Zobrazte na displeji graficky úroveň hluku alebo svetla v miestnosti pomocou väčšej alebo menšej kružnice nakreslenej v strede displeja kocky. Program môže vyžadovať kalibráciu zvukového senzora tak, aby v prvom bola pre úroveň šumu pozadia (bežiacich počítačov) jeho hodnota 0.
Pomôcka 1	Najprv si naštudujte a vyskúšajte možnosti kreslenia v bloku zobrazenia. Ten vie nakresliť kružnicu (ale aj bod a úsečku ako aj obrázok z viacerých útvarov, ale to teraz nepotrebujeme) nejakej veľkosti na nejakom mieste a jej polomer môže byť riadený dátami zo vstupného konektora (vtedy ignoruje polomer zadaný v konfiguračnom paneli).  

Pomôcka 2

Nezabudnite prepočítať silu zvuku či svetla z intervalu 0 až 100 do intervalu možných polomerov tak, aby kružnica nevyšla z obrazovky. Myslite pri tom na to, že kocka NXT pozná len celé čísla, preto aj operáciu delenia chápe ako celočíselnú (použite poznatky o počítaní s celými číslami z predmetu Programovanie).



Premenné (voliteľné, pre pokročilých)

Premenné slúžia, podobne ako v iných programovacích jazykoch, na uloženie konkrétnej hodnoty údajov aby sme ho mohli použiť neskôr.

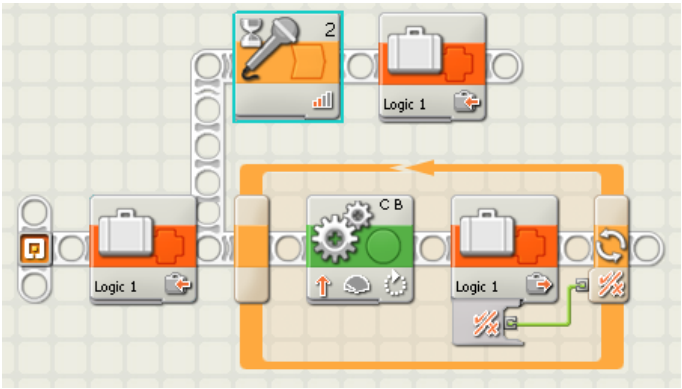
V robotických programoch sa vyskytuje situácia, keď si hodnotu senzora potrebujeme práve takto "odložiť". Teda použiť ju v programe až neskôr, než bola zistená alebo keď už daný senzor má inú hodnotu. Pomocou premenných môžeme aj preniesť hodnotu nameranú v jednej vetve programu do inej paralelne bežiacej vetvy. Ukážeme si to na programe:

Zadanie
Tancuj 4

Robot sa pomaly točí na mieste až kým ho nezastavíme tlieskaním (silným zvukom). Nezastane však hneď, keď ho počuje, ale najprv dokončí celú obrátku.

Myšlienka programu je takáto:

Na zapamätanie, či bol už potlesk použijeme logickú (v Pascale by sme povedali, že booleovskú) premennú ***Logic 1*** (mohli by sme ju nazvať aj krajšie, ale takáto premenná už existuje, tak ju použijeme). Na začiatku programu teda priradíme do programu hodnotu Nie (***False***). Potom sa program rozdelí na dve paralelné vetvy. Jedna bude len čakať na potlesk a keď ho počuje, tak zapíše do premennej Logic 1 hodnotu Áno (***True***) a skončí. V druhej paralelnej vetve sa bude robot točiť v cykle až kým hodnota premennej ***Logic 1*** nenadobudne hodnotu ***True***.



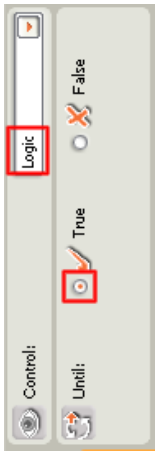
Teraz už dokážete naprogramovať najzložitejšiu verziu modelky, ktorá sa točí kým jej nezatlieskajú, ale pritom vie odísť správnym smerom späť po móle:

Zadanie
Modelka 3

Zmeňte program Modelka 1 z podkapitoly o pohyboch tak, aby sa pri točení na konci móla modelka točila stále až kým jej diváci nezatlieskajú (silný zvuk). Po skončení točenia však musí odísť správnym smerom späť po móle.

Narábanie s premennými sa vykonáva v blokoch premennej. Jeden takýto blok predstavuje jednu operáciu s premennou. Operácia je zápis alebo čítanie.

Príkaz cyklu môže byť riadený aj dátami, musíme to však zvoliť v jeho konfiguračnom paneli, ako aj to, pri akej logickej hodnote má skončiť:



Čím je programovanie robota odlišné

Teraz už máte pomerne bohaté skúsenosti s programovaním robota. Určite ste si všimli, že programovanie v rámci robotiky sa občas dosť líši od programovania všeobecnosti (tak ako sme sa to učili v moduloch predmetu Programovanie). Treba myslieť na rôzne veci, ktoré pri programe v počítači nehrajú žiadnu rolu. Aj takáto skúsenosť rozvíja u žiakov hodnotné schopnosti.

Niektoré rozdiely medzi programovaním počítačov a programovaním robotov:

- **vplyvy prostredia** - v triede je hluk, rôzna intenzita svetla, podlaha nemusí byť vždy rovná - aj toto sú faktory, ktoré môžu prispieť k nepresnostiam v správaní robota, je treba s nimi počítať;
- **mechanické obmedzenia a nepresnosti** - robot je zložený zo skutočných súčiastok, ich materiál sa môže opotrebovať, motory sa používaním kazia, každý môže ísť pri rovnakom nastavení inak rýchlo;
- **programovanie v reálnom čase -** pri reakcii na senzory, napríklad keď robot ide naraziť, má program reálne časové obmedzenie, v ktorom musí danú akciu vykonať inak by stratila zmysel;
- **paralelné akcie robota a programu** - hardvér robota vykonáva nejakú činnosť, ale program zároveň čaká.

Zhrnutie

Naučili sme sa základy programovania v ikonickom programovacom jazyku LEGO Mindstroms EDU.

Poznáme:

- zásady programovania kocky NXT v ikonickom jazyku,
- pracovné postupy pri tvorbe a ladení programov,
- špecifiká programovania robotov.

Vieme použiť

- výkonné bloky pre pohyb, zvuky, zobrazenie na displeji,
- riadiace bloky pre čakanie, cykly, vetvenie programu a zastavenie,
- bloky senzorov ako zdroje dát,
- dátové konektory a káblíky na prenos dát medzi blokmi,
- premenné na uchovanie údajov.

Komunikácia medzi kockami

V tejto časti budú účastníci vzdelávania v skupinách riešiť úlohy zamerané na posielanie správ medzi dvoma NXT kockami.

Každý tím dostane kartičku s číslami úloh, ktoré máte riešiť. Kartička ďalej obsahuje meno iného tímu, s ktorým budete komunikovať prostredníctvom vášho robota. Ak sa s týmto tímom potrebujete počas riešenia i dohodnúť/poradiť, smelo do toho!

Kartičky sú len jednou z možností, ako určiť, ktorý tím bude komunikovať s ktorým, lektor môže zvážiť iný spôsob rozdelenia tímov.

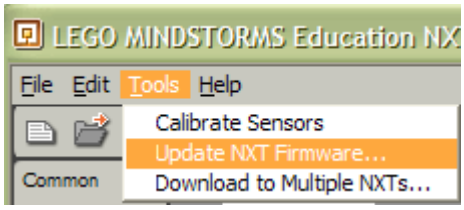
Tímy sa budú striedať v tom, kto posiela a kto prijíma správy, preto nerieši každý tím všetky úlohy.

Tímy je vhodné nejako pomenovať, aby sa s nimi lepšie pracovalo.

Úloha 1

Pripojte svoju kocku k počítaču a stiahnite do nej firmware. Tým sa zaručí, že budú všetky kocky mať rovnaký obsah a zmažú sa tým aj ich rôzne nastavenia.

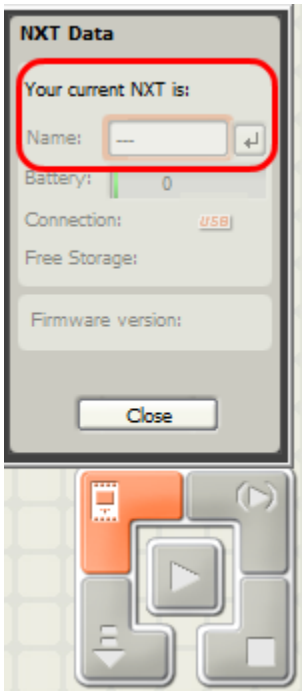
- Otvorte si programovacie prostredie Lego Mindstorms Education NXT a zvoľte príkaz ***Tools> Update NXT firmware***.



Úloha 2

Premenujte svoju kocku podľa čísla alebo písmena na jej spodnej strane. Kocku nazvite slovom: jednicka, dvojka, acko, becko...

Kocka sa dá pomenovať cez menu ***NXT window***.



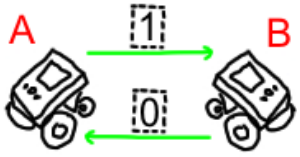
Úloha 3

V ďalších úlohách bude váš robot komunikovať s robotom tímu zadaného na kartičke.

- Presvedčte sa, či má váš robot zapnuté Bluetooth spojenie (ak nie, nastavte ho).
- Presvedčte sa, či má váš robot nastavenú viditeľnosť pre iné roboty (nastavte ho na viditeľného).
- Dokážete nájsť iné kocky vo vašom okolí. Je medzi nimi aj robot spolupracujúceho tímu?

Ako nastaviť tak aby mohli komunikovať

1. Na displeji NXT kocky A vyberte v menu položku ***Bluetooth***.
2. Zvoľte položku ***Search*** a nechajte kocku prehľadať okolie a nájsť ostatné Bluetooth zariadenia.
3. Zo zoznamu si **vyberte zariadenie**, s ktorým sa chcete spojiť - teda NXT kocku B.
4. Na displeji kocky A **priraďte kocke B číslo spojenia**, napríklad 1. Toto číslo si zapamätajte, pretože ho budete používať pri programovaní.
5. Ak si vyberiete zariadenie po prvý raz, tak sa vás vaša NXT kocka A opýta na ***Passkey*** (kľúčové slovo) vybraného zariadenia. Stačí stlačiť oranžové tlačidlo ak používate klasické slovo 1234, alebo zadať vlastný kód. Oslovená kocka B musí poznať Váš ***Passkey***, aby mohla potvrdiť spojenie. To znamená, že dve NXT musia zadať to isté kľúčové slovo, aby mohli byť spojené.
6. Na kocku B príde žiadosť o potvrdenie kľúčového slova, naťukajte teda dohodnuté slovo a potvrďte.
7. Teraz môžu spolu NXT kocky A a B komunikovať a posielať si navzájom správy.



Posielame a prijímame správu

Ak sú kocky medzi sebou spojené môžu si posielať správy. Na odosielanie a prijímanie máme k dispozícii niekoľko príkazov.

Príklad programu pre posielanie správy a nastavenie príkazu "Send Message"

Pošli správu (*Send message*)

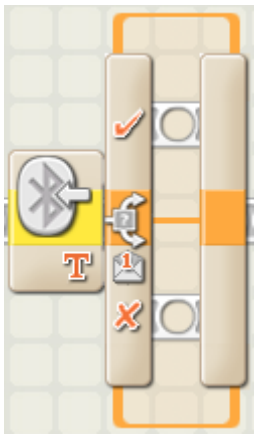
- odošle správu,
- správa môže byť text, číslo alebo logická hodnota,
- ak necháte políčko so správou prázdne, posiela sa nula 0.

Prijmi správu (*Receive message*)

- prijme správu,
- správa môže byť text, číslo alebo logická hodnota,
- prijatú správu môže poslať ďalej inému príkazu cez svoj dátový konektor.

Čakaj (*Wait - Sensor, Receive message*)

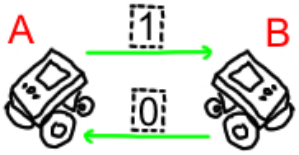
- čaká na správu.



Prepínanie (*Switch - Sensor, Receive message*)

- ak prijme správu, vyhodnotí jej obsah a vykoná jednu z vetiev podľa toho, ktorú podmienku obsah správy spĺňa,
- ak správa prejde týmto blokom a nespĺňa žiadnu z podmienok vo vetvách, vykoná sa tá vetva, ktorá je označená ako základná (***default***),
- po prejdení týmto blokom sa obsah správy stratí, preto sa nedá znovu testovať vo vnorenom prepínacom bloku bez použitia premennej.

Úloha 4a	Napíšte program, ktorý každých 5 sekúnd vyšle správu s textom "ahoj". Hotový program uložte pod názvom **posli.rbt**.
Úloha 4b	Napíšte program, ktorý čaká na textovú správu a keď nejakú dostane, vypíše ju na displej. Hotový program uložte pod názvom **prijmi.rbt**.



Kocka, ktorá bude posielať správy, nech je kocka A z nášho návodu na spojenie dvoch kociek. V príkaze Pošli správu (***Send message***) musíme teda nastaviť položku ***Connection*** na hodnotu 1. Je to číslo spojenia, ktoré sme nastavili pre kocku B.

Ak by správu odosielala kocka B, číslo spojenia ***Connection*** by muselo mať hodnotu 0 - 0 je číslo spojenia na kocku A.

Autíčko na ovládanie

Spolu s ďalším tímom vytvoríte robotické autíčko na ovládanie. Jedna kocka bude predstavovať ovládač a druhá autíčko. Ovládač bude autíčku posielať pokyny dopredu, vpravo, vľavo... a autíčko sa bude podľa nich pohybovať.

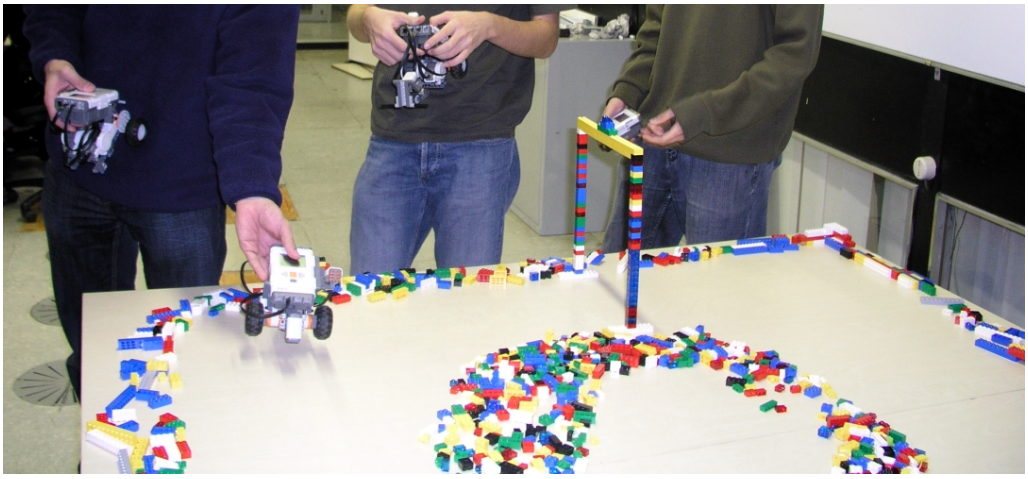
Úloha 5a	Vytvorte program na posielanie pokynov na ovládanie autíčka. Vaša kocka bude fungovať ako diaľkový ovládač: keď na nej stlačíte pravé tlačidlo, pošle správu o pohybe doprava. Analogicky bude fungovať stlačenie ľavého tlačidla. Zamyslite sa nad tým, aké ďalšie signály potrebujete, ak má kocka prejsť rôzne typy dráh (zákruty, rovná cesta). Hotový program uložte pod názvom **autoPosli.rbt**.
Úloha 5b	Vytvorte program, pomocou ktorého bude robot reagovať na prijaté správy od iného robota. Dokáže teda dešifrovať pokyny na odbočenie doprava, doľava a ďalšie signály od robota – diaľkového ovládača. Hotový program uložte pod názvom **autoPrijem.rbt**.
Úloha 5c	Spoločnou úlohou vašich robotov je prejsť pripravenú dráhu. Stopnite si čas potrebný na jej prejdenie. Ako by ste docielili, aby váš robot išiel rýchlejšie/presnejšie?

Táto úloha sa dá vyriešiť viacerými spôsobmi. Dobré riešenie je aj s použitím vnorených rozhodovacích blokov (***Switch***), aj keď sa pri ňom niektoré vyslané a prijaté správy "stratia". V riešení môžete, ale aj nemusíte použiť premenné.

Jedno z možných riešení môžete nájsť medzi súbormi v e-learningovom prostredí Moodle.

Priebeh súťaže

- Dráha pre súťaž autíčok sa dá pripraviť z plastových škatúľ od stavebníc, alebo z nepoužitých kociek.
- Čas je možné odmerať stopkami, mobilným telefónom, alebo na počítači.
- Osvedčilo sa nám zapisovať tri súťažné kolá a pripočítavať trestné sekundy za porušenie dráhy.
- Vyhrať môže tím, ktorý bude mať vôbec najlepší čas v celej súťaži, alebo tím, ktorý bude mať najlepší súčet všetkých troch časov.



Súťaž robotických autíčok

Switch môže mať viac než dve vetvy:

- označte blok ***Switch***,
- nastavte jeho položku ***Control*** na hodnotu ***Value***,
- pozrite, či je zvolené nastavenie ***Flat view***, ak áno, zrušte ho,
- kliknite na tlačidlo „+“ a objaví sa ďalšia možná vetva.

(Pozri obrázok dolu.) Takto rozvetvený *Switch* sa už nedá zobraziť vo formáte ***Flat view***. V takomto rozhodovacom bloku by ste mali testovať iba číselné premenné.

Potápajúca sa lodička

Dva roboty, s ktorými budete pracovať, predstavujú dve lodičky na mori. Jedna lodička sa dostala do ťažkostí a postupne morzeovkou vysiela signál SOS. Druhá loďka jej po prijatí signálu ide na pomoc.

Úloha 6a	Vytvorte program na zadávanie signálov morzeovky. Ľavým tlačidlom pošlete „krátky tón“, pravým tlačidlom „dlhý tón“. Pomocou vášho programu vyťukajte správu SOS a sledujte, akú správu príjme robot vašich spolupracovníkov. Hotový program uložte pod názvom **sosVysielanie.rbt**.
Úloha 6b	Vytvorte program na prijatie signálov od inej kocky. Kocka vám posiela dlhý alebo krátky tón. Vytvorte program, ktorý zahrá dlhý tón po prijatí zodpovedajúceho signálu. Analogicky krátky tón. Hotový program uložte pod názvom **sosPrijem.rbt**.

Pri tomto zadaní je nutné zabezpečiť, aby sa žiadna správa nestratila. V riešení môžete, ale aj nemusíte použiť premenné.

Jedno z možných riešení môžete nájsť medzi súbormi v prostredí Moodle.

Námet na ďalšiu úlohu	S vašim spolupracujúcim tímom a ďalšími kockami vytvorte tanečný zbor. Jedna kocka bude predstavovať choreografa. Bude posielať signály ostatným kockám, čo majú spraviť. Dohodnite sa na choreografii, pokynoch a skúste pohyb robotov zosynchronizovať s hudobnou skladbou podľa vášho výberu.

Zhrnutie

Naučili sme sa spojiť dve NXT kocky a posielať a prijímať medzi nimi správy.

Čo sme sa naučili v tomto module

Zhrnutie

Dozvedeli sme sa, čo je to robotika a aké je jej miesto vo vzdelávaní. Vysvetlili sme si prečo budeme používať vybranú stavebnicu LEGO a dozvedeli sme sa aj o iných robotických platformách.

Poznáme aktívne prvky stavebnice Lego Mindstorms a vieme, ako ich pripojiť k programovateľnej kocke. Pomocou senzorov dokážeme namerať rôzne vlastnosti prostredia. Vytvárame jednoduché programy priamo v programovateľnej kocke.

Naučili sme sa základy programovania v ikonickom programovacom jazyku LEGO Mindstroms EDU.

Poznáme: zásady programovania kocky NXT v ikonickom jazyku, pracovné postupy pri tvorbe a ladení programov, špecifiká programovania robotov.

Vieme použiť: výkonné bloky pre pohyb, zvuky, zobrazenie na displeji, riadiace bloky pre čakanie, cykly, vetvenie programu a zastavenie, bloky senzorov ako zdroje dát, dátové konektory a káblíky na prenos dát medzi blokmi, premenné na uchovanie údajov.

Naučili sme sa spojiť dve NXT kocky a posielať a prijímať medzi nimi správy.

Predpokladané výstupné vedomosti

Účastník vzdelávania

- rozpoznáva jednotlivé typy senzorov,
- dokáže zobraziť hodnoty namerané senzormi na displeji kocky,
- vytvára jednoduché programy programovaním priamo v NXT kocke robotického modelu.

Ďalšie výstupné kompetencie účastníka vzdelávania sa týkajú rozvinutia úrovne ovládania robotického zariadenia. Účastník vzdelávania

- vytvára jednoduché programy na pohyb a otáčanie robota,
- v programovacom prostredí definuje správanie robota ako reakciu na podnety prostredia.

Pri vytváraní vlastných programov dokáže účastník vzdelávania

- využiť paralelizmus pre ovládanie rôznych častí robotického modelu,
- nájsť a odstrániť prípadné nedostatky svojich programov vyplývajúce z nesprávneho zapojenia senzorov či motorov,
- otestovať správanie svojho robota, analyzovať možné nedostatky a iteratívne vylepšiť jeho program tak, aby spĺňal kritériá kladené na robotické zariadenie.

Preverenie výstupných vedomostí

Preverenie vedomostí budú vykonávať lektori priebežne. Na rozdiel od čisto programátorských modulov nemôžeme zadať domácu úlohu, lebo zďaleka nie všetci účastníci majú k dispozícii sadu LEGO NXT.

Literatúra a použité zdroje

[1] School of Electrical and Computer Engineering at the Georgia Institute of Technology (2007) *LEGO NXT Brick Programming Guide* http://www.ece.gatech.edu/academic/courses/ece1882/docs/NXTProgrammingGuide.pdf

[2] Siraj-Blatchford, J., Whitebread, D. (2007) Supporting Information and Communications Technology in the Early Years. Open University Press, McGraw-Hill Education, 4. vydanie. ISBN 0-335-20942-4

[3] Rusk, N., Resnick, M., Berg, R., a Pezalla-Granlund, M. (2008) New Pathways into Robotics: Strategies for Broadening Participation. Journal of Science Education and Technology.

Na ďalšie štúdium účastníkom odporúčame:

- Cehelský, P. (2004) Stavebnice LEGO Dacta – prostriedok na grafické znázornenie fyzikálnych veličín. Zborník príspevkov zo 4. celoštátnej konferencie Infovek. Bratislava: ÚIPŠ, str. 214 – 218. ISBN 80-7098-381-7
- Cehelský, P. (2005) LEGO Dacta CtrLab a RoboLab v základnej škole. Zborník príspevkov z 5. celoštátnej konferencie Infovek. Bratislava: ÚIPŠ, str. 73 – 78. ISBN 80-7098-422-8
- Kabátová, M., Pekárová, J. (2008) Hra = učenie sa. LEGO a robotika vo vyučovaní budúcich učiteľov. Didinfo 2008. Banská Bystrica: FPV UMB. ISBN 978-80-8083-556-9
- Petrovič, P., Balogh, R., Pekárová, J. (2008) Robotické vzdelávacie iniciatívy. In: Informatika v škole a v praxi. Zborník 4. ročníka medzinárodnej konferencie. Ružomberok: Pedagogická fakulta Katolíckej univerzity v Ružomberku, str. 239 – 248. ISBN 978-80-8084-362-5
- Plichtová, J. (2005) RoboLab do škôl áno či nie? Zborník príspevkov z 5. celoštátnej konferencie Infovek. Bratislava: ÚIPŠ, str. 78 – 81. ISBN 80-7098-422-8
- Šestáková, E. (2004) Inovácia – projekt skleník. Zborník príspevkov zo 4. celoštátnej konferencie Infovek. Bratislava: ÚIPŠ, str. 199 – 202. ISBN 80-7098-381-7
- BUMGARDNER, J. The Origins of Mindstorms. Wired, 2007. http://www.wired.com/geekdad/2007/03/the_origins_of_/ *Carnegie Mellon Robotics Academy: Introduction to Robotics.* 2006. http://www1.lego.com/education/search/default.asp?l2id=0_1&page=7_1
- INDELL, D. *LEGO Mindstorms: The Structure of an Engineering (R)evolution*. MIT, 2000. http://web.mit.edu/6.933/www/Fall2000/LegoMindstorms.pdf
- RESNICK, M., Martin F., Sargent R. et al. Programmable Bricks: Toys to think with. In *IBM Systems Journal*, 1996, vol. 35, Numbers 3 & 4. MIT Media Lab. Dostupné tiež na www: http://www.research.ibm.com/journal/sj/353/sectionc/martin.html
- ŽELÚDEK, J., POLČIN, D. Stavebnica Lego Robolab v prostredí ZŠ. In *Informatika v škole a praxi*. Zborník 4. ročníka medzinárodnej konferencie, s. 289 – 295. Ružomberok: Katolícka univerzita, 2008. ISBN 978-80-8084-362-5.
- Slovenský preklad používateľskej príručky LEGO Mindstorms Education http://eduxe.cz/download/files/9797_lme_manual_sk.pdf

Tento študijný materiál vznikol ako súčasť národného projektu Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika v rámci Aktivity „Vzdelávanie nekvalifikovaných učiteľov informatiky na 2. stupni ZŠ a na SŠ“.

Autori © RNDr. Peter Tomcsányi
PaedDr. Martina Kabátová
PaedDr. Janka Pekárová
Mgr. Ľubomír Janovický

Názov Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

Podnázov Robotické stavebnice vo vzdelávaní

Študijný materiál prešiel recenzným pokračovaním.

Recenzenti RNDr. Slávka Blichová
PaedDr. Miroslav Vojtek

Počet strán 40

Náklad 300 ks

Prvé vydanie, Bratislava 2010

Vydal Štátny pedagogický ústav, Pluhová 8, 830 00 Bratislava, v súčinnosti s Univerzitou Pavla Jozefa Šafárika v Košiciach, Univerzitou Komenského v Bratislave, Univerzitou Konštantína Filozofa v Nitre, Univerzitou Mateja Bela v Banskej Bystrici a Žilinskou univerzitou v Žiline

Vytlačil BRATIA SABOVCI, s r.o., Zvolen

ISBN 978-80-8118-044-6