ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA PEDAGOGICKÁ
KATEDRA VÝPOČETNÍ A DIDAKTICKÉ TECHNIKY

LEGO MINDSTORMS NXT - VYUŽITÍ PROGRAMOVACÍCH PROSTŘEDÍ NXT-G A ROBOTC

DIPLOMOVÁ PRÁCE

Bc. Jan Baťko
Učitelství pro 2. stupeň ZŠ, obor Inf-Te

Vedoucí práce: Mgr. Tomáš Jakeš

Plzeň, 2014

Prohlašuji, že jsem kvalifikační práci vypracoval samostatně s použitím uvedené literatury a zdrojů informací.

V Plzni, 26. března 2014

.....

vlastnoruční podpis

Poděkování

Rád bych na tomto místě poděkoval vedoucímu práce Mgr. Tomáši Jakešovi za jeho věcné připomínky a rady, které mi pomohly při tvorbě této práce.

OBSAH

ÚVOD

Robotická stavebnice LEGO Mindstorms NXT nabývá v posledních letech čím dál větší oblibě nejen u jednotlivých uživatelů, ale hlavně ve sféře vzdělávání. Mnoho škol ji začleňuje z různých důvodů do výuky. Jedním z nich je i možnost alternativního způsobu výuky programování pomocí různých programovacích prostředí. Bohužel dostupnost česky psaných výukových materiálů pro programování robotické stavebnice je velmi malá.

Cílem práce je představit možnosti využití robotické stavebnice LEGO Mindstorms NXT na všech stupních vzdělávání a následně porovnat možnosti robotických programovacích prostředí NXT-G a RobotC. Na základě předložené osnovy dále jako součást práce vznikne výukový materiál, který představí základní programové konstrukty obou prostředí a vysvětlí možnosti využití modulů robotické stavebnice. Tím podpoříme výuku robotického programování u nás.

V první kapitole představíme robotickou stavebnici a historii jejího vývoje. Dále navážeme popisem možností jejího využití na různých stupních škol. Zaměříme se na motorické schopnosti a dovednosti žáků, jejich myšlení a další dispozice, na jejichž základě popíšeme, jakým způsobem a v jakých předmětech je vhodné robotickou stavebnici ve vyučování na daném stupni využívat.

V druhé kapitole popíšeme programovací prostředí NXT-G a RobotC. Zaměříme se na proces vytváření zdrojového kódu programu, popíšeme jejich specifika a míru uživatelské podpory. Následně porovnáme možnosti obou prostředí na základě zvolených kritérií a výsledky okomentujeme.

Nakonec představíme výukový materiál, který vznikne v rámci této práce. Dozvíte se, které cíle se při jeho tvorbě budeme snažit naplnit, jakým způsobem a v jakém prostředí kurz vznikne. Popíšeme také jeho výslednou podobu. V závěru doplníme informace o tom, jak by mohl být využit ve výuce.

1 LEGO MINDSTORMS NXT

LEGO Mindstorms NXT je programovatelná robotická stavebnice, která si prošla během několika desítek let rozsáhlým vývojem. Značných změn se během tohoto vývoje dočkal jak celkový koncept stavebnice, tak i řídící jednotka, která je jejím základním stavebním prvkem. Společnost LEGO vyrábí několik typů sad robotických stavebnic. Rozdíl mezi nimi je v počtech a typech obsažených dílů. Na trhu nalezneme i jiné výrobce příslušenství robotické stavebnice než je firma LEGO. Díky tomu se nám otvírají širší možnosti praktického využití robotické stavebnice.

Využívání různých typů robotických stavebnic nabralo rychlý spád. V dnešní době se používají ve výuce prakticky na všech stupních škol. Setkáme se s nimi na školách základních, středních, gymnáziích ale také na vysokých školách.

V této kapitole se seznámíte s historickým vývojem robotické stavebnice a také se současnými možnostmi jejího využívání. Dozvíte se, jaké součásti základní sada stavebnice obsahuje a jaká rozšiřující zařízení pro její plnohodnotné využívání můžeme používat. Následně popíšeme možnosti jejího využití na jednotlivých stupních škol.

1.1 HISTORIE A VÝVOJ

Stavebnice LEGO Mindstorms NXT pochází z dílny specializovaného oddělení známé společnosti LEGO, která vyrábí oblíbené dětské stavebnice stejnojmenného názvu. Prvopočátky vývoje stavebnice sahají až do 60. let minulého století. V té době se profesor Seymour Papert z Media Lab (oddělení na Massachusetts Institut of Technology) začal zabývat vývojem nových technologií pro děti. V 80. letech na něj navázal profesor Mitchel Resnick, který se zabýval propojením hraček, počítače a učení ve výuce. [1]

Od roku 1985 byl vývoj v Media Lab sponzorován společností LEGO. V roce 1988 vznikl na základě této spolupráce první produkt s názvem LEGO tc Logo. Tato stavebnice studentům umožňovala programovat chování vlastní vytvořené konstrukce z dílů stavebnice LEGO. Nevýhodou ovšem bylo, že vytvořená konstrukce musela být neustále připojena k počítači kabelem. Z tohoto důvodu se ještě v témže roce začalo s vývojem první programovatelné kostky. [1] Mezi léty 1989 a 1996 bylo vyvinutu několik verzí

programovatelných kostek. Poslední z nich se jmenovala MIT Programmable Brick. Nebyla ovšem uvolněna do komerčního prodeje. [2]

Na těchto základech tedy začala ve spolupráci společnosti LEGO s Media Lab vznikat programovatelná stavebnice, která je založena na technologii MIT Brick. Jejím charakteristickým rysem je uložení řídícího mikroprocesoru do řídící kostky NXT.

V roce 1998 byl v londýnském Museum of Modern Art představen nový produkt společnosti LEGO s názvem LEGO Mindstorms Robotic Invention Kit. V tehdejší době bylo možné zakoupit sadu, která obsahovala 717 kusů za 200 dolarů. Sada obsahovala mnoho součástek, které známe i ze současných modelů. Senzory, kolečka s gumovými plášti, ozubená kola, technické díly různých velikostí a tvarů a součásti pro spojování dílců stavebnice. Základem sady byla programovatelná kostka RCX Brick, která se stala nástupcem modelu MIT Programmable Brick. Kostka obsahovala tři vstupní a tři výstupní porty, které svým vzhledem připomínaly klasické díly LEGO. Horní panel obsahoval LCD displej a čtyři ovládací tlačítka. Komunikace mezi počítačem a kostkou byla zajištěna pomocí vysílání a přijímání infračerveného signálu. Kostka obsahovala 8 bitový mikrokontroler a paměť o kapacitě 16 KB. [2] [3] [4]

Obrázek 1 - Řídící jednotka RCX [5]

Rozšíření robotické stavebnice nabralo rychlý spád. Ještě v témže roce byla založena první robotická soutěž pro studenty středních škol s názvem FIRST LEGO League. Robotické sady se začaly dostávat s různými obměnami na trh nejen v Americe, ale také v Evropě a Asii. [6]

Platforma LEGO Mindstorms NXT 1.0 byla uvedena na trh v srpnu 2006. Řídící jednotka této verze obsahovala dva mikroprocesory. Hlavní byl 32 bitový Atmel ARM7 mikroprocesor s frekvencí přístupu do paměti 48 MHz, 256 KB FLASH pamětí a 64 KB RAM. Řídící jednotka obsahovala také pomocný 8 bitový mikroprocesor Atmel AVR s frekvencí přístupu do paměti 8 MHz, 4 KB FLASH pamětí a 512 B RAM. Pro komunikaci s počítačem byla řídící jednotka opatřena komunikačním USB 2.0 portem. Připojení vstupních zařízení bylo možné pomocí čtyř vstupních portů, pro výstupní zařízení byly k dispozici tři porty výstupní. Propojování bylo možné pomocí speciálních šestipramenných vodičů. Výjimku tvoří světelné kostky, které byly převzaty z předchozí verze stavebnice. Pro jejich připojení byly použity vodiče s redukcí RCX. Čelní strana řídící jednotky byla opatřena černobílým displejem o rozměrech 100 x 64 pixelů (26 x 40,6 mm). Ovládání a navigace v menu bylo prováděno pomocí čtyř tlačítek umístěných pod displejem. V programech bylo díky vestavěnému reproduktoru poprvé možné využít také zvukovou signalizaci. Řídící jednotka byla opatřena bluetooth modulem, díky čemuž bylo umožněno komunikovat s dalšími zařízeními. V základní sadě stavebnice LEGO Mindstorms NXT 8527 byly kromě řady technických dílů obsaženy ještě čtyři základní senzory (světelný, dotykový, ultrazvukový a zvukový). [7] [8]

Přesně o tři roky později, v srpnu 2009, byla oficiálně představena nová verze stavebnice LEGO Mindstorms NXT 2.0. Verze 2.0 je ve své podstatě totožná se stavebnicí verze 1.0. Rozdílem je přidání barevného senzoru do základní sady. Došlo také k aktualizaci firmware řídící jednotky. Od verze 2.0 lze pracovat s čísly s pohyblivou řádovou čárkou. [6]

Obrázek 2 - Řídící jednotka LEGO Mindstorms NXT [9]

Nejnovější verzí stavebnice je model LEGO Mindstorms EV3, který byl představen v lednu roku 2013 na výstavě International Consumer Electronics Show u příležitosti 15. výročí

vzniku LEGO Mindstorms. [6] Řídící jednotka této verze stavebnice se dočkala několika změn. Disponuje procesorem o frekvenci 300 MHz a pamětí s kapacitou 64 MB. Na řídící jednotku byl nově přidán port pro microSD kartu (podporována je také karta SDHC), pomocí které lze více rozšířit kapacitu řídící jednotky. Novinkou je podsvícení čelního panelu s tlačítky, které vyjadřuje možné stavy jako spouštění, upozornění, zaneprázdnění a podobně. Podsvícení či blikání panelu lze vyvolávat programově. Stejně jako řídící jednotka verze NXT 2.0 obsahuje i ta ve verzi EV3 čtyři vstupní porty. Novinkou je rozšíření počtu výstupních portů ze tří na čtyři. Vedle portu pro SD kartu je na řídící jednotce umístěn také USB port, který je možné použít pro připojení USB Wi-Fi adaptéru k připojení do sítě nebo vzájemnému propojení více řídících jednotek EV3. [10]

Přesto, že v současné době existuje verze EV3, práce se zaměřuje na předchozí model NXT 2.0. Tento model je rozšířen na značném počtu škol, které se zapojují do různých aktivit a soutěží postavených na jeho využití. V době zadání práce nebyla EV3 k dispozici (v ČR prodávána až ve 3. čtvrtletí roku 2013) a jediné podporované programovací prostředí pro tuto verzi bylo EV3. Podpora další programovacích prostředí jako je RobotC je plánována až na počátek roku 2014.

Obrázek 3 - Řídící jednotka EV3 [9]

1.2 CHARAKTERISTIKA LEGO MINDSTORMS NXT

Uživatel nebo vzdělávací zařízení, které začne využívat robotickou stavebnici LEGO Mindstorms NXT má k dispozici širokou paletu různých technických dílů a modulů. Používání stavebnice není omezeno pouze na díly obsažené v základní sadě. Nalezneme mnoho dalších výrobců robotických součástí a senzorů, které LEGO Mindstorms NXT podporuje.

1.2.1 ZÁKLADNÍ SADA STAVEBNICE NXT 2.0

Hlavní částí základní sady stavebnice LEGO Mindstorms NXT 9797, která zařizuje její programovatelnost, je řídící jednotka NXT. Jednotka obsahuje vestavěnou mikroprocesorovou jednotku. Její čelní strana obsahuje černobílý displej s ovládacími tlačítky. Obě strany jednotky jsou osazeny třemi výstupními a čtyřmi vstupními porty. Řídící jednotka je opatřena také USB rozhraním, které umožňuje propojení s počítačem pomocí standardního USB vodiče. K dispozici je také možnost bezdrátového přenosu dat pomocí vestavěného bluetooth modulu.

Ovládání, pohon a celkový pohyb robota obstarávají a ovlivňují různá vstupní a výstupní zařízení. Tato zařízení nazýváme moduly. Jedná se o komponenty, které jsou k řídící jednotce připojeny pomocí některého z portů. Díky tomu, že každý modul má specifickou funkci, se robot může pohybovat, reagovat na okolní prostředí, zjišťovat různé veličiny či reagovat na vstupy. Mezi výstupní moduly sloužící k pohonu robota řadíme servomotory nebo lineární motory. Do skupiny vstupních modulů řadíme rozličné typy senzorů, které slouží k orientaci robota v prostoru nebo k zjišťování různorodých dat. Mezi moduly řadíme také další rozšiřující zařízení sloužící k ovládání nebo k připojování dalších modulů. Řadíme sem různé druhy multiplexerů a slučovačů, které nám nabízejí nadstandardní možnosti při práci se vstupními a výstupními zařízeními. [11]

Co se výstupních zařízení týče, základní sada obsahuje tři servomotory a dále světelné výstražní lampy, což jsou vlastně LEGO kostky opatřené vláknovou žárovkou se zvýšenou odolností. Za výstupní zařízení můžeme považovat také displej řídící jednotky, na který je možné prezentovat data v různé podobě. Řídící jednotka je opatřena zvukovým výstupem, díky čemuž může být program doplněn o různá zvuková upozornění.

Ze vstupních zařízení obsahuje základní sada čtyři typy senzorů. Ultrazvukový senzor, který slouží k měření vzdálenosti k překážce za pomoci vysílání a následného přijímání ultrazvukového signálu. Dále zvukový senzor, který zpracovává a vyhodnocuje intenzitu zvuků v okolí. Třetím senzorem je světelný senzor, který pomocí čelně umístěného snímače měří intenzitu okolního světla. Posledním senzorem je dotykový senzor. Základní sada obsahuje tyto senzory dva. Někdy je senzor nazýván také tlačítko, protože při využití v programování má totožnou funkci.

Všechna připojitelná vstupní a výstupní zařízení jsou s řídící jednotkou NXT propojena pomocí speciálních šestipramenných vodičů. Vodiče se od sebe liší pouze svojí délkou. Odlišné vodiče jsou používány pouze pro připojení světelných kostek. Jelikož kostky byly převzaty do sady NXT 2.0 z předchozí verze RCX, bylo potřeba zajistit jejich kompatibilitu s novou řídící jednotkou. Vodič je proto opatřen RCX redukcí, která zajišťuje kompatibilitu mezi starým a novým rozhraním.

Obrázek 4 - Základní sada robotické stavebnice LEGO Mindstorms NXT [12]

Kromě vyjmenovaných vstupních a výstupních zařízení nalezneme v základní sadě dalších více než 570 technických dílů sloužících ke zkonstruování robota. V drtivé většině se jedná o díly LEGO Technic různých velikostí a profilů. Dále zde nalezneme kolečka opatřená gumovými plášti sloužící pro pohyb robota. Převody pohyblivých částí můžeme realizovat pomocí ozubených kol různých průměrů. Součástí stavebnice jsou také drobné spojovací dílce, které nabízejí různé možnosti propojování technických součástí. K sadě jsou dodávány dva barevné míče (modrý a červený), které slouží v úlohách na rozeznávání barev pomocí světelného senzoru nebo rozšiřujícího barevného senzoru, případně k realizaci třídící linky a dalších modelů, které konstruktér navrhne. Uživatel tak není při práci se stavebnicí nijak striktně veden, ale má široký prostor pro vlastní realizaci a fantazii.

1.2.2 ROZŠIŘUJÍCÍ MODULY

Při vytváření složitějších robotů a rozsáhlejších modelů, které budou vyžadovat pokročilejší orientaci v prostoru nebo zjišťování veličin, které nemůžeme pomocí modulů obsažených v základní sadě zpracovávat, je možné využít další senzory od jiných výrobců.

Mezi jedny z nejznámějších výrobců specializujících se na výrobu modulů pro robotickou stavebnici LEGO Mindstorms NXT jsou firmy Hitechnic a Mindsensors.

Společnost Hitechnic se zabývá výrobou rozšiřujících zařízení, která mají certifikaci od společnosti LEGO. Řídící jednotkou NXT jsou plně podporována a pro jejich bezproblémový chod je většinou nutné pouze stažení příslušných ovladačů pro použití ve zvoleném programovacím prostředí. Jednou skupinou modulů, kterými se společnost zabývá, jsou senzory. Jejich vzhled je totožný se senzory obsaženými v základní sadě stavebnice. Všechny jsou umístěny ve standardním krytu opatřeným úchyty pro montáž na tělo robota. Mezi senzory vyráběnými společností Hitechnic patří:

- senzor natočení - měří úhel natočení robota,
- senzor akcelerace - dokáže zjišťovat hodnotu zrychlení ve třech osách senzoru,
- barometrický senzor - slouží ke zjišťování atmosférického tlaku nebo teploty,
- barevný senzor - rozlišuje barevný odstín snímaného povrchu,
- EOPD senzor - slouží k detekci a případnému určení vzdálenosti tělesa od robota,
- senzor síly - dokáže měřit sílu působící v přímém směru na hřídel zasunutou v čelní části senzoru,
- gyroskopický senzor - vrací hodnotu otáčení robota ve stupních měřenou v jedné ose,
- IR senzor - za pomoci IR signálu umožňuje načítat v programu hodnoty senzorů připojených k řídící jednotce pomocí redukce RCX; jeho hlavní funkcí je řízení rychlosti R/C vlaků a modelů jiných stavebnic,
- vyhledávač infračerveného signálu - detekuje směr, ze kterého byl přijat infračervený signál (využívá se při robotickém fotbale),
- přijímač infračerveného signálu - slouží k příjmu signálů z dálkového ovladače LEGO,
- kompasový senzor - s přesností na 1° vrací aktuální hodnotu azimutu na základě měření magnetického pole Země,
- senzor magnetického pole - detekuje intenzitu magnetického pole.

Hitechnic nevyvíjí pouze moduly v podobě senzorů, ale také další doplňková zařízení a sady. Neodmyslitelnou součástí soutěží v robotickém fotbale je infračervený elektronický míč. Ten vysílá infračervený signál a je tak možné jej detekovat pomocí vyhledávače infračervených signálů.

Počet vstupních portů řídící jednotky může být v určitých situacích limitující. Vytvoření složitější a rozsáhlejší konstrukce robota si bude vyžadovat použití více vstupních zařízení. Společnost Hitechnic umožňuje řešit tento problém pomocí multiplexer vstupů. K dispozici máme následující dva:

- multiplexer vstupů pro senzory,
- multiplexer vstupů pro dotykové senzory.

Prvně jmenovaný umožňuje prostřednictvím svých vstupních portů přivést až čtyři senzory na jediný vstupní port řídící jednotky. Jejich rozlišení je následně provedeno díky adresování na sběrnici I2C. Multiplexer vstupů pro dotykové senzory má naprosto totožnou funkci, ovšem na jeho vstupy lze připojit pouze dotykové senzory. S jeho pomocí tak lze vytvořit například ovládací panel složený z několika tlačítek v podobě dotykových senzorů. Robot bude následně reagovat na stisk jednotlivých tlačítek a vykonávat různé pohyby.

Dalším výrobcem rozšiřujících modulů pro LEGO Mindstorms NXT je společnost Mindsensors. Stejně jako společnost Hitechnic, tak i Mindsensors vyrábí rozšiřující senzory. Co se funkčnosti týče, firma nám ve své podstatě nabízí stejné typy senzorů jako Hitechnic. Jejich vzhled je ale naprosto odlišný. Pro jejich používání je nutné pouze stažení aktuálních knihoven ovladačů pro používané programovací prostředí. Mezi zařízeními Mindsensors podporovanými LEGO Mindstorms NXT nalezneme také několik netradičních modulů. Jedním z nich je světelný senzor, který se skládá z osmi malých světelných snímačů, které mohou vyhodnocovat polohu robota přesněji, než klasický světelný senzor. Toto zařízení je určené především pro realizaci úloh, ve kterých se robot pohybuje po čáře na základě snímání povrchu, po kterém se pohybuje. Dalším netradičním modulem je dotykový panel, který je možné připevnit na displej řídící jednotky. Panel tak překrývá plochu jejího displeje. Díky tomu můžeme zjišťovat, na jaké pozici dotykového displeje jsme se povrchu dotkly stylusem. Panel je připojen ke vstupnímu portu řídící jednotky.

Pro možnosti rozšíření počtu vstupních portů je možné využít několik multiplexerů vstupů. První z nich umožňuje přivést na jediný vstupní port hodnoty až ze čtyř senzorů. Další multiplexer se specializuje pouze na připojení jednoho, až čtyř dotykových senzorů

na jediný vstupní port. Poslední multiplexer je určen k připojení až tří rozšiřujících modulů na zvolený vstupní port. Jeho použití je vhodné spíše pro doplňkové moduly, jelikož nepodporuje nejčastěji používané LEGO senzory. Rozšíření počtu portů není možné pouze u vstupních portů, ale také u výstupních. Připojením multiplexeru výstupních portů můžeme k jedinému výstupnímu portu připojit najednou dva servomotory a navíc jeden digitální senzor.

Společnosti Hitechnic a Mindsensors nejsou jedinými výrobci rozšiřujících modulů pro LEGO Mindstorms NXT. Existuje mnoho dalších, kteří se specializují na jiné spektrum využití robotické stavebnice. Jedním z nich je společnost Verniere, která zajišťuje bohatou podporu pro využití robotické stavebnice ve vyučování přírodovědných předmětů. Vyrábí různá připojitelná měřící zařízení, která jsou následně využívána v hodinách fyziky, chemie či přírodopisu k realizaci různých pokusů a měření.

1.3 VYUŽITÍ VE VYUČOVÁNÍ

Robotická stavebnice LEGO Mindstorms NXT je výrobcem určena dětem od osmi let věku. Tento věk představuje žáky prvního stupně základní školy. Horní věkový limit není stanoven. Za její pomoci by tedy mělo být možné vést výuku na různých stupních a také typech škol. Schopnost práce se stavebnicí však určitě budou ovlivňovat senzorické a motorické schopnosti žáků a také úroveň abstraktního myšlení. Tyto schopnosti a dovednosti se však s postupem věku mění. Zaměříme se proto na to, jaké jsou senzorické a motorické schopnosti žáků v daných vývojových etapách, abychom zjistili, zda jsou v těchto obdobích schopni robotickou stavebnici plně využívat. Dále popíšeme úroveň rozvinutí abstraktního myšlení v jednotlivých obdobích, z čehož bude patrné, na kolik jsou žáci, v závislosti na svojí mentální vyspělosti, při řešení úloh limitování. Poté se podíváme na vytváření konstrukcí a modelů robotů. Popíšeme doporučený postup stavby a nastíníme i problémy, které mohou při stavbě nastat. Popíšeme problémy související s vytvářením programů a představíme možné typy programovacích prostředí, která jsou vhodná pro využití v určitém období vzdělávání. Nakonec vysvětlíme, jak je vhodné při začlenění robotické stavebnice do vyučování na jednotlivých stupních vzdělávání postupovat a v jakých předmětech je možné výuku realizovat.

1.3.1 VYUŽITÍ NA PRVNÍM STUPNI ZÁKLADNÍ ŠKOLY

Senzorické a motorické předpoklady

Motorické funkce dozrávají u dítěte již v předškolním věku. Postupně se vyvíjejí a zdokonalují také manipulační dovednosti. Těsně před nástupem povinné školní docházky dítě zvládá správně držet psací náčiní. Pohyby při jeho používání začínají být přesnější a cílenější. [13]

Psychomotorický vývoj dětí v předškolním věku a v období prvního stupně základní školy neprobíhá rovnoměrně. U určitých motorických dovedností můžeme registrovat vývojové zpoždění. Postupem času se ovšem rozdíly zmenšují, až vymizí úplně. [13]

Na začátku školní docházky dítě zvládá činnosti, ve kterých používá bimanuální souhru rukou. Dokáže se bez pomoci obléci, zapínat větší i menší knoflíky nebo vystřihovat jednoduché tvary. Přibližně v 6 až 8 letech věku zvládá složitější úkony za pomoci obou rukou. [13]

U praktických činností jsou pohyby nejprve směřovány do ramenního a loketního kloubu. V pozdějších letech se začíná vyvíjet jemná motorika soustředěná do zápěstí a jednotlivých prstů ruky. [14]

Nedostatečně rozvinutá jemná motorika bude na prvním stupni základní školy žáky limitovat hlavně při sestavování robota. Nebudou schopni umisťovat jednotlivé drobné technické díly na obtížné pozice a spojovat je ve složitější celky.

Pokud v takto mladém věku přistoupíme k využití LEGO Mindstorms NXT ve výuce, je vhodné provádět konstruování pouze podle plánku. Můžeme ovšem použít i některou z alternativ. Jednou z nich je robotická stavebnice LEGO WeDo, která je určena právě žákům mladšího školního věku. Stavebnice obsahuje standardní díly LEGO, které doplňují technické díly LEGO Technic podobné těm, které nalezneme u LEGO Mindstorms NXT. Díky podstatně větší velikosti dílů a jednoduchému systému spojování nebude žákům sestavení jednoduchého robota činit příliš velké problémy. Sada navíc obsahuje pouze jednoduchý motor a několik základních senzorů v podobě tlačítek pro možnost ovládání a není tak příliš odborná. Odlišný je i způsob programování robota.

Abstraktní myšlení a logické operace

Na prvním stupni základní školy ještě žáci nemají natolik rozvinuté myšlení, aby dokázali navrhovat svá vlastní a hlavně funkční robotická zařízení. Chybí jim zkušenosti, které získají až v dalších stupních vzdělávání.

V prvních letech školní docházky se podstatně mění způsob uvažování dětí. Žáci v tomto věku ještě nedovedou uvažovat abstraktně. Jejich uvažování přechází podle J. Piageta k úrovni konkrétních logických operací, která je charakteristická respektováním základních logických zákonů a reality. (Vágnerová, 2000, s. 148) Žák dokáže logicky odvodit různé trvalé vlastnosti objektů a jeho myšlení je založeno na konkrétních zkušenostech. Postupem času dochází vlivem školní docházky k rozvíjení dalších dovedností. Uvažování žáků se tak stává mnohem komplexnější. [15]

Rozvoj logických operací probíhá v dalších letech v rámci výuky. Tím, že se žáci snaží chápat souvislosti a vztahy mezi jednoduchými objekty a posléze mezi pojmy, se rozvíjí logické uvažování. Rozvíjena je také strategie uvažování, neboli schopnost vhodně volit způsob řešení problémů. [15]

Oproti dětem na začátku školní docházky se děti na přelomu prvního a druhého stupně dokážou zaměřovat na detaily a odhalit i méně zjevné vztahy mezi nimi. Při řešení problémů využívají myšlenkových operací a hledají jádro problému. Neaplikují tak metodu pokus omyl, pomocí které řeší problémy velmi často mladší žáci. [15]

Nerozvinuté abstraktní myšlení u dětí na prvním stupni základní školy je podstatným problémem znemožňujícím plnohodnotné nasazení robotické stavebnice do výuky. Žáci nedokážou uvažovat na základě představ. Činí jim problémy řešení složitější konstrukce. Při vytváření se žáci orientují hlavně na základě pozorování reálného světa a v jejich činnostech se odrážejí jejich získané zkušenosti.

Na druhou stranu, pokud použijeme stavebnici LEGO WeDo, mohou být k vytváření modelů motivování způsobem provedení stavebnice. Díly a postup vytváření se totiž podobají klasické stavebnici LEGO, kterou znají. Pokud je tedy cílem řešení úlohy vytvoření atraktivního modelu, bude snáze přijat žáky za vlastní. U žáků prvního stupně to jsou hlavně modely zvířat či autíček vytvářených pomocí jednodušších stavebnic.

Je proto dobré je se stavebnicí LEGO Mindstorms NXT v tomto období seznámit, ovšem pro její plnohodnotné využívání musí nejprve získat potřebné dispozice a zkušenosti, které získají právě při práci s některou jinou stavebnicí, adekvátní jejich věku.

Vytváření konstrukce robota

První věcí, kterou je nutné před vytvářením robota učinit, je vytvořit projekt nebo úlohu, kterou budou žáci řešit. Přístup ke konstrukci robota musí být vzhledem k popisovaným schopnostem a dovednostem žáků odlišný, než je tomu například na druhém stupni ZŠ. V pozdějších letech může námět k řešení pocházet i ze strany žáků. Na prvním stupni je žádoucí, aby vytvoření úloh bylo v režii učitele. V pozdějších letech může, z důvodu přibývajících zkušeností, námět k řešení pocházet i ze strany žáků. Učitel většinou přichází do hodiny s již hotovou konstrukcí robota. Žákům následně představí jeho funkci a výsledek, ke kterému by měli při vytváření dojít. Pokud je model pro žáky tohoto věku dostatečně atraktivní, působí jako výrazný motivační prvek pro to, aby žáci začali s tvorbou a lákalo je vytvořit si model vlastní. Velmi výrazným prvkem jsou zde stavebnice pro mladší žáky typu LEGO Wedo. Pro jejich využití existuje několik vzorových úloh, ve kterých má robot podobu zvířete či jednoduché hračky. Žákům tak může připomínat stavbu pomocí jiné, podobné stavebnice.

Vzhledem k jednoduššímu charakteru stavebnice LEGO Wedo, se její využití z pohledu vytváření konstrukce jeví na prvním stupni vhodnější, než LEGO Mindstorms NXT. U ní se totiž nepracuje pouze s díly, které se bez problémů a jednoduše spojují, ale používají se také prvky, které se mnohdy díky svojí drobné velikosti a ne vždy jednoduchému umístění, připojují obtížně. Mladším žáků by tak práce s ní mohla činit značné obtíže.

Sestavování vlastních modelů žáky prvního stupně tedy není příliš vhodné. Pokud je budeme směřovat k tomu, aby si práci s technickými díly vyzkoušeli, je dobré, aby sestavení jednoduchého modelu prováděli pomocí plánku. Díky tomu budou vedeni krok za krokem ke zdárnému výsledku. Navíc se učí pracovat podle návodu a dodržovat doporučený postup.

Programování robotické stavebnice

Práce s robotickou stavebnicí neobsahuje pouze stavbu zařízení. Finální výtvor je nutné následně oživit příslušným programem. K tomu se používají rozličná vývojová prostředí fungující na bázi různých programovacích jazyků. Jedno mají ale společné. Pro práci s nimi je důležité chápání samotných principů a konstruktů programování a daného programovacího jazyka a také rozvinuté logické myšlení, které je důležité při vymýšlení optimální podoby programu. Nedílnou součástí jsou rozvinuté základní schopnosti algoritmizace.

Na prvním stupni základní školy se žáci pod vedením učitele učí logicky uvažovat. Cílem je, aby pochopily souvislosti a vztahy mezi reálnými objekty. Učí se způsoby řešení jednoduchých problémů a rozvíjí se jejich strategie uvažování. [15]

Pro programování je důležité osvojení abstraktního myšlení. U žáků prvního stupně není ještě plně rozvinuto. K provádění různých logických operací a řešení problémů při vytváření programu tak nemají žáci potřebné předpoklady. Pro jejich rozvíjení se používají odlišné metody.

U žáků mladšího školního věku je vhodnější používat nástroje, u kterých je rychle a názorně vidět výsledek jejich činnosti. To o robotické stavebnici nemůžeme úplně říci. Sestavený program musíme nejprve nahrát do řídící jednotky a výsledek programování vidíme až po jeho spuštění. Pokud ovšem v programu uděláme chybu, robot nemusí provést činnost vůbec žádnou a žák tak okamžitě nemusí mít představu o tom, proč zařízení nefunguje.

Logické uvažování a vytváření správných postupů v podobě jednoduché algoritmizace je tak u takto mladých žáků vhodné rozvíjet spíše pomocí jednodušších nástrojů. Žáci se učí řešit jednoduché elementární úkoly, a tím se učí základům algoritmizace. Učí se programovat v jiných dětských vývojových prostředích, ve kterých se používá jednoduchý, většinou graficky orientovaný programovací jazyk, úměrný jejich věku. Mohou to být například programovací prostředí EasyLogo, Baltík nebo Scratch. Výhodou jejich používání je to, že žáci při programování získají okamžitou vizuální zpětnou vazbu. Názorně vidí, co se díky programu, který vytvořili, provádí a bude pro ně snazší reagovat

na případné chyby. Začlenění programování robotické stavebnice do výuky na prvním stupni základní školy je tedy spíše vhodné na jeho konci.

Východiska výuky v rámci různých předmětů

Pokud chceme žáky se stavebnicí seznámit, můžeme je nechat ovládat již sestaveného robota pomocí externího ovladače. Žáci si tak stavebnici prakticky vyzkouší. Ovládání pomocí joysticku nebo jiného dotykového zařízení, kterým může být mobilní telefon či tablet, navíc rozvíjíme jemnou motoriku žáků a jejich orientaci v prostoru potřebnou pro správnou koordinaci pohybů robota.

Dalším krokem může být vlastnoruční sestavení jednoduššího modelu robota podle předem připraveného plánku či manuálu. Mělo by se jednat o jednoduchou konstrukci, ve které nebude zapotřebí příliš složitého umisťování jednotlivých dílů, což by žákům prvního stupně činilo značné problémy. Činnost by sloužila jak k rozvoji jemné motoriky při práci s drobnými technickými dílci, tak i k rozvoji schopnosti dodržet doporučený postup a dojít s jeho pomocí k požadovanému výsledku.

Následně můžeme žákům umožnit oživit si robota vlastními silami. Pomocí jednoduššího programovacího prostředí si mohou sami vyzkoušet, jak se programování provádí. Nepřichází zde ovšem v úvahu vytváření komplexnějších úloh. Žáci by si pouze vyzkoušeli jednoduché příkazy sloužící např. k rozjetí a opětovnému zastavení robota. Zacházení více do hloubky a další rozvíjení programu není v tomto věku vhodné.

Robotickou stavebnici můžeme na prvním stupni základní školy zařadit do výuky informatiky. Přihlédneme-li ovšem k faktu, že nepovažujeme za vhodné začít v tomto věku s jejím programováním a také k tomu, že v Rámcovém vzdělávacím programu pro základní vzdělávání nenalezneme u vzdělávací oblasti Informační a komunikační technologie pro první stupeň žádný vhodný okruh učiva, do kterého by bylo možné ji zařadit, musíme zvážit jinou alternativu. Mnohem vhodnější se tak jeví její využití při výuce technické výchovy. Ve vzdělávací oblasti Člověk a svět práce nalezneme modul Konstrukční činnosti, jehož náplní je práce se stavebnicemi, jejich montáž a demontáž. Zde se proto LEGO Mindstorms jeví jako vhodná varianta. Můžeme s ní zde žáky seznámit a připravit je na její pozdější hlubší využívání v dalších stupních vzdělávání.

1.3.2 VYUŽITÍ NA DRUHÉM STUPNI ZÁKLADNÍ ŠKOLY

Senzorické a motorické předpoklady

Jemná motorika žáka se začíná prudce zlepšovat v období od 3 do 6 let věku. Zvyšuje se jeho schopnost vykonávat běžné činnosti. Neustálé zlepšování motorických schopností registrujeme i na druhém stupni základní školy přibližně do věku 15 let. Následně se začne ustalovat. V období puberty je vývoj utlumen a kulminuje přibližně kolem 17. roku. [13]

Z pohledu zlepšujících se motorických dovedností žáků tak získáváme v průběhu druhého stupně základní školy mnohem širší možnosti ve využití robotické stavebnice. Je nutné ovšem rozlišit dvě situace. Jedná se o období přechodu z prvního stupně na druhý a období konce druhého stupně. Pokud se na začátku druhého stupně žáci se stavebnicí setkávají poprvé, je vhodnější postupovat obdobně jako na prvním stupni. Díky lepším motorickým předpokladům již ovšem nemusíme používat jednoduchou stavebnici typu LEGO WeDo, ale je možné rovnou začít s LEGO Mindstorms NXT. Použijeme-li jednodušší úlohy a dáme žákům k dispozici plánek pro stavbu, měli by být schopni robota sestrojit. Mnohem větší možnosti skýtá použití ve výuce v pozdějších ročnících druhého stupně. Kulminující motorické schopnosti umožňují snadněji využívat i drobnější díly.

Abstraktní myšlení a logické operace

V období středního školního věku, což je mezi 9. a 12. rokem dítěte, dochází k postupnému rozvoji logického myšlení. Tento přechod souvisí se změnou dětského uvažování a vnímání reality. [15]

U dětí v dalších letech nastává období puberty, ve kterém si osvojí způsob abstraktního uvažování. Díky tomu dokážou problém řešit jiným způsobem než doposud. Při řešení problému dokážou vymýšlet mnohé další možnosti řešení, které jsou v některých případech nereálné. [15]

Podle D. Keatinga (1991) je možné rozdíl mezi myšlením na úrovni konkrétních logických operací, které používají mladší děti a myšlením pubescentních dětí vyjádřit ve třech hlavních rysech. Pubescent klade důraz na uvažování o několika možnostech řešení. Nebere v potaz pouze skutečnost, ale zaměřuje se i na další možnosti. Snaží se najít lepší variantu. Ve svém uvažování postupuje systematicky. Pracuje s určitou hypotézou

a zamýšlí se nad různými možnostmi. Třetím rysem je to, že pubescent dokáže své myšlenky při řešení problému vzájemně kombinovat a začleňovat. (Vágnerová, 2000, s. 217)

Postupná změna v myšlení žáků je podstatnou dispozicí pro práci s robotickou stavebnicí. Ve větší míře se začne projevovat spíše v pozdějších ročnících druhého stupně. Díky tomu ovšem žáci nebudou při práci se stavebnicí pracovat pouze na základě svých dříve získaných zkušeností, jako tomu bylo v mladším školním věku. Při řešení úloh budou více experimentovat a kombinovat nové možnosti. Zaměří se více na detaily a měli by i více přemýšlet nad celkovou funkčností robota. Abstraktní myšlení umožní také posun v možnostech začlenění programování do výuky robotiky.

Vytváření konstrukce robota

Na začátku druhého stupně můžeme při využívání robotické stavebnice narážet na podobné obtíže, jako tomu bylo na prvním stupni. S postupným rozvojem jemné motoriky žáků a vývojem abstraktního myšlení se ale schopnosti žáků budou zlepšovat.

V úvodu je vhodnější přistupovat ke stavbě robotů na základě plánku. Pro ještě lepší názornost je dobré, aby žáci vytvářeli model, který jim předtím vyučující představil a případně vytvořil jako vzor. Pomůže jim to překonat počáteční obtíže. Vytváření robotů a robotických konstrukcí je tedy dobré provádět postupně, od nejjednodušších po složité. Díky tomu si žáci vyzkouší, jakým způsobem se jednotlivé technické díly spojují a jaké základní spoje se využívají při sestavování modelu.

Až se žáci se stavebnicí seznámí a naučí se řešit některé problémy, můžeme přejít k vytváření vlastních modelů. V počátcích mohou žáci stavět pouze jednoduchá vozítka, na která není potřeba umisťovat příliš velké množství technických dílů. Postupně je možné přecházet ke složitějším úlohám, které budou vyžadovat propracovanější konstrukci. Možností je také navržení základního modelu robota, na kterého budeme následně umisťovat další moduly, až vznikne složitější zařízení plnící více funkcí.

Postupem času pravděpodobně výuka dojde do fáze, kdy žáci začnou navrhovat vlastní řešení a nebude již nutné je k němu striktně vést. Učitel zde bude vystupovat jako ten,

kdo přichází do hodiny s problémem k vyřešení. Žáci následně navrhují možná řešení a podobu konstrukce.

Programování robotické stavebnice

Rozvoj abstraktního myšlení úzce souvisí s vývojem logického myšlení žáka. V robotickém programování ho žáci nejvíce uplatní při programování ovládacího softwaru robota. Než přistoupíme na druhém stupni k programování, musíme vhodně zvolit programové prostředí. Musíme tedy zvážit, zda použít prostředí, ve kterém se programuje pomocí zápisu zdrojového kódu (např. RobotC) nebo prostředí, ve kterém je kód tvořen navzájem propojovanými grafickými bloky (např. NXT-G). Vhodnější variantou je v tomto věku použití grafického programovacího prostředí. Žáci se v kódu dokážou rychleji orientovat na základě vzhledu jednotlivých bloků. Navíc nemusí zapisovat žádný programový kód. Nepotřebují ani znát různé příkazy, funkce a syntaxi příslušného programovacího jazyka. Je pro ně snazší zvolit blok pro ovládání konkrétního modulu a nastavit jeho parametry.

Stejně jako u vytváření konstrukce robota, musíme i u programování začít od nejjednodušších kroků. Žáci se tedy nejprve učí řešit úlohy, ve kterých jsou pouze řízeny jednoduché pohyby robota. Většinou se jedná o rozjetí a opětovné zastavení. Postupně můžeme přidávat další úkony a začít i využívat různé moduly. Díky jejich rozšiřujícím funkcím se programy začnou rozrůstat a stávat komplexnějšími.

Východiska výuky v rámci různých předmětů

Postup začlenění do výuky může být podobný jako na prvním stupni základní školy. Jestliže se žáci nikdy dříve se stavebnicí nesetkali, je nejprve vhodné seznámit je s ní pomocí vyzkoušení ovládání hotového robota. Žáci tak uvidí, co může být výsledkem spojování různých dílů a vyzkouší si, jaké možnosti jim stavebnice nabízí a jak se mohou sestavené modely ovládat. Postupně mohou žáci řešit jednoduché úlohy, k jejichž řešení si sami sestaví jednoduchého robota, nejprve podle plánku, a následně sami. Třetím krokem výuky poté může být začlenění programování.

Výhodou využití robotické stavebnice na druhém stupni základní školy je její možnost zařazení do výuky ve více předmětech. Běžně se využívá hlavně ve výuce informatiky, kde se za její pomoci vyučuje algoritmizace a základy programování. Začleňuje se ovšem také do výuky technické výchovy, kde se s její pomocí rozvíjí hlavě jemná motorika žáků při

vytváření konstrukčních modelů a robotů. Na druhém stupni základní školy se navíc začíná s výukou fyziky. Některé moduly je možné využít také pro realizaci různých úloh a laboratorních měření. Žáci tak data mohou sami získávat, vyhodnocovat a aplikovat v příkladech. U některých pokusů mohou pouze přihlížet tomu, jak je provádí vyučující, jiné si mohou pomocí stavebnice vyzkoušet sami. Mohou také dlouhodobě zaznamenávat různá data z okolí. Pomocí barometrického senzoru atmosférický tlak, pomocí teplotního senzoru okolní teplotu a podobně. Následně ze získaných dat mohou vytvářet různé přehledy, analýzy či grafy v hodině informatiky. Díky tomu nám vzniknou i mezipředmětové vazby. Robotická stavebnice se na školách nepoužívá pouze ve vyučování, ale také v mnoha volnočasových aktivitách a kroužcích.

Pro žáky druhého stupně základní školy může být robotická stavebnice vhodným prostředkem k tomu, aby si vyzkoušeli své dispozice i pro případné budoucí studium na střední škole s technickým zaměřením. Získají totiž základní zkušenosti s konstruováním a také s programováním.

1.3.3 VYUŽITÍ VE VÝUCE NA STŘEDNÍ ŠKOLE

Senzorické a motorické schopnosti žáků

Ve středoškolském věku, tedy mezi 15. a 18. rokem je u studentů plně rozvinuta manipulace a spolupráce obou rukou. Na vrcholu by měla být také jejich schopnost vykonávat běžné denní aktivity. Na základě mnoha výzkumů je tento věk v souvislosti s rozvojem motorických schopností označován jako kulminační. Studenti by tak měli mít pro práci s robotickou stavebnicí velmi dobré předpoklady. [13]

Zvyšující se schopnosti se projeví hlavně v manipulaci se stavebnicí a při vytváření modelů. Postupem času se budou eliminovat problémy, které při spojování a usazování dílů vyvstávají. Pohyby studentů budou přesnější a vytváření robotů rychlejší.

Abstraktní myšlení a logické operace

Myšlení studentů střední školy se oproti úrovni nabyté na základní škole příliš nemění. Pro toto vývojové období je charakteristická pružná přizpůsobivost a schopnost využívat další nové způsoby řešení. Při řešení problému tak studenti sahají často k velmi radikálním

řešením. Vzhledem k nedostatku jejich zkušeností ovšem nemusí být správná. Mohou mít také tendence k podlehnutí svým emocím. [15]

Díky vysoké míře rozvinutého myšlení se při práci s robotickou stavebnicí budou do práce žáků promítat jejich vlastní nápady. Rozvinuté abstraktní myšlení ovlivní jejich náhled na problém. Měli by navrhovat originální řešení, nacházet při řešení obtíže a snažit se jim předcházet. Tyto schopnosti se projeví hlavně v pozdějších letech středoškolské docházky. V období přechodu ze základní školy pokrok patrně nebude ještě tak příliš velký.

Vytváření konstrukce robota

Vzhledem k vysoké intelektuální vyspělosti studentů střední školy lze k práci s robotickou stavebnicí přistoupit bez hlubšího představování. Studenti mohou pracovat na vytváření modelů podle plánku, ovšem spíše budou mít tendence vytvářet modely vlastní. Způsob myšlení v tomto věku je k tomu předurčuje.

Mnohem méně problémů, než žáci základní školy, by měli mít v pozdějším středoškolském věku také se spojováním drobnějších technických dílů. Měli by rychleji reagovat na případné problémy, které při stavbě robota nastanou. Pružněji by měli také upravovat konstrukci, aby vyhovovala potřebám aktuální úlohy.

Při stavbě složitějších modelů je možné zadávat úlohy ve skupinách. Studenti si tak vyzkouší práci v týmu a spolupráci na společném tématu. Budou muset zvažovat postup tvorby robota a společně navrhovat optimální řešení jeho konstrukce.

Programování robotické stavebnice

Střední škola či gymnázium již ve své výuce často obsahují výuku konkrétního typu programování či programovacího jazyka. Místo toho, aby se studenti zabývali zápisem programového kódu a výsledek řešení a správnost programu ověřovali výpisem nebo simulací vytvořeného programu, tak stavebnice LEGO Mindstorms NXT jim umožňuje ověřit si jeho správnost prakticky. Studenti vědí, k jakému výsledku by při vytváření měli dojít a tím, že program nahrají do řídící jednotky a spustí, získají zpětnou vazbu o správnosti svého řešení v podobě činnosti robota. Programování tak pro ně může být poutavější, pokud na jeho konci vidí konkrétní řešení, kterým je v našem případě požadované chování robota nebo chod jiného zařízení. Programování robotické

stavebnice tedy dává stejné možnosti jako programování v jiném jazyce a vývojovém prostředí, ovšem přidanou hodnotou je zde názornost. Studenti si postupně mohou vytvářet správné programovací návyky a zásady, osvojovat si různé programovací konstrukty a chápat jejich význam, který je v různých programovacích prostředích podobný, v určitých případech totožný.

Pokud se studenti s programováním ještě nesetkali, může pro ně být vhodnější použít nejprve grafické programovací prostředí, ve kterém snáze pochopí jednotlivé programové konstrukty. Program pro ně může být zpočátku srozumitelnější. Postupem času je možné přejít k programovacímu prostředí, ve kterém je program tvořen zápisem programového kódu.

Východiska výuky v rámci různých předmětů

Úroveň využívání robotické stavebnice ve výuce na střední škole či gymnáziu se liší v závislosti na typu školy. Pokud má škola technické zaměření s vysokou hodinovou dotací výuky informatiky, bude mít pro používání robotické stavebnice samozřejmě větší prostor. S technickým zaměřením souvisí také bližší vztah studentů k problematice. Z tohoto důvodu může být LEGO Mindstorms NXT použito ve větší míře. Vytvářené úlohy a programy mohou obsahovat obtížnější programové konstrukty a mohou být propracovanější. Podíváme-li se na školy s netechnickým zaměřením, je úroveň probíraného učiva a jeho volba na učiteli. Je nutné zvážit, jak moc důležitá je výuka robotického programování pro zaměření žáků. V tomto případě tak stavebnice nebude nejspíše sloužit pouze k osvojení programovacích návyků, ale k širšímu rozvoji logického myšlení a řešení logických úloh.

Robotická stavebnice tedy na střední škole či gymnáziu získává široké uplatnění. Jednou z možností je její zařazení do výuky informatiky v rámci programování. Druhou možností je zařazení do předmětu, který se přímo specializuje na programování. Díky velkému množství rozšiřujících modulů a výukových sad je možné také zařazení do výuky fyziky či chemie. Stavebnice zde nalezne využití při realizaci laboratorních pokusů a měření. Z měření mohou vznikat také různé projekty, které budou žáci realizovat ve skupinách.

Na přelomu základní a střední školy či gymnázia se žáci často zúčastní různých soutěží, které jsou zaměřené na robotické programování. Mohou tak porovnat své znalosti

se svými vrstevníky a zapojit se do zajímavých projektů a soutěží regionálního, národního či dokonce mezinárodního charakteru. Jednou z nich je First Lego League. Soutěž je určena dětským týmům od 10 do 16 let. Soutěžit mohou ve 4 různých disciplínách. V první části musí týmem sestavený robot v časovém limitu vykonat co největší počet úkolů. V další části žáci prezentují svůj vytvořený projekt na zadané téma. Následně popíší také design a postup tvorby robota. Hodnocena je navíc spolupráce a komunikace v týmu.[16]

1.3.4 VYUŽITÍ VE VÝUCE NA VŠ

Senzorické a motorické schopnosti žáků

Ve stádiu studia na vysoké škole jsou již senzorické a motorické schopnosti žáků plně rozvinuté. V tomto věku vývoj ustává a úroveň schopností zůstává na stejné úrovni až do přibližně 50 let. [13] Pro využívání robotické stavebnice tak v tomto věku z pohledu senzorických a motorických schopností nenalezneme žádné limity. Vývoj je samozřejmě individuální a tak také záleží na zručnosti každého jednotlivce.

Abstraktní myšlení a logické operace

Období studia na vysoké škole je někdy také nazýváno mladou dospělostí. Jedná se o období zhruba od 20. roku. Myšlení ve vysokoškolském věku ovlivňují hlavně zkušenosti. Při řešení problému již nedochází k jeho zjednodušování. Student dokáže brát v úvahu všechny jeho složky. Navíc dokáže být sebekritický. Co se myšlení týče, mladý člověk je v tomto věku flexibilní a otevřený. Je schopen kombinovat logický způsob uvažování a subjektivní přístup k řešení problému. [15]

Studenti vysoké školy by tak měli mít nejlepší předpoklady k řešení i složitých programových úloh. Plně rozvinuté abstraktní myšlení by jim mělo napomoci při vymýšlení různých alternativ robotické konstrukce. Pokročilé schopnosti logického myšlení naopak uplatní při řešení programů pro oživení robota.

Vytváření konstrukce robota

Při studiu na vysoké škole se s robotickou stavebnicí pravděpodobně setkají pouze studenti studující technické obory. Můžeme předpokládat, že jim konstruování bude poměrně vlastní. Mnohem méně problémů by jim mělo činit také spojování dílů. Mentální

vyspělost a během předchozího studia nastřádané zkušenosti navíc dávají výraznou výhodu k sestrojování i složitých zařízení. Při využívání robotické stavebnice ve vysokoškolské sféře se tak můžeme často setkat s rozsáhlými projekty řízenými velkým počtem rozšiřujících modulů. Mezi nimi můžeme jmenovat různé třídící linky, rozsáhlá manipulační zařízení či roboty orientujícími se v prostoru s velkou přesností díky mnoha modulům, které reagují na změnu hodnot zjištěných z okolí.

Programování robotické stavebnice

Na vysoké škole nalezneme výuku programování na každém technicky zaměřeném oboru. Jako jedna z alternativ pro jeho výuku se tak může jevit i robotická stavebnice LEGO Mindstorms NXT. Jakým způsobem bude výuka orientována, hodně závisí také na volbě programovacího prostředí. Budeme-li chtít s její pomocí vyučovat programování v jazyce Java, můžeme použít programovací prostředí NXJ. Naopak na syntaxi jazyka C jsou založena prostředí RobotC a NXC.

Programování na vysoké škole bude pravděpodobně více orientováno k využívání programovacích prostředí, ve kterých je program vytvářen zápisem programového kódu. Grafická programovací prostředí se na tomto stupni budou využívat pouze v případech, kdy budeme studentům chtít tuto variantu programování představit.

Náročnost programových úloh zde také stoupá. Vzhledem k rozvinutým schopnostem logického a abstraktního myšlení jsou studenti schopni samostatně řešit složitější úkoly. Měli by také dokázat ověřovat správnou funkci programu a případně pracovat s chybou. Problém by jim nemělo činit také zefektivnění programu a funkce robota.

Východiska výuky v rámci různých předmětů

První z několika možností využití robotické stavebnice na vysoké škole je ve studiu technických oborů pedagogického zaměření. Budoucí učitelé, kteří jednou mohou vyučovat za pomoci robotické stavebnice, zde poznají možnosti, které jim pro výkon budoucího povolání nabízí. Může to být pro ně jedna z budoucích možností aplikace teorie do praxe. Právě toto je situace, ve které je možné ve výuce použít i grafické programovací prostředí. Můžeme tak studentům představit několik možností programování robotické stavebnice.

Také na vysoké škole, stejně jako na škole střední, může být robotické programování využito jako prostředek výuky programovacího jazyka C nebo Java. U některých technických oborů se studenti v prvním ročníku setkají s výukou základů programování. Pro žáky, kteří se na střední škole s programováním nesetkali, může být programování pomocí robotické stavebnice užitečným zpestřením výuky a pomůckou, která jim díky svojí názornosti usnadní prvotní osvojení programových konstruktů.

Robotická stavebnice se používá také v dalších technických oborech, které nejsou zaměřeny na vzdělávání. Setkáme se s různými odbornými pracemi, které jsou zaměřeny na ovládací či zabezpečovací prvky a jejich konstrukce a funkce je simulována na příkladech sestavených z robotické stavebnice. LEGO Mindstorms NXT zde plní funkci náhrady za mnohem nákladnější prvky, které jsou běžně v této oblasti používané.

Nejenom LEGO Mindstorms NXT, ale i další stavebnice se používají i ve sférách nejodbornějších jakou může být kybernetika a obory zabývající se složitějším řízením. Opět mohou být využívány během studia pro simulaci jednotlivých konstruktů pozdějších složitějších zařízení.

2 PROGRAMOVACÍ PROSTŘEDÍ PRO LEGO MINDSTORMS NXT

Programování robotické stavebnice LEGO Mindstorms NXT je zajištěno pomocí specializovaných programovacích prostředí. Standardním doporučeným nástrojem pro programování je ikonické programovací prostředí NXT-G, které je navržené firmou National Instruments. Není ovšem jediným prostředím, které můžeme využívat. Existuje mnoho dalších. Jedním z nich je programovací prostředí RobotC, které je založené na programovacím jazyce C. Toto prostředí je možné využívat pouze po zakoupení patřičné licence. Volně dostupné je programovací prostředí NXJ, jehož základem je programovací jazyk Java. Vlastní programovací jazyk založený na syntaxi programovacích jazyků Modula a Pascal nazvaný Lua používá pro programování robotické stavebnice programovací prostředí PBLUA. Nalezneme i mnoho dalších, než tyto vyjmenované. My se dále zaměříme na programovací prostředí NXT-G a RobotC a jejich možnosti využití v programování robotické stavebnice ve školním prostředí.

2.1 NXT-G

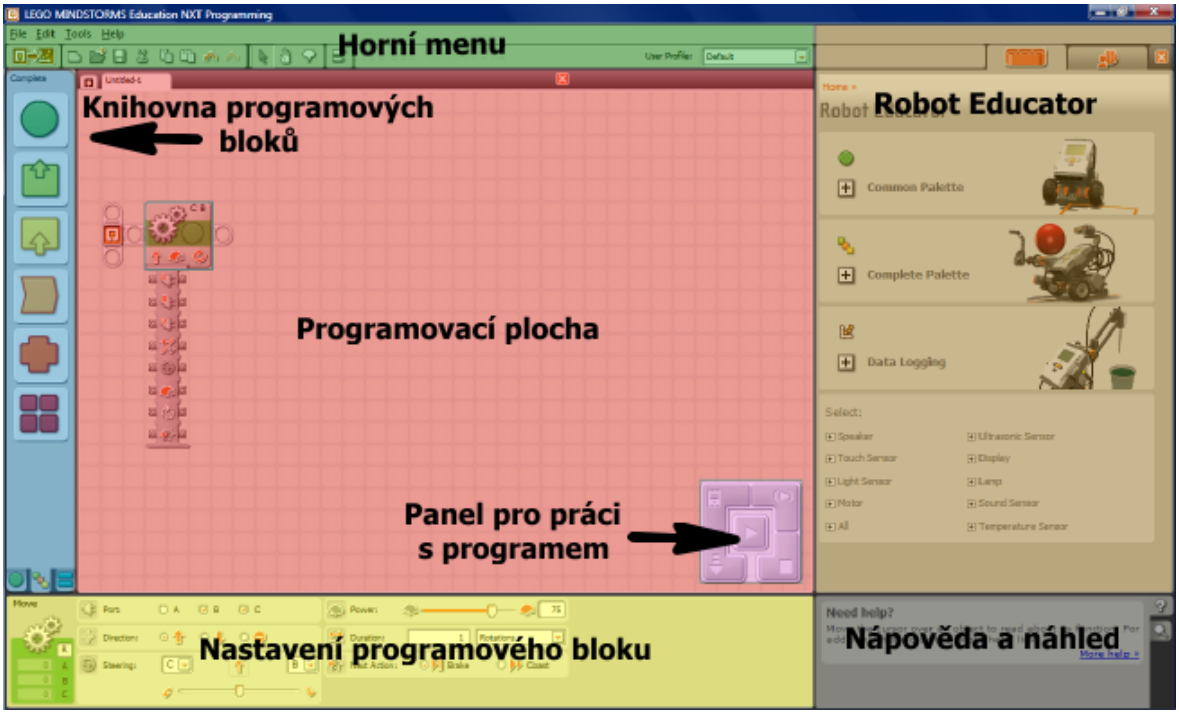
Programovací prostředí NXT-G je základním vývojovým softwarem dodávaným se stavebnicí LEGO Mindstorms NXT. Výrobcem je určeno od osmi let věku. Jedná se o takzvané ikonické vývojové prostředí. Zdrojový kód programu je totiž složen z ikon v podobě jednotlivých programových bloků, které lze propojovat.

Pro jeho využívání a bezproblémový chod musí náš počítač splňovat následující minimální systémové a hardwarové požadavky:

- procesor Intel Pentium nebo jiný kompatibilní s frekvencí 800 MHz a vyšší,
- operační systém Windows XP SP2 nebo Windows Vista a novější,
- 256 MB RAM,
- 500 MB volného diskového prostoru,
- monitor s rozlišením 1024x768 pixelů,
- alespoň jeden dostupný USB port (pro připojení řídící jednotky k počítači).
- volitelně bluetooth adaptér (pro komunikaci s řídící jednotkou pomocí bluetooth). [17]

2.1.1 POPIS PROSTŘEDÍ

Okno programovacího prostředí NXT-G můžeme rozdělit do několika částí. Na Obrázku 5 je můžete vidět barevně zvýrazněné.



Obrázek 5 - Programovací prostředí NXT-G s vyznačenými částmi

Světle zelená barva značí **horní menu**, ve kterém nalezneme záložky pro práci se soubory, pro úpravy programu, doplňkové nástroje programovacího prostředí a nápovědu.

Na levé straně je **světle modrou barvou** zvýrazněna **knihovna programových bloků**. Při jejím používání si můžeme vybrat mezi třemi typy zobrazení. Je možné si zobrazit zásobník pouze základních bloků, kompletní paletu všech bloků, či paletu vlastních vytvořených bloků. Knihovna je rozdělena do několika kategorií, ve kterých jsou příslušné bloky roztříděny. Nově přidávané bloky můžeme následně umisťovat do kterékoliv z nich.

Červená barva zvýrazňuje **programovací plochu**. Zde vytváříme program pro naše robotické zařízení. Na levé straně je zobrazen počáteční bod programu spolu s místem pro umístění prvního bloku. Programové bloky na plochu umisťujeme vybráním v příslušné části knihovny a přetažením na požadované místo v programu.

V pravém dolním rohu programovací plochy je bílou barvou zvýrazněn **panel pro práci s programem**. Obsahuje pět tlačítek. Levé horní tlačítko spouští funkci pro práci s pamětí

robotické jednotky. Ostatní tlačítka slouží k práci s programem. Jedná se o jeho spuštění, zastavení a nahrání do paměti řídící jednotky.

Ve spodní části, pod programovací plochou se nachází **žlutě** označená část pro **nastavení programového bloku**. Po kliknutí na některý z bloků programu se na tomto místě zobrazí veškeré volby pro jeho nastavení.

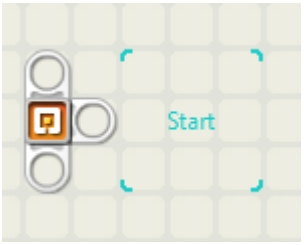
Zbývají poslední dvě části umístěné z pohledu uživatele vpravo. Větší z nich v horní části, která je na obrázku zvýrazněna **hnědou barvou**, se nazývá **Robot Educator**. Jedná se o rozšířenou formu nápovědy. Nalezneme zde popis a použití základních programových bloků a také ilustrační videa. Pomocí tlačítka v horní části je možné tuto sekci skrýt a získat tak přehled nad větší částí programovací plochy.

Poslední sekcí programovacího prostředí, kterou popíšeme, je okno v pravém dolním rohu, které je zvýrazněno **černou barvou**. Okno obsahuje dvě záložky. V horní části nalezneme odkaz na **online nápovědu**. Spodní záložka slouží jako **náhled programu**. Pokud vytvoříme rozsáhlejší programovou konstrukci, neuvidíme ji na programovací ploše celou. Díky této funkci se v programu můžeme orientovat a posouvat. V náhledu je vždy vidět celý program, ve kterém je zvýrazněna aktuálně zobrazovaná část programu na programovací ploše.

2.1.2 ZPŮSOB ZÁPISU PROGRAMOVÉHO KÓDU

Programovací prostředí NXT-G je nazýváno jako ikonické programovací prostředí. To znamená, že program je vytvářen na základě vkládání, logického uspořádávání a propojování ikon v podobě programových bloků. Každý blok symbolizuje funkci některého z používaných modulů a programových konstruktů. V zásobníku programových bloků, který nalezneme v programovacím prostředí z pohledu uživatele vlevo, jsou umístěny bloky pro ovládání různých vstupních a výstupních zařízení, provádění matematických operací, bloky pro programové řízení, pro práci s proměnnými a konstantami a mnohé další. Pokud chceme využívat rozšiřující moduly, musíme je do prostředí doinstalovat. Bloky určené k ovládání speciálních modulů nalezneme na stránkách výrobce daného modulu.

Samotný program se vytváří na pracovní ploše umístěné uprostřed, která zabírá největší část programovacího prostředí. Začátek programu označuje symbol s logem NXT-G. Nápisem Start je vyznačeno místo pro umístění prvního bloku programu.



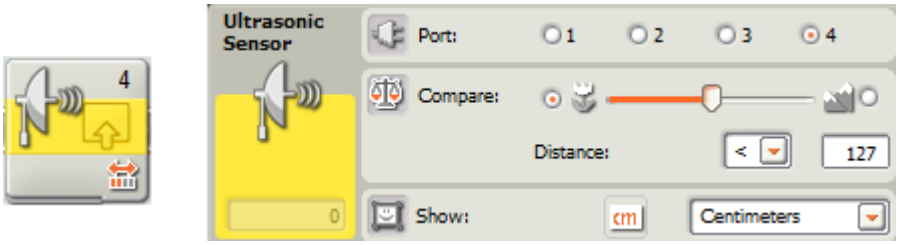
Obrázek 6 - Počáteční bod programu v NXT-G

NXT-G umožňuje i paralelní programování. Díky tomu nemusí být program pouze lineární. Paralelní chod programu zajistíme roztažením některé ze dvou bočních větví umístěných v počátečním bodě programu. Na ní následně umístíme příslušnou část programu. Současně tak mohou být vykonávány až tři jeho na sobě nezávislé části.

2.1.3 PRÁCE S PROGRAMOVÝMI BLOKY

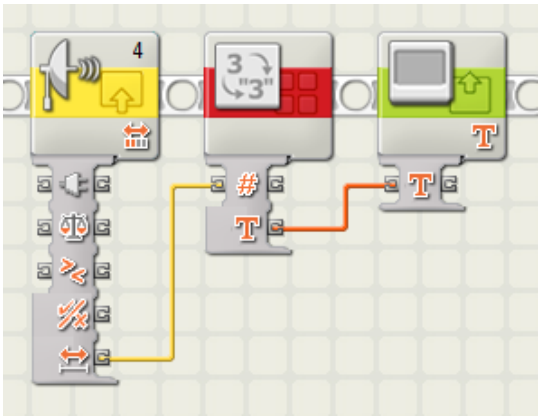
S bloky přetaženými na programovací plochu můžeme libovolně manipulovat a přemisťovat je do libovolné z větví. Musíme dát pozor, aby byly správně umístěny na větvi programu. Bloky, které budou umístěny mimo, nejsou brány jako součást programu. Po jeho kompilaci a spuštění se vykoná pouze kód umístěný na některé ze tří větví.

Blok přetažený na programovací plochu má čtvercový tvar. Je charakteristický svým grafickým provedením a ikonou, označující, jaká je jeho funkce. Po kliknutí na kterýkoliv z programových bloků (např. ultrazvukový senzor) se ve spodní části programovacího prostředí, přímo pod programovací plochou, zobrazí nastavení bloku. Každý blok obsahuje vlastní nastavení, které se odvíjí od jeho funkce. U ultrazvukového senzoru volíme port, ke kterému je připojen, vzdálenost určenou k porovnání s hodnotou snímanou senzorem a jednotky, ve kterých bude měření prováděno.



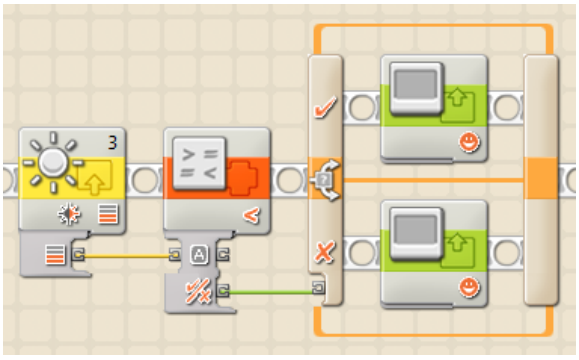
Obrázek 7 - Ukázka bloku ultrazvukového senzoru vlevo a nastavení bloku vpravo

Ve spodní části bloku umístěného na programovací ploše lze rozbalit rozbočovač, který obsahuje často značný počet konektorů. Konektory slouží hlavně k přenosu dat mezi jednotlivými bloky. Díky nim můžeme bloky různě propojovat. S jejich pomocí můžeme např. zobrazit hodnotu zjištěnou senzorem na displeji, řídit vykonávání cyklu, provést ukládání zjišťovaných či vypočítaných hodnot do proměnné nebo přenášet data při matematických operacích.



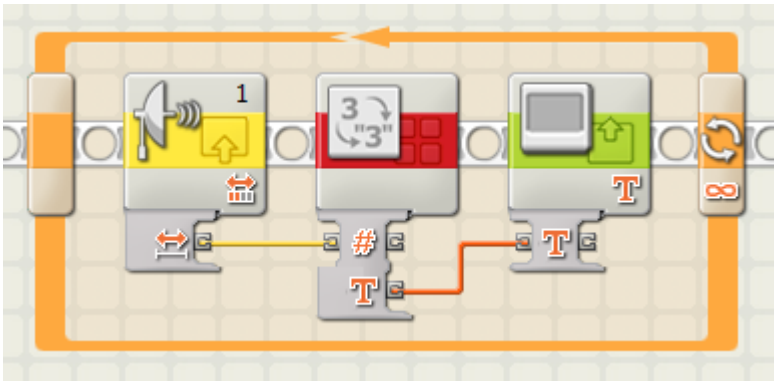
Obrázek 8 - Příklad přenosu dat mezi bloky pomocí vodičů

Netradiční vzhled oproti ostatním má blok podmíněného vykonávání Switch a blok pro použití cyklu Loop. Vykonávání podmíněného příkazu může být v NXT-G řízeno hodnotou nebo senzorem. Blok podmíněného vykonávání Switch navíc umožňuje ve svém nastavení přidat více podmínek, jejichž splnění následně ověřujeme. Tímto způsobem můžeme vytvořit přepínač case. Cyklus, který je v NXT-G reprezentovaný blokem Loop, může být řízen několika způsoby. Jedná se o řízení senzorem, časem, zadaným počtem průchodů nebo logickou hodnotou. Poslední možností je nekonečné provádění cyklu. Podmínky i cykly do sebe můžeme podle potřeby i vnořovat.



Obrázek 9 - Příklad použití podmínky

Programový kód, který se má vykonávat při splnění nebo nesplnění podmínky či při průchodu cyklem, musíme umisťovat do těla bloků. Tuto činnost vykonáme opět pomocí přetažení. Jakmile se umisťovaný blok ocitne nad jedním z těchto bloků, jeho tělo se přizpůsobí tak, aby do něj mohl být blok umístěn.



Obrázek 10 - Příklad použití cyklu

2.1.4 PROMĚNNÉ, KONSTANTY A DATOVÉ TYPY

V programovacím prostředí NXT-G můžeme při vytváření programu využívat tři datové typy. Jedná se o datový typ **Number** pro práci s čísly, **Text** pro práci se znaky nebo řetězci znaků a **Logic** pro práci s logickými hodnotami True (pravda) nebo False (nepravda). Díky tomuto jednoduchému členění tak např. u čísel nemusíme rozlišovat, zda se jedná o celá nebo reálná čísla a jakého budou číselného rozsahu.

Datové typy jsou využívány nejen u vstupů a výstupů modulů, ale je nutné je volit i při deklaraci proměnné. V programovacím prostředí NXT-G se proměnné a konstanty deklarují v horním menu programu. Při deklaraci volíme pouze název proměnné či konstanty a jeden ze tří vyjmenovaných datových typů.

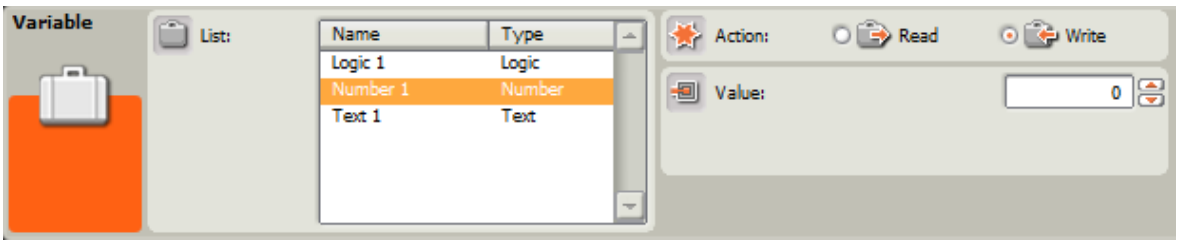
V programu jsou proměnné reprezentovány bloky **Variable** a konstanty bloky **Constant**. Oba bloky jsou označeny symbolem kufříku. Neměnnost hodnot uložených v konstantě symbolizuje zámek umístěný na kufříku.

Obrázek 11 - Bloky pro proměnnou (vlevo) a konstantu (vpravo)

Proměnnou můžeme použít buďto ke čtení nebo pro zápis. V závislosti na tom, se nám u bloku zobrazí příslušné konektory. Při čtení bude zobrazen pouze jeden výstupní, kdežto u zápisu vstupní i výstupní zároveň.

Obrázek 12 - Blok proměnné nastavený pro čtení (vlevo) a pro zápis (vpravo)

Režim proměnné volíme po kliknutí na programový blok v jeho nastavení. Zároveň musíme také vybrat, kterou z nadefinovaných proměnných chceme v aktuálním případě použít. V nastavení programového bloku můžeme také nastavit hodnotu, které bude proměnná nabývat.



Obrázek 13 - Nastavení programového bloku pro proměnnou

U použití konstanty je to podobné. Rozdíl je v tom, že u ní nenalezneme možnost pro zápis hodnot. Konstanta totiž oproti proměnné slouží pouze ke čtení fixně zadané

hodnoty. Musíme tedy pouze zvolit, kterou nadefinovanou konstantu budeme používat a zadat hodnotu, která v ní bude uložena.

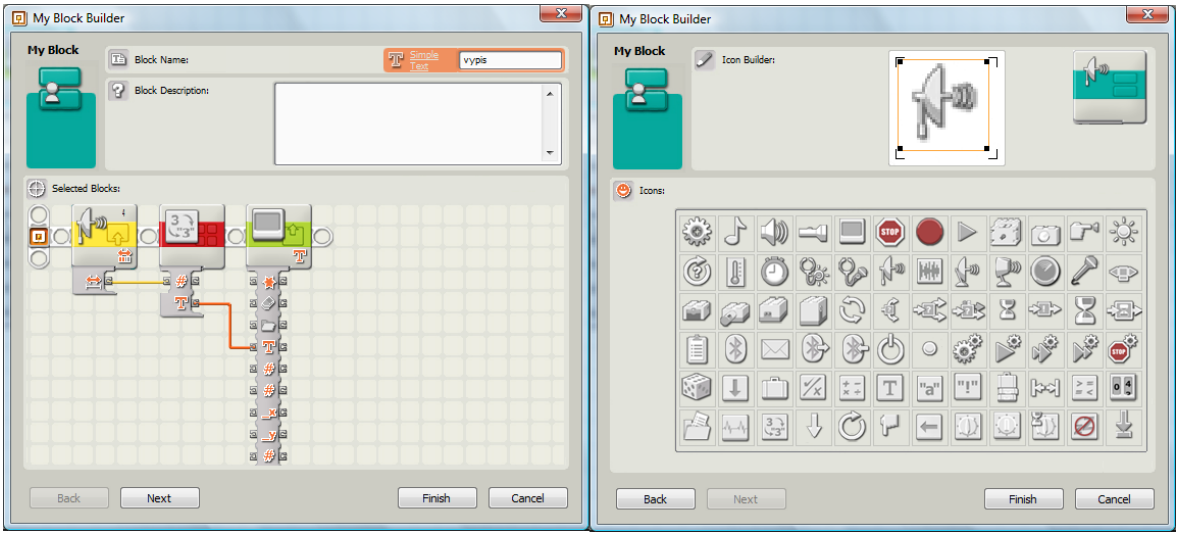
2.1.5 TVORBA VLASTNÍCH METOD

Vytváření programu pomocí logického skládání a propojování bloků může být sice na první pohled jednoduché a přehledné, ovšem můžeme narazit na situace, ve kterých se nám bude hodit použít vlastní definovaný blok. Námi vytvořený programový blok obsahuje část programu (několik programových bloků).

K vytvoření vlastního bloku nás může vést několik důvodů. Prvním z nich je potřeba zjednodušit rozsáhlý program. Díky narůstajícímu počtu bloků umístěných do programu, bude program nabývat na rozsáhlosti a může ztrácet přehlednost. Vytvořením vlastního bloku, který bude obsahovat určitou funkci, matematický výpočet, či jinou sekvenci kroků, můžeme tomuto problému předejít.

Druhým důvodem, který nás k vytvoření vlastního bloku může vést, je opětovná využitelnost bloku. V některých programech budeme často využívat určitý programový konstrukt nebo sekvenci příkazů. Opětovné vytváření může být zdlouhavé. Vytvořením vlastního bloku získáme možnost tuto část kódu použít opakovaně, v kterémkoliv programu, aniž bychom jí museli znovu vytvářet.

Před vytvářením vlastní metody musíme v programu nejprve označit bloky, které v ní budou obsaženy. Volbu pro vytváření následně nalezneme v horním menu. Blok může být buďto zcela samostatný nebo může obsahovat i vstupy a výstupy. Proces jeho tvorby se skládá ze dvou kroků. V prvním kroku si volíme název bloku, který můžeme případně doplnit o komentář, který nám v budoucnosti může usnadnit orientaci v tom, co blok vykonává. V druhém kroku vytváříme grafickou podobu ikony, která bude použita v knihovně programových bloků. Vytvořený blok se v knihovně programových bloků uloží do zásobníku pro vlastní bloky Custom palette, do sekce My Blocks. [18]

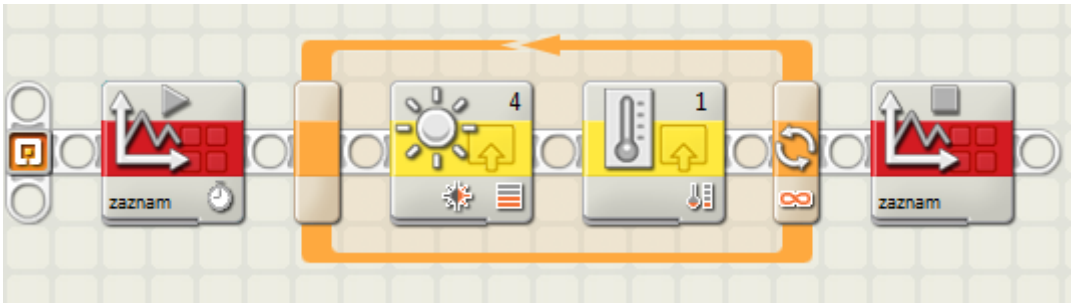


Obrázek 14 - Popis (vlevo) a grafický návrh (vpravo) při vytváření vlastního bloku

2.1.6 ANALÝZA DAT

Při použití v praxi se může objevit potřeba nejen hodnoty zpracovávat, ale také přesně vyhodnocovat jejich úroveň v určitém okamžiku nebo situaci. Programovací prostředí NXT-G obsahuje vestavěnou funkci Data Logging. Ta slouží k zaznamenávání a vizualizaci dat zjištěných senzory.

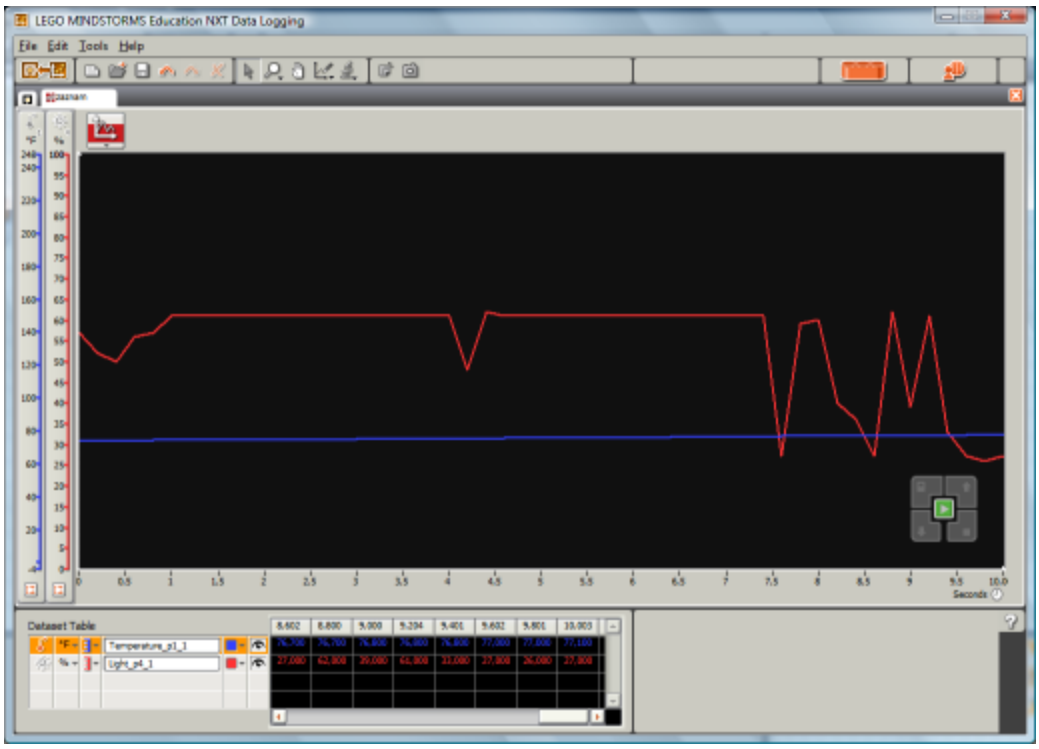
Funkci můžeme využít po celou dobu programu nebo můžeme testovat pouze jeho určitý úsek. Pro spuštění záznamu hodnot slouží blok Start Datalog. V nastavení tohoto bloku volíme, jak dlouho má záznam trvat, jaká má být frekvence snímání hodnot, a také senzor, jehož data chceme zaznamenávat. Určit také musíme, ke kterému portu je senzor připojen.



Obrázek 15 - Použití bloků pro Data Logging

Po spuštění programu jsou hodnoty zaznamenávány do grafu na základě nastavené vzorkovací frekvence. Pokud zjišťujeme více hodnot současně, křivky grafu jsou barevně odlišeny. Pro každý modul se navíc vlevo zobrazuje vlastní svislá osa hodnot s příslušnými

jednotkami. Hodnoty zjištěné v jednotlivých časových okamžicích jsou současně zaznamenávány do tabulky pod grafem.



Obrázek 16 - Funkce Data Logging

Funkci Data Logging využijeme např. při laboratorních pokusech, kde můžeme do grafu a tabulky zaznamenávat hodnoty ze zvukového, teplotního nebo barometrického senzoru. Využitelný je také v situacích, ve kterých je potřeba zajistit stabilitu robota. Můžeme analyzovat hodnoty z gyroskopického nebo akceleračního senzoru, zjišťovat jak se robot při svém pohybu chová a na základě získaných dat upravovat program a tedy i chování robota.

2.1.7 UŽIVATELSKÁ PODPORA

Spolu se zakoupenou sadou robotické stavebnice a instalací programovacího prostředí NXT-G získáme také uživatelskou příručku. Ta se zaměřuje na obecné představení robotické sady. Nalezneme v ní ovšem také základní informace o programovacím prostředí. Slouží jako prvotní náhled pro nezkušeného uživatele. Představuje základní funkce a části prostředí, spolu s informacemi o vytvoření prvního jednoduchého programu.

V samotném programovacím prostředí se nachází sekce Robot Educator. Ta slouží jako rozšířená nápověda. Nachází se v ní popis jednotlivých programovacích bloků a vysvětlení práce s nimi. Kromě toho zde nalezneme i návody k řešení jednoduchých programových konstruktů. Vše je doplněno obrázky, které vysvětlují postup tvorby programu krok za krokem, a demonstračními videi. Nevýhodou Robot Educator může být fakt, že je kompletně v anglickém jazyce.

Mnoho užitečných informací je možné nalézt také na oficiálním webu výrobce. V současné době se ale spíše věnuje rozvoji nové verze EV3. Při případných problémech je tedy vhodnější vyhledat pomoc na jiném webu, který se věnuje používání verze LEGO Mindstorms NXT. Podporou programování v programovacím prostředí NXT-G se totiž zabývá také mnoho neoficiálních webových stránek. Nalezneme zde diskuse nad řešeními různých programových úloh, některé náměty a případně také programy ke stažení. Většina z nich je zahraničních. Jedním z nejpovedenějších je např. web www.nxtprograms.com, na kterém nalezneme mnoho užitečných programových řešení. Každý projekt je doplněn o konstrukční řešení robota a ukázkové video jeho funkce. Ke stažení zde nalezneme také hotový program. U tohoto webu může být mírnou překážkou to, že je v anglickém jazyce.

České weby se zabývají z velké části robotickými soutěžemi. Některé užitečné materiály dávají k dispozici také různé katedry českých vysokých škol, které LEGO Mindstorms NXT využívají.

2.2 ROBOTC

Za vývojem programovacího prostředí RobotC stojí společnost Robomatter. Ta se zabývá vývojem tohoto prostředí nejen pro stavebnici LEGO Mindstorms NXT, ale také pro jiné platformy jako VEX, Arduino nebo TETRIX. Programovací jazyk RobotC je založen na principech programovacího jazyka C.

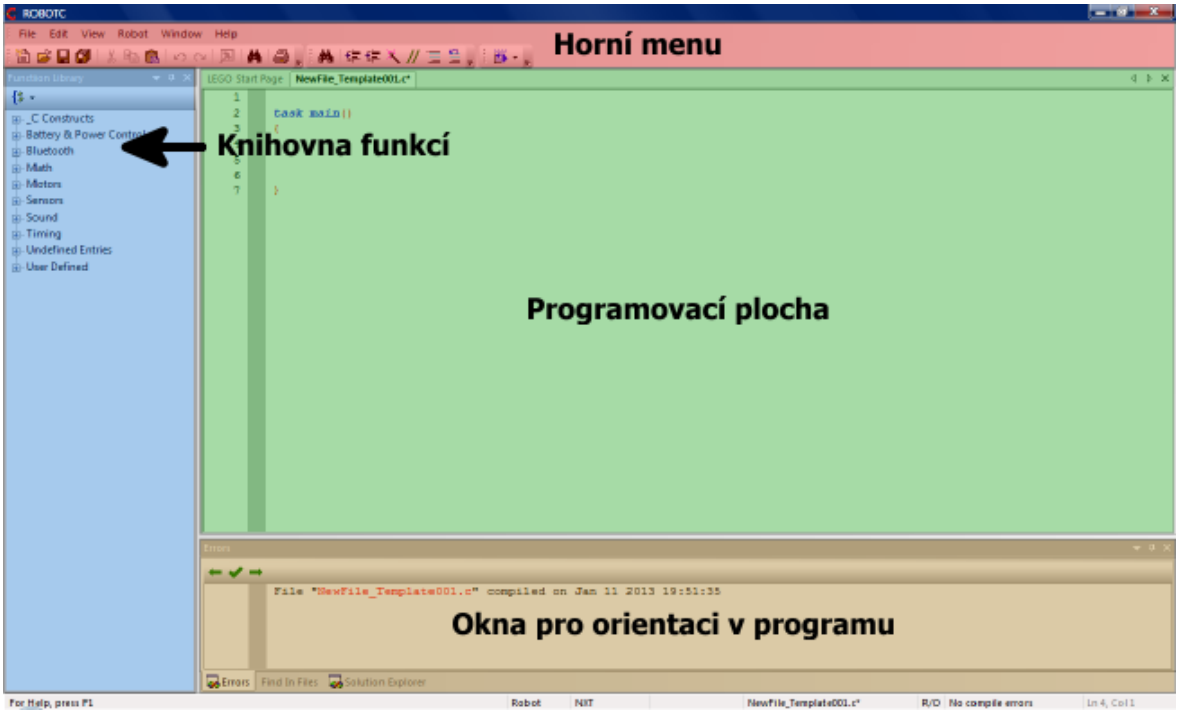
Pokud jej chceme pro programování robotické stavebnice využívat, musí náš počítač splňovat následující minimální systémové a hardwarové požadavky:

- procesor Intel Pentium nebo jiný kompatibilní s frekvencí 800 MHz a vyšší,
- operační systém Windows XP Professional nebo Home Edition SP2 a novější,

- 256 MB RAM,
- alespoň 30 MB volného diskového prostoru,
- alespoň jeden dostupný USB port (pro připojení řídící jednotky k počítači),
- volitelně bluetooth adaptér (pro komunikaci s řídící jednotkou pomocí bluetooth). [19]

2.2.1 POPIS PROSTŘEDÍ

Okno programovacího prostředí můžeme rozdělit na čtyři základní části. Barevně vyznačené je můžete vidět na Obrázku 17.



Obrázek 17 - Programovací prostředí RobotC s vyznačenými částmi

Červenou barvou je zvýrazněno klasické programové **horní menu**. Nalezneme v něm volby pro práci se souborem, jako vytvoření nového, otevření existujícího, uložení nebo zavření souboru. Dále také možnosti pro editaci zdrojového kódu a volby pro práci s vytvářeným programem, jako kompilace nebo nahrání do řídící jednotky. Menu obsahuje také mnoho možností pro přizpůsobení programovacího prostředí, případně spuštění doplňkových funkcí programu.

V levé části okna nalezneme **knihovnu funkcí**, která je na obrázku vyznačena **modrou barvou**. Knihovna obsahuje všechny funkce obsažené v základní instalaci prostředí. Slouží k dohledání potřebné funkce pro ovládání některého z modulů v případě, že uživatel

nezná její přesný zápis nebo zadávané parametry. Funkce jsou rozděleny do různých kategorií. Pro zvýšení přehlednosti je možno volit ze tří úrovní množství zobrazovaných funkcí (Basic, Expert, Super User). Při jejich použití je nemusíme po vyhledání v knihovně do programu ručně přepisovat, ale je možné umístit je na požadované místo v programu přetažením.

Největší část programovacího prostředí zabírá oblast zvýrazněná **zeleně**. Jedná se o **programovací plochu**. Zde vytváříme programový kód. Ten můžeme zapisovat přímo, nebo vybírat potřebné funkce z knihovny funkcí. Na levé straně programovací plochy se zobrazuje číslování jednotlivých řádků programu, což v něm umožňuje lepší orientaci při opravě chyb zjištěných během kompilace.

Poslední část, která se nachází dole pod programovací plochou a je zvýrazněna **hnědou barvou**, je sekce pro zobrazení **oken pro orientaci v programu**. Nalezneme zde okno pro chybová hlášení zjištěná při kompilaci programu nebo okno pro vyhledávání v souborech. Volitelně je zde možné spustit okno pro zobrazení bodů přerušení programu při jeho testování nebo okno pro záložky. Okna je možné používat jako fixní nebo je zobrazit jako plovoucí.

2.2.2 ZPŮSOB VYTVÁŘENÍ PROGRAMU

Program vytvářený v programovacím prostředí RobotC je založen na stejnojmenném programovacím jazyce, který je postaven na základě syntaxe jazyka C. Po vytvoření nového programu se na programovací ploše zobrazí jeho základní konstrukce. Příkazy zadáváme do sekce main.

```
1
2   task main()
3   {
4
5
6
7   }
```

Obrázek 18 - Základ nového programu v RobotC

Program můžeme v prostředí vytvářet zápisem programového kódu na programovací ploše. Zápis je možné provádět ručně nebo pomocí přetažení potřebné funkce

z jejich knihovny. Po přetažení se funkce na plochu vypíše bez svých parametrů, které musíme zadat.

Každý způsob vytváření nese i svá úskalí. Budeme-li programový kód zapisovat ručně, programovací prostředí nám začne po započetí zápisu funkce nebo příkazu nabízet pomocí našeptávače metody a funkce, které začínají na stejná písmena, jako ta která jsme zapsali. To může urychlit práci při vytváření programu. Musíme ale znát metody a funkce, které chceme použít. Vytváření pomocí přetažení z knihovny funkcí může vypadat jednodušeji. Jeho hlavní nevýhodou je ale pomalost. Nejprve totiž musíme potřebnou funkci nalézt. Ne vždy ovšem na první pohled uvidíme, v které kategorii se nachází. Následně ji musíme přetáhnout na programovací plochu. U každé funkce se nám do její konstrukce vypíše nápověda parametrů, které musíme zadat. Tu musíme smazat a zadat správné parametry. Vytváření celého programu tímto způsobem by tak bylo velice časově neefektivní. V RobotC lze vytvářet také paralelní běh programu. Více o možnostech paralelního programování v kapitole 2.2.4.

Pro vytváření programu v programovacím prostředí RobotC je nutné mít určité znalosti syntaxe a sémantiky tohoto programovacího jazyka. Pro lepší přehlednost programu a rychlejší orientaci v něm jsou jednotlivé elementy programového kódu na programovací ploše barevně odlišeny.

```
1  // deklarace ultrazvukoveho senzoru
2  #pragma config(Sensor, S1, ultrazvuk, sensorI2CHiTechnicAccel)
3
4  task main()
5  {
6   // pocatecni inicializace promenne
7   int vzdalenost = 0;
8
9   // nekonecny cyklus zarucujici neustaly vypis snimane hodnoty na displej
10  while(true)
11  {
12    vzdalenost = SensorValue(ultrazvuk);
13    nxtDisplayCenteredTextLine(0, "Zjistena vzdalenost");
14    nxtDisplayCenteredBigTextLine(2, "%d", vzdalenost);
15    wait1Msec(100);
16  }
17 }
```

Obrázek 19 - Ukázka programu v RobotC

2.2.3 DATOVÉ TYPY

Při deklaraci proměnných nebo konstant můžeme v programovacím prostředí RobotC využít poměrně širokou paletu datových typů. Liší se typem dat, rozsahem a hodnotami, kterých mohou nabývat.

V případech, kdy budeme ověřovat, zda byla splněna určitá podmínka, využijeme logický datový typ boolean. Jeho návratovou hodnotou je True (pravda) nebo False (nepravda) v závislosti na tom, zda podmínka byla splněna či nikoliv. Pro práci s řetězci znaků je určen datový typ string. Dále můžeme v RobotC využívat několik datových typů pro práci s čísly. Mezi nejčastěji používané číselné datové typy patří integer, který definuje kladná i záporná čísla včetně nuly. Druhým hojně používaným číselným datovým typem je float, který slouží k využití reálných čísel v programování. V případě potřeby můžeme využít další datové typy. Jejich kompletní přehled pro programovací prostředí RobotC, včetně jejich rozsahů a zkratek pro zápis v kódu, naleznete v následující tabulce (viz. Tabuka 1).

Datové typy v RobotC		
Datový typ	*Popis*	*Zápis v kódu*
Boolean	Logický datový typ - nabývá hodnot True nebo False.	bool
Byte	Celá čísla od -128 do 127.	byte
Char	Celá čísla od -128 do 127. Jejich hodnotu je možné vyjádřit znakem.	char
Float	Reálná čísla.	float
Long	Celá čísla od -2,147,483,648 do 2,147,483,647.	long
Integer	Celá čísla (kladná i záporná), včetně nuly od -2,147,483,648 do 2,147,483,647.	int
Short	Celá čísla od -32,768 do 32,767.	short
String	Řetězec znaků.	string
Word	Celá čísla od -32,768 do 32,767.	word
Ubyte	Celá čísla nabývající pouze kladných hodnot od 0 do 255.	ubyte
Void	Datový typ pro funkce bez návratové hodnoty.	void

Tabulka 1 - Přehled datových typů v RobotC [20]

2.2.4 TVORBA VLASTNÍCH METOD

Vytváření vlastních metod máme v programovacím prostředí RobotC k dispozici hlavně z toho důvodu, abychom si mohli zjednodušit psaní rozsáhlejšího programu a mohli na jeho více místech používat totožné funkce za pomocí pouhého jejich zavolání.

Vlastní metody můžeme buďto sami zapsat, nebo si jejich konstrukci přetáhnout z knihovny funkcí, kde jsou umístěny v kategorii _C Constructs. K dispozici máme dva typy metod. První z nich je vlastní funkce s návratovou hodnotou nebo bez ní a druhá dotaz task.

Vlastní funkce se zapisuje v úvodu programu. Při jejím zápisu musíme udat název a parametry funkce v podobě proměnných. Můžeme ji také přetáhnout z knihovny funkcí. Mezi složené závorky se následně umístí příslušné příkazy. Funkce může nebo nemusí vracet nějaký výsledek. To závisí na tom, jakého datového typu je její návratová hodnota. Na Obrázku 20 můžete vidět funkci bez návratové hodnoty. Ta je datového typu void. Pro použití v programu následně funkci zavoláme na libovolném místě pomocí jejího názvu a zadanou hodnotou jednoho či více parametrů zadaných v závorce.

```
1   void motory(int rychlost)
2   {
3     motor[motorA] = rychlost;
4     motor[motorB] = rychlost;
5     wait1Msec(3000);
6   }
7
8   task main()
9    {
10     motory(100);
11   }
```

Obrázek 20 - Vlastní funkce bez návratové hodnoty

Zápis funkce s návratovou hodnotou je naprosto totožný. Musíme ovšem zvolit datový typ, kterého bude výsledek funkce nabývat. K získání výsledku funkce nakonec použijeme příkaz return. Její použití v programu se provádí stejně jako u funkce bez návratové hodnoty. Příklad funkce s návratovou hodnotou můžete vidět na Obrázku 21.

```
1   int vypocet(int a)
2   {
3     a = (a + 5)*12;
4     return a;
5   }
6
7   task main()
8   {
9     int x = 4;
10    int vysledek = vypocet(x);
11    nxtDisplayString(1, "x: %d", vysledek);
12    wait1Msec(5000);
13  }
```

Obrázek 21 - Vlastní funkce s návratovou hodnotou

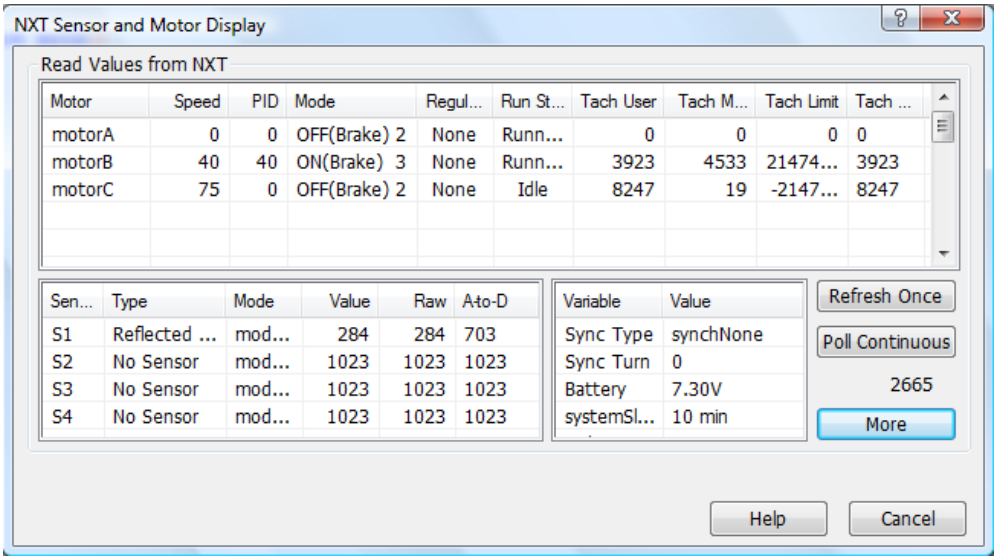
Druhou možností tvorby vlastních metod v RobotC je dotaz task. Ten je ve své podstatě zapsaná sekvence zdrojového kódu, která je následně v programu volána. Zapisuje se opět před hlavní částí programu. Uživatel volí název dotazu a následně umístí do jeho těla požadovanou část kódu. Dotaz se může volat na libovolném místě programu pomocí příkazu StartTask a zastavovat příkazem StopTask nebo StopAllTasks. Jeho nastavení umožňuje mnohé rozšířené možnosti. Můžeme např. pomocí speciálního příkazu pozastavit klasický plánovač úloh procesoru a využít jeho výkon pro vykonávání konkrétního dotazu. Pokud používáme v programu dotazů více, je možné jim přidělit prioritu vykonávání. Tím určíme, v jakém pořadí mají být plánovačem úloh zařazeny k vykonání.

```
1   task prehraj()
2   {
3       PlayTone(784, 500);
4       wait1Msec(3000);
5   }
6
7   task main()
8   {
9     StartTask(prehraj);
10  }
11
```

Obrázek 22 - Příklad jednoduchého dotazu task

2.2.5 ANALÝZA DAT

Programovací prostředí RobotC neobsahuje žádnou vestavěnou funkci pro záznam či vizualizaci dat. Pokud chceme data zaznamenávat, musíme tak učinit programově. V horním menu se ale nachází funkce Poll NXT Brick, která nám umožní data zobrazovat.



Obrázek 23 - Funkce Poll NXT Brick

Zobrazování probíhá v reálném čase pomocí propojení řídící jednotky NXT s počítačem prostřednictvím USB portu. V programovém okně se nám dynamicky zobrazují hodnoty naměřené připojenými motory a senzory. U motorů můžeme zjišťovat aktuální intenzitu otáčení nebo různé nastavené parametry, jako např. PID regulaci. Důležitou věcí, která nás u motorů může zajímat, pravděpodobně budou naměřené otáčky. Na jejich základě je možné analyzovat jednotlivé části programu a upravovat chování robota a jeho pohybu, případně měřit robotem ujetou vzdálenost.

V další části sledujeme hodnoty vracené senzory. Vyjádřeny jsou jak v klasické podobě, tak v čisté, neškálované formě RAW. Doplňkovou funkcí je zobrazení údajů o řídící jednotce. Jedná se o úroveň nabytí baterie řídící jednotky, nastavenou úroveň hlasitosti reproduktoru a dobu, po které se jednotka automaticky vypne.

Funkce obsahuje také rozšířené možnosti. Zde můžeme náš program testovat. Funkce Poll NXT Brick nám totiž umožňuje zasílat motorům hodnoty pro jejich ovládání nebo nastavovat režimy snímání senzorů.

2.2.6 UŽIVATELSKÁ PODPORA

Kolem programovacího prostředí RobotC se vytvořila silná komunita, která aktivně přispívá k jeho popularizaci a vývoji. Nejvíce informací se soustřeďuje na oficiální webové stránce www.robotc.net. Nalezneme zde informace o RobotC pro všechny podporované platformy stavebnice spolu s možností získání 30 denní zkušební verze programu. Web obsahuje také interaktivní kurz pro začátečníky. V něm se dozví základní informace o programování LEGO Mindstorms NXT pomocí RobotC. Kurz je doplněný i o komentovaná ilustrační videa. Američtí uživatelé mají k dispozici také možnost zapsat se do různých kurzů robotického programování pořádaných National Robotics Engineering Center v Pittsburghu. V rámci webových stránek funguje také diskuse, ve které můžeme získat okamžitou zpětnou vazbu na jakýkoliv dotaz týkající se robotické stavebnice nebo programovacího prostředí. Vývojáři také hojně přispívají na různé blogy, na které nalezneme na webu odkazy. Členové týmu pro podporu RobotC pořádají navíc živá vysílání prostřednictví sociální sítě Google+. V nich se věnují problematice vytváření programu v RobotC. Různí přispěvatelé na webu publikují své vytvořené projekty spolu s popisem. Nezkušení uživatelé tak zde mohou získat inspiraci pro vytváření vlastních robotů. Nevýhodou pro českého návštěvníka webu může být fakt, že web je kompletně v angličtině a ve stejném jazyce probíhá i veškerá komunikace na něm.

2.3 SROVNÁNÍ PROGRAMOVACÍCH PROSTŘEDÍ

Tato kapitola se zaměřuje na porovnání programovacích prostředí NXT-G a RobotC. Abychom zhodnotili společné, ale také odlišné funkce a možnosti, zvolili jsme si několik kritérií, na která jsme se následně podrobněji zaměřili. Výsledky jsou na závěr prezentovány v tabulkách a doplněny komentářem.

2.3.1 KRITÉRIA K POROVNÁNÍ

V rámci snahy o komplexní porovnání uvedených programovacích prostředí jsme se zaměřili jak na objektivní kritéria, která popisují jejich možnosti, tak i na subjektivní kritéria, která vychází z používání programu. Bylo vytvořeno celkem sedm rozsáhlejších skupin kritérií. Zaměřili jsme se na programovací jazyky, na nichž jsou obě programovací prostředí postavena. Porovnávali jsme jejich možnosti a některé aspekty použití. Zhodnotili jsme, jakým způsobem se v obou prostředích pracuje s moduly stavebnice

LEGO Mindstorms NXT. Zvolili jsme také kritéria, která se týkají základních, ale i doplňkových funkcí obou programovacích prostředí.

Programové řízení

Tato skupina kritérií se zaměřuje na možnosti řízení programu v obou programovacích prostředích. Zjišťovali jsme, jaké typy cyklů a podmíněných výrazů je v nich možné použít. Jako další jsme ověřovali, zda umožňují ovlivňovat vykonávání programu pomocí časovačů nebo pomocí oddálení vykonávání příkazu.

Proměnné, konstanty a jejich datové typy

U proměnných a konstant jsme zjišťovali, jakým způsobem v obou programovacích prostředích probíhá jejich vytvoření a jak se následně v programu používají. S proměnnými a konstantami souvisí také datové typy, jichž nabývají. Ověřovali jsme proto, které máme v každém prostředí k dispozici a jakých hodnot se týkají. Zaměřili jsme se také na to, zda je možné a nutné v programu provádět jejich přetypování.

Logické a matematické operace

Pro programování je nezbytné provádění logických operací. Porovnali jsme, jaké logické operace můžeme v obou prostředích použít. Zaměřili jsme se také na provádění výpočtů pomocí matematických operací. Postupovali jsme od základních operací po pokročilé matematické funkce. Na závěr jsme srovnali ještě možnosti využití intervalů v programu a generování náhodných čísel.

Možnosti výstupní signalizace

U výstupní signalizace jsme porovnávali, zda obě programovací prostředí umožňují využívat stejné typy zvukové a světelné signalizace. Zaměřili jsme se také na to, jakým způsobem a co se dá vykreslovat na displej a jak se provádí jeho mazání.

Práce s moduly

Jedny z hlavních částí robotické stavebnice jsou vstupní a výstupní moduly. Analyzovali jsme, zda se s nimi v obou programovacích prostředích pracuje stejným způsobem a jestli je při jejich používání nutné provádět stejné činnosti. Jedná se hlavně o konfiguraci, kalibraci, deklaraci před použitím v programu a jejich následné využití. Zjišťovali jsme

také, jaká je dostupnost ovladačů pro rozšiřující moduly. Poslední částí této skupiny je porovnání možností multiplexování vstupních a výstupních zařízení v obou prostředích.

Vlastní metody a paralelní programování

Do této kategorie byly zařazeny rozšiřující možnosti hojně využívané při programování. Ověřovali jsme, zda a jakým způsobem umožňují vytváření vlastních metod a paralelní programování. U paralelního programování jsme zkoumali jeho základní, ale i rozšiřující možnosti. Do rozšiřujících jsme zahrnuli např. možnost práce se semaforem.

Doplňkové funkce prostředí

Závěrečná skupina sledovaných kritérií obsahuje zjištěné doplňkové funkce programovacích prostředí. Porovnávali jsme zde možnosti zobrazování a zaznamenávání dat, vkládání komentářů do programového kódu či možnost vytváření uživatelských profilů. Zjišťovali jsme také, zda je v programovacích prostředích možné spravovat paměť řídící jednotky stavebnice. Sledovali jsme, jakým způsobem je uživatel při vytváření programu upozorňován na případné chyby v programovém kódu a jak se může v rozsáhlejším programu orientovat.

2.3.2 POSOUZENÍ VÝSLEDKŮ

Výsledky analýzy každé skupiny kritérií jsou uvedeny do tabulky a doplněny o komentář, který podrobněji popisuje odlišnosti na základě dílčích kritérií zkoumaných v dané oblasti.

Programové řízení			
Sledovaná kritéria		***NXT-G***	***RobotC***
Cykly	S podmínkou na začátku	✘	✓
	S podmínkou na konci	✓	✓
	S pevným počtem průchodů	✓	✓
Podmínky	Podmíněné příkazy	✓	✓
	Výběrové příkazy (case)	✓	✓
Časovače		✓	✓
Oddálení vykonávání příkazu		✓	✓

Tabulka 2 - Posouzení kritérií týkajících se programového řízení

Obě programovací prostředí umožňují řídit činnosti programu pomocí cyklů a podmínek. Programovací prostředí NXT-G obsahuje pro práci s cykly blok Loop. Cyklus může být prováděn neustále nebo může být řízen různými způsoby. Podmínka jeho provádění je ale pokaždé umístěna na konci cyklu. Na rozdíl od RobotC, ve kterém lze využívat cyklus s podmínkou na začátku, na konci i s pevným počtem průchodů, tak v NXT-G cyklus s podmínkou na začátku vytvořit nelze.

U podmínek neregistrujeme žádné odlišnosti. Jak v NXT-G, tak i v RobotC se dá použít podmínka řízená hodnotou nebo stavem zjištěným senzorem. Stejně tak můžeme v obou prostředích vytvořit výběrový příkaz case. Totožná je i práce s časovači. Jediný rozdíl je v jejich počtu. V NXT-G máme k dispozici tři časovače, oproti čtyřem, které nabízí RobotC. Pro řízení činností v programu obsahují obě prostředí ještě příkazy pro oddálení vykonávání příkazu (v NXT-G prováděno blokem Wait).

Proměnné, konstanty a jejich datové typy			
Sledovaná kritéria		***NXT-G***	***RobotC***
Vytvoření		v horním menu prostředí	deklarací v programu
Použití		prostřednictvím programového bloku	prostřednictvím názvu z deklarace
Datové typy	Počet	3	10
	Řetězcové	✓	✓
	Logické	✓	✓
	Celočíselné	✓	✓
	Reálné	✗	✓
	Pole	✗	✓
Přetypování	Možnost přetypování	✓	✓
	Nutnost přetypování	✓	✓

Tabulka 3 - Posouzení kritérií týkajících se proměnných, konstant a datových typů

Značným způsobem se v NXT-G a RobotC liší práce s proměnnými a konstantami. V NXT-G je deklarace poněkud zdlouhavější. Musí se provádět v horním menu programovacího prostředí, kde se zvolí její název a datový typ. Použití poté probíhá klasickým způsobem, tedy prostřednictvím programového bloku pro proměnnou nebo konstantu. Znovu ovšem musíme volit, o kterou proměnnou se jedná a zda bude použita pro čtení či zápis. Práce s proměnnými je tak poměrně uživatelsky nepřívětivá. V RobotC se deklarace provádí zápisem v programovém kódu. Skládá se z označení datového typu, názvu proměnné, případně hodnoty, které v okamžiku deklarace nabývá.

S využitím proměnných a konstant souvisí datové typy, kterých nabývají. Programovací prostředí NXT-G umožňuje využívat tři datové typy. Jedná se o Number pro celá čísla, Text pro řetězce znaků a Logic pro logické hodnoty. Oproti tomu RobotC má možnosti širší. Umožňuje navíc pracovat např. s reálnými čísly nebo dokonce s polem.

Obě prostředí podporují přetypování proměnných v programu. V NXT-G se tato činnost provádí při převodu čísla na textovou podobu blokem Number to Text. V RobotC je možné přetypovat hodnoty číselného datového typu dvěma způsoby. První je implicitní přetypování, při kterém nezadáváme cílový datový typ a nedochází při něm ke ztrátě dat. Druhý je explicitní, při kterém je nutné určit cílový datový typ. Během tohoto převodu dojde k částečné ztrátě dat.

Logické operace				
Sledovaná kritéria			***NXT-G***	***RobotC***
Logické operace	Operátory	>	✓	✓
		<	✓	✓
		=	✓	✓
		>=	✘	✓
		<=	✘	✓
	Logické funkce	AND	✓	✓
		OR	✓	✓
		XOR	✓	✓
		NOT	✓	✓

Tabulka 4 - Posouzení kritérií týkajících se logických operací

U logických operací jsme nejprve ověřovali, zda můžeme v obou prostředích využívat logické operátory pro tvorbu logických výrazů. V NXT-G se pro jejich realizaci používá blok Compare. Ten umožňuje ověřovat, zda je zadané číslo nebo vstup bloku menší, větší nebo roven jinému číslu či vstupu. V jeho nastavení ale nenalezneme volbu pro použití operátoru větší rovno nebo menší rovno. Tyto dva operátory v NXT-G využít nemůžeme. Oproti tomu v RobotC nalezneme pro tvorbu logických výrazů všechny operátory. Dále jsme se zaměřili na logické funkce. Z pohledu jejich využití jsme mezi oběma prostředími nenalezli žádný rozdíl. V obou je možné využít čtyři základní logické funkce (AND, OR, XOR a NOT).

Matematické operace			
Sledovaná kritéria		***NXT-G***	***RobotC***
Základní matematické operace	Sčítání	✓	✓
	Odčítání	✓	✓
	Násobení	✓	✓
	Dělení	✓	✓
Pokročilé matematické funkce	Goniometrické funkce	✓ (nutnost doinstalování - pouze sinus a cosinus)	✓
	Cyklometrické funkce	✓ (nutnost doinstalování - pouze arcus tangens)	✓
	Mocniny	✘	✓
	Exponenciála se základem e	✘	✓
	Odmocniny	✓ (pouze druhá odmocnina)	✓ (n-tá odmocnina)
	Logaritmy	✘	✓
	Absolutní hodnoty	✓	✓
Náhodná čísla		✓	✓
Intervaly		✓	✓

Tabulka 5 - Posouzení kritérií týkajících se matematických operací

Matematické operace byly pro porovnání rozděleny do několika skupin. První z nich jsou základní operace jako sčítání, odčítání, násobení a dělení. V NXT-G se s nimi pracuje pomocí jediného bloku pro matematické operace nazvaného Math. V RobotC je provádíme zápisem s použitím patřičného symbolu (+, -, * nebo /). Rozdíly nalezneme

u možností výpočtů se složitějšími matematickými funkcemi. NXT-G obsahuje pouze bloky pro goniometrické (blok HTSinCos) a cyklometrické (blok HTATan2) funkce. Ty musíme navíc pro použití do prostředí doinstalovat, protože nejsou součástí základní instalace. Blok HTSinCos ale umožňuje využívat pouze funkce sinus a cosinus. Podobně je tomu i u bloku HTATan2, který obsahuje pouze funkci arcus tangens. Práce s odmocninami a absolutní hodnotou je možná pomocí bloku Math pro základní matematické operace. Můžeme ale použít pouze druhou odmocninu. Pokud bychom chtěli provádět další funkce, museli bychom tak učinit pomocí složitého propojení několika bloků, což by bylo poměrně náročné a k úspěšnému řešení bychom ani dojít nemuseli. V RobotC můžeme oproti NXT-G provádět všechny goniometrické i cyklometrické funkce. Stejně tak i provádět výpočty s mocninami nebo logaritmy. Použít lze také exponenciálu se základem e. Pro každou matematickou funkci existuje vždy příslušná programová funkce, která ji po zadání parametru vykoná.

Společnými operacemi pro obě prostředí jsou možnosti generování náhodných čísel v programu a vyjadřování intervalů. Rozdíl je opět pouze v použití (v NXT-G pomocí bloku, v RobotC zápisem pomocí matematických symbolů a funkcí).

Možnosti výstupní signalizace				
Sledovaná kritéria			***NXT-G***	***RobotC***
Výstup na displej	Obrázky		✓	✓
	Úsečka		✓	✓
	Text	Základní velikost	✓	✓
		Velký text	✘	✓
	Bod displeje		✓	✓
	Geometrické tvary	S výplní	✘	✓ (kruh, elipsa, čtverec nebo obdélník)
		Bez výplně	✓ (pouze kruh)	✓ (kruh, elipsa, čtverec nebo obdélník)
	Mazání displeje	Celý	✓	✓
		Řádek	✘	✓
		Bod	✘	✓
Zvukový výstup	Tón		✓	✓
	Zvukový soubor		✓	✓
Světelná signalizace (světelné kostky)			✓ (s redukcí RCX)	✓ (s redukcí RCX)

Tabulka 6 - Posouzení kritérií týkajících se možností výstupní signalizace

Ve skupině kritérií zaměřené na možnosti výstupní signalizace jsme se jako první věnovali možnostem výstupu na displej. Pro veškeré možnosti vykreslování na displej nebo jeho mazání se v NXT-G používá blok Display. V RobotC jsou úkony realizovány speciálními funkcemi a příkazy. Obě prostředí shodně umožňují zobrazovat na displeji obrázky ve formátu .ric, text nebo úsečky definované dvěma body Kartézské soustavy souřadnic. V NXT-G nemůžeme volit větší velikost textu, než je základní, zabírající jeden řádek

displeje. U RobotC je k dispozici ještě možnost výpisu větší velikosti textu. U obou prostředí můžeme na displeji vykreslovat pouze jediný jeho bod definovaný přesnými souřadnicemi. Rozdíly nalezneme u vykreslování geometrických tvarů. RobotC obsahuje funkce, díky kterým můžeme vykreslovat tvary (kruh, elipsa, čtverec nebo obdélník) s výplní. U NXT-G tuto možnost nenalezneme. V NXT-G navíc můžeme vykreslit pouze kruh.

Podstatný rozdíl registrujeme u možností mazání plochy displeje. V NXT-G je možné mazat pouze celý displej. RobotC obsahuje funkce pro mazání jeho jednotlivých řádků nebo dokonce zvolených bodů displeje, což umožňuje vykreslení složitějších celků na jeho plochu.

Při práci se zvukovým výstupem řídící jednotky můžeme v obou prostředích využít k přehrání zvukové tóny nebo soubory ve formátu .rso. Jediný rozdíl, který u práce se zvuky nalezneme, je možnost nastavení frekvence v hertzích v RobotC. NXT-G tuto volbu neumožňuje.

Poslední možností vnější signalizace jsou světelné kostky převzaté ze starší stavebnice RCX. V NXT-G se pro jejich ovládání používá programový blok Lamp. V RobotC nenalezneme žádnou speciální funkci pro jejich řízení. Ovládají se totožnými funkcemi jako servomotory, což může být v programu, ve kterém použijeme oba tyto moduly, poněkud zavádějící.

Práce s moduly			
Sledovaná kritéria		***NXT-G***	***RobotC***
Nutnost deklarace modulu		✗ není potřeba	✓ v horním menu
Kalibrace modulu		✓	✓
Použití modulu		pomocí programového bloku	pomocí příkazů a funkcí
Dostupnost ovladačů rozšiřujících modulů		na stránkách výrobce modulu	většina přiložena k programovacímu prostředí
Multiplexování	Vstupů	✓	✓
	Výstupů	✓	✓

Tabulka 7 - Posouzení kritérií týkajících se práce s moduly

U práce s moduly jsme sledovali, jakým procesem si musí uživatel při jejich použití v programu projít. Prvním krokem byla deklarace před započetím programování. Ta se provádí pouze v programovacím prostředí RobotC. Provedeme ji v jeho horním menu. V NXT-G nic takového absolvovat nemusíme, protože konfigurační údaje jsou obsaženy v nastavení příslušného programového bloku modulu. Dále jsme se zaměřili na kalibraci senzoru. Ta se provádí kvůli zajištění přesného snímání hodnot. Její umožnění je důležité a je možné ji provádět v obou prostředích. V NXT-G se provádí pomocí funkce umístěné v horním menu. V RobotC je prováděna často za pomoci speciálně určené funkce pro konkrétní senzor. Ovládání modulu se dále vykonává způsobem charakteristickým programovacímu prostředí. V NXT-G pomocí programového bloku a v RobotC prostřednictvím příkazů a funkcí.

Základní instalace obou programovacích prostředí umožňuje pracovat se základními typy modulů robotické stavebnice. Při zakoupení rozšiřujících modulů může dojít k tomu, že nebudeme mít k dispozici potřebné ovladače. Bloky pro ovládání modulů v NXT-G získáme ve většině případů přímo na stránkách výrobce daného modulu. Ovladače v podobě knihoven funkcí pro RobotC, jsou přiloženy k instalaci programovacího

prostředí. Pokud by k dispozici nebyly, dají se získat na oficiálních stránkách, které se věnují jeho podpoře.

Jelikož v rozsáhlejších modelech robota se používá velké množství modulů, které ovlivňují jeho funkci a nemusí pro ně být vždy k dispozici na řídící jednotce dostatečný počet volných vstupních nebo výstupních portů, zaměřili jsme se také na to, zda obě programovací prostředí podporují multiplexování. Výsledkem bylo zjištění, že jej obě podporují. Rozdíl je ale v realizaci. U multiplexování výstupů musíme v NXT-G používat pro řízení modulů bloky, které multiplexování podporují a stáhnout si je tedy na stránkách výrobce. V RobotC je nutné použít speciální knihovnu funkcí určenou pro multiplexer, která obsahuje funkce sloužící k jeho propojení s řídící jednotkou a modulem. U multiplexování vstupů záleží na použitém zařízení. Některé potřebuje ke správnému fungování speciální bloky nebo knihovny, jiné nikoliv.

Vlastní metody a paralelní programování			
Sledovaná kritéria		***NXT-G***	***RobotC***
Vytváření vlastních metod		✓	✓
Paralelní programování	Počet paralelně spuštěných úloh	maximálně 3	maximálně 10
	Možnost nastavení priorit spuštění	✘	✓
	Možnost spuštění či zastavení na libovolném místě	✘	✓
Pokročilejší prvky paralelního programování	Semafor	✘	✓

Tabulka 8 - Posouzení kritérií týkajících se dalších možností programování

Rozšiřující možnosti programování zahrnují dvě sledovaná kritéria. Jako první jsme zjišťovali, zda programovací prostředí umožňují vytváření vlastních metod. V NXT-G je tato funkce realizována jako vytvoření vlastního programového bloku, který obsahuje část zdrojového kódu v podobě několika bloků. Funkce je umístěna v horním menu a realizována jednoduchým průvodcem. V RobotC máme dvě možnosti vytvoření. První je vlastní funkce s návratovou nebo bez návratové hodnoty a druhá dotaz task. Vytvoření

vlastní funkce v RobotC je srovnatelné s vytvořením vlastního bloku v NXT-G. Dotaz task je speciální možnost. Obsahuje část programového kódu, jehož vykonání je voláno na libovolném místě programu. Dotaz můžeme spouštět a zastavovat na libovolném místě. Pokud použijeme dotazů více, můžeme u nich nastavovat priority jejich spuštění. Vytváření vlastních funkcí a dotazů je způsob, kterým je v RobotC realizováno paralelní programování. V NXT-G je paralelní chod programu vytvořen pomocí jednoho ze tří vláken, která jdou rozvinout v počátečním bodě vytvářeného programu. Po spuštění jsou následně vykonávána nezávisle na sobě. Přiřazování priorit vykonávání nebo řízení spuštění paralelní části programu v NXT-G možné není.

U paralelního programování jsme ještě ověřovali, zda nemají obě programovací prostředí nějakou rozšiřující funkci. Zjistili jsme, že v RobotC je oproti NXT-G možné využívat semafor. Ten v programování slouží k řízení přístupu do sdílené paměti nebo plánovače úloh.

Doplňkové funkce prostředí			
Sledovaná kritéria		***NXT-G***	***RobotC***
Zobrazení a vizualizace dat	Zjišťování v reálném čase	✓	✓
	Zaznamenávání	✓	✘
	Grafické zobrazení	✓	✘
Správa paměti		✓	✓
Vkládání komentářů		✓	✓
Vytváření uživatelských profilů		✓	✘
Upozornění na chyby		změnou vzhledu datového vodiče	při kompilaci programu
Orientace v rozsáhlém programu		pomocí částečného náhledu	pomocí posuvníku

Tabulka 9 - Posouzení kritérií o doplňkových funkcích programovacích prostředí

Poslední skupina kritérií porovnává nadstandardní funkce obou prostředí. První z nich je možnost zobrazování a zaznamenávání hodnot ze vstupních nebo výstupních modulů.

NXT-G obsahuje funkci Data Logging, díky které můžeme data nejen snímat, ale také zaznamenávat a zobrazovat do grafu. RobotC sice obsahuje funkci Poll NXT Brick, ale ta umožňuje pouze zjišťování dat v reálném čase. Záznam dat v RobotC bychom tak museli provést programově. Zaznamenání nebo vizualizaci dat graficky zde neprovedeme. Naopak obě prostředí mají vestavěnou funkci, díky které můžeme spravovat paměť řídící jednotky NXT. Umožňuje přehled o programech uložených v paměti a jejich správu.

Pro lepší orientaci v programu je důležité mít možnost vkládat do programového kódu komentáře. Zjistili jsme, že obě programovací prostředí to umožňují. Jelikož je ale NXT-G grafické programovací prostředí, vkládá se komentář na programovací plochu jako textové pole. Nevýhodou toho je, že při manipulaci s programovými bloky zůstává komentář na stejném místě a může tak dojít k nepřesnému umístění komentáře v programu. V RobotC se komentář vkládá za dvě lomítka na potřebné místo. K něčemu podobnému jako u NXT-G tak dojít nemůže.

Odlišností oproti RobotC je možnost vytvářet v NXT-G uživatelské profily. RobotC žádnou podobnou funkci nemá.

Zkoumali jsme také, jak je uživatel informován o tom, že v programu udělal chybu. V NXT-G je upozorněn pouze v případě, že při propojení dvou bloků spojil ty, které nejsou stejného datového typu a propojení tak není možné. Datový vodič v tomto případě okamžitě změní svoji podobu na přerušovanou. U propojování bloků ale nalezneme více problémů. Můžeme se setkat se situací, kdy vypadá, že jsou dva bloky propojené, ale vodič ve skutečnosti není připojen k danému konektoru bloku. V jiném případě může vodič směřovat k některému z konektorů, ale ve skutečnosti bude připojen k jinému. V NXT-G je kompilace prováděna při nahrání programu do řídící jednotky. V RobotC je možné ji provádět nezávisle na nahrání a to nám umožňuje odhalovat chybný zápis programového kódu.

Posledním zkoumaným kritériem této sekce byly možnosti orientace v rozsáhlejším programu. NXT-G obsahuje pouze malý náhled umístěný vpravo dole. Na něm vidíme zvýrazněnou část programu, která je aktuálně zobrazena v okně. Neumožňuje ale oddálit vytvářený program, abychom viděli větší část. Orientace v rozsáhlém programu je tak značně složitá a dohledávání případné chyby obtížné a zdlouhavé. Oproti tomu orientace

v programu v RobotC se nijak neliší od jiných textových programovacích prostředí. Pohyb je vykonáván pomocí postranního posuvníku. Konkrétní část nebo funkci programu můžeme také dohledat pomocí funkce najít, která se spustí po stisku klávesové zkratky Ctrl+F.

2.3.3 ZÁVĚREČNÉ SHRNUTÍ

Na závěr se pokusíme shrnout největší klady a zápory obou programovacích prostředí. Největší výhodou NXT-G je jeho jednoduchost. Uživatel nemusí provádět žádné složité nastavování nebo deklarace. Jedinou takovou činností je import nových programových bloků. Nepříjemností může být zdlouhavější a ne na první pohled zřejmá deklarace proměnných. Z pohledu nižšího počtu datových typů nevyvstává příliš velký problém. Uživatel nemusí přemýšlet nad tím, který konkrétní datový typ zvolit, rozlišuje pouze, o jaký typ dat se jedná. U žáků mladšího školního věku není nutné rozlišovat několik druhů číselných datových typů, tak jako je tomu u RobotC.

Práce s rozšiřujícími moduly se u NXT-G také jeví snazší než v RobotC. Důvod je následující. Po importu programového bloku do prostředí a vložení na programovací plochu, uživatel okamžitě vidí, jaké možnosti nastavení má u modulu k dispozici. V RobotC musí jako první zavolat příslušnou knihovnu funkcí. Bez toho aniž by prozkoumal její zdrojový kód, ovšem nezjistí, jaké funkce může k ovládání modulu použít. Musí ze zdrojového kódu souboru proto vyčíst, jak se funkce správně zapisuje a jaké jsou její parametry.

Podíváme-li se ale také na nevýhody NXT-G, tak tou nejvýznamnější bude pro nezkoušeného uživatele poměrně obtížná orientace v rozsáhlém programu a to hned ze dvou důvodů. Jeden je nedostatečný náhled v hotovém programu a s tím související chybějící zoom a druhý nepříliš přehledné propojování programových bloků.

Programovací prostředí RobotC obsahuje několik pokročilejších možností než NXT-G. Z porovnání matematických operací, které lze v programovacích prostředích provádět, vyplývá, že RobotC nabízí uživateli širší možnosti výpočtů. Používá také mnohem více číselných datových typů a umožňuje práci s polem. Nalezli jsme také několik drobných odlišností, které ale nejsou pro porovnání příliš zásadní. Jako nevýhodu můžeme

jmenovat chybějící možnost zaznamenávání dat, jako je tomu u funkce Data Logging v NXT-G.

Na základě srovnání a vyjmenovaných kladů a záporů obou programovacích prostředí můžeme říci, že NXT-G je vhodnější pro výuku začátečníků, kteří mají s robotickou stavebnicí nebo s programováním obecně buďto velmi málo nebo vůbec žádné zkušenosti. Z našeho pohledu by tak jeho zařazení do výuky bylo vhodné na druhém stupni základní školy. Nahrává tomu jeho nepříliš velká náročnost na znalosti uživatele. Naopak RobotC je vhodnější pro pokročilejší programátory, kteří již dokážou pracovat s textově zapisovaným zdrojovým kódem a bez problémů se v něm orientují. Pro vytváření složitějších programů obsahuje RobotC navíc několik rozšiřujících možností. Jeho zařazení do výuky je tak vhodnější realizovat spíše v pozdějších stupních vzdělávání.

3 MULTIMEDIÁLNÍ VÝUKOVÝ MATERIÁL

Na Katedře výpočetní a didaktické techniky Pedagogické fakulty Západočeské univerzity v Plzni vznikly v roce 2010 webové stránky https://www.lego.zcu.cz. Web si klade za cíl poskytnout návštěvníkům co nejvíce užitečných informací o možnostech využití robotické stavebnice ve vyučování. Zaměřuje se především na LEGO Mindstorms NXT 2.0. Nalezneme zde informace o základních i rozšiřujících modulech stavebnice nebo programovacích prostředích. Web obsahuje také množství vzorových programových úloh. Jedním z úkolů práce bylo vytvořit multimediální výukový materiál, který vznikl v rámci této webové stránky.

3.1 CÍL TVORBY VÝUKOVÉHO MATERIÁLU

Před započetím práce bylo nutné si stanovit, co by mělo být výsledkem tvorby. Jelikož v českém prostředí chybí dostupné materiály věnující se programování robotické stavebnice, vytkli jsme si za cíl vytvořit na webových stránkách kurz, který bude představovat jednotlivé moduly stavebnice LEGO Mindstorms NXT a vybraných programových konstruktů a jejich použití bude implementovat do programovacích prostředí NXT-G a RobotC. Kurz by měl sloužit jako podpůrný materiál pro programování robotické stavebnice. Obsah výukového kurzu se odvíjí od osnovy, která byla pro jeho vypracování předložena vedoucím diplomové práce. Při vytváření kurzu byla podle potřeby upravována a některé články doplněny. Osnovu naleznete k nahlédnutí v příloze (viz. Příloha 1).

3.2 VÝCHODISKA TVORBY VÝUKOVÉHO MATERIÁLU

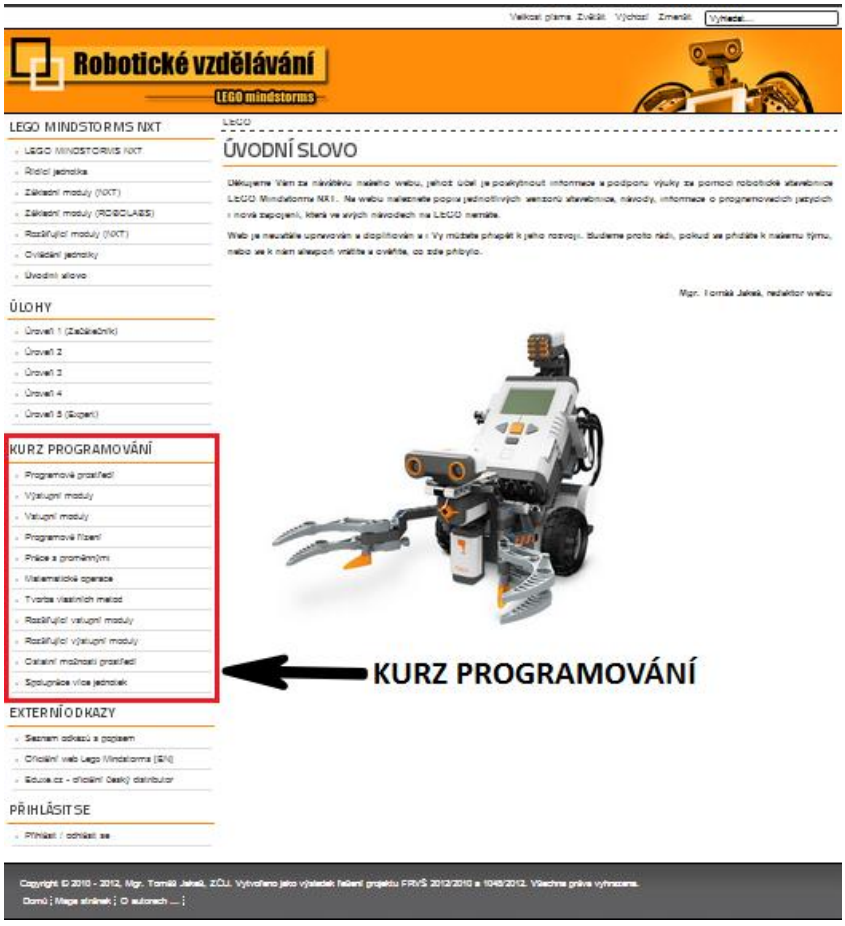
Při vytváření výukového materiálu jsme si stanovili také několik východisek týkajících se jeho obsahu:

- představit programovací prostředí NXT-G a RobotC a jejich funkce,
- představit možnosti práce s jednotlivými programovými konstrukty v NXT-G a RobotC,
- poukázat na rozdíly mezi ikonickým a textovým robotickým programovacím prostředím,
- představit možnosti využití základních i vybraných rozšiřujících modulů v obou programovacích prostředích,

- demonstrovat na příkladech praktické využití jednotlivých modulů,
- doplnit vhodně výukový materiál o videa či animace,
- doplnit materiál vhodnými grafickými prvky.

3.3 POPIS VÝUKOVÉHO MATERIÁLU

Koncept webové stránky je rozdělen do několika částí. První se věnuje obecnému představení robotické stavebnice LEGO Mindstorms NXT, možnostem jejího programování a představení funkcí. Další sekce obsahuje vzorové programové úlohy, které demonstrují funkce některých modulů. Pod tuto sekci v menu v levé části stránky byl umístěn výukový materiál. Jeho umístění v rámci úvodní stránky webu můžete vidět na následujícím obrázku.



Obrázek 24 - Umístění výukového materiálu na webu https://www.lego.zcu.cz

Na základě osnovy jsou články kurzu rozděleny do několika následujících kapitol:

Programovací prostředí a jeho ovládání: Obsahuje články věnující se popisu obou programovacích prostředí a úvodním krokům při vytváření programu v nich, od založení

nového programu až po jeho nahrání do řídící jednotky NXT. Zahrnuje také informace o aktualizaci firmware a správě paměti robotické stavebnice.

Výstupní moduly: Kapitola se věnuje základním výstupním modulům robotické stavebnice. Obsahuje informace o využití jednoho či více servomotorů, možnosti využití displeje, zvukového výstupu a světelných kostek.

Vstupní moduly: Kapitola o čtyřech základních vstupních senzorech a jejich možnostech praktického použití. Jedná se o ultrazvukový, zvukový, světelný a dotykový senzor.

Programové řízení: Zahrnuje informace o možnostech řízení programu pomocí cyklů a podmínek. Jmenuje a představuje všechny jejich druhy a jejich charakteristické použití vysvětluje na příkladech.

Práce s proměnnými: Představuje postup práce a možnosti využití proměnných a konstant v obou programovacích prostředích od jejich deklarace, až po čtení a zápis. Věnuje se také datovým typům, kterých mohou proměnné nabývat.

Matematické operace: Popisuje dostupné logické a matematické operace v NXT-G a RobotC. Představuje způsob jejich zápisu a praktické využití.

Tvorba vlastních metod: Kapitola se zaměřuje na možnosti vytváření vlastních metod v obou prostředích.

Rozšiřující vstupní moduly: Představuje rozšiřující vstupní senzory od dalších výrobců, jejich použití a praktické využití. V kapitole jsou zahrnuty také články o možnostech multiplexování vstupních portů.

Rozšiřující výstupní moduly: Kapitola popisuje výstupní zařízení od jiných výrobců než je společnost LEGO. Doplněny jsou také informace o možnostech multiplexování výstupních portů.

Ostatní možnosti prostředí: V této kapitole jsou popsány další funkce obou prostředí, jako např. zaznamenávání a vizualizace hodnot. Nalezneme zde ale také popsané úkony, které jsou charakteristické pro některé ze dvou programovacích prostředí. Jedná se hlavně o úvodní deklaraci modulů v RobotC nebo instalaci nových bloků v NXT-G.

Spolupráce více jednotek: Poslední kapitola se věnuje možnostem komunikace a spolupráce více jednotek pomocí vestavěného modulu Bluetooth.

Struktura článků výukového materiálu je rozdělena do dvou částí. V první části se návštěvník u každého programového konstruktu nebo modulu dozví jeho základní popis, který je následovaný popisem možností práce s ním v obou programovacích prostředích. U NXT-G nalezneme podrobný popis práce s programovým blokem, u RobotC nejpoužívanější příkazy a funkce pro danou problematiku, případně některé další potřebné doplňující informace. Druhá část se věnuje praktickému využití. Na jednoduchých příkladech, nebo pouze částech rozsáhlejších programů, je zde představena práce s daným konstruktem či modulem. Široce je praktické využití rozvedeno hlavně u vstupních a výstupních zařízení. Veškerý popis je doplněn o obrázky a grafiku. V textu byly použity originální ikony, které nalezneme v programovacím prostředí. Jednotlivé příkazy nebo části kódu v RobotC jsou upraveny tak, aby je bylo možné z webu snadno zkopírovat. Komplexní úlohy v praktické části si je možné stáhnout ve formátech obou programovacích prostředí. Příklady jsou navíc doplněny několika videi, která tématiku dokreslují. Pro případ, že by uživatel potřeboval nastudovat informace související s aktuálním článkem, jsou články na některých místech propojeny odkazy.

3.3.1 POUŽITÉ VÝVOJOVÉ PROSTŘEDÍ

Správa webové stránky je zajištěna pomocí redakčního systému Joomla! ve verzi 2.5. Ten nabízí při tvorbě webu mnoho funkcí. Umožňuje stránky spravovat z jakéhokoliv počítače díky možnosti správy pomocí webového rozhraní. Obsahuje hierarchickou správu skupin uživatelů. Každá skupina má nastavena specifická uživatelská práva, a je tak ošetřeno, jak zásadně se může podílet na tvorbě a správě webové stránky. Redakční systém obsahuje také vnitřní správu veškerého obsahu, včetně souborů, obrázků a multimédií. V případě, že se na tvorbě příspěvků pro webovou stránku podílí více přispěvatelů, je možné jejich články spravovat, upravovat, přesouvat či mazat. Pro vytváření příspěvků je k dispozici vestavěný WYSIWYG editor se základními funkcemi pro práci s textem. K dispozici je ale mnoho dalších funkcí, jako propojení s FTP serverem, emailové služby nebo plná podpora kanálu RSS. Značnou výhodu dávají tvůrci webu v redakčním systému moduly a pluginy, díky kterým může web různě modifikovat. Jedná se o specializované nástroje, které slouží

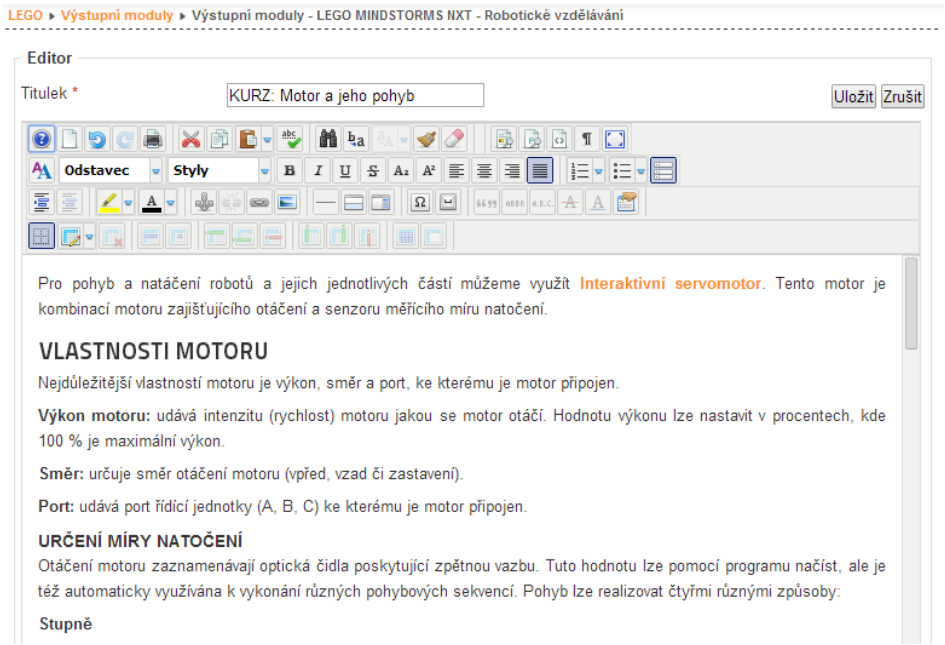
k vytvoření nebo obohacení určité části webu. Realizují se s jejich pomocí například různé galerie, možnosti správy souborů, přehrávání videí na webu a podobně.

3.3.2 POUŽITÁ ROZŠÍŘENÍ A KOMPONENTY

Při práci na výukovém materiálu bylo použito několik modulů a pluginů, které umožnily používat rozšířené funkce pro práci s webem. Doplnily také finální vzhled článků a umožnily používání některých elementů pro práci s textem nebo multimediálními prvky.

WYSIWYG editor textu

Vytváření jednotlivých článků výukového materiálu bylo prováděno pomocí vestavěného WYSIWYG editoru. Na webu jsme použili editor **JCE**. Jeho základní podoba obsahuje nástroje pro práci s textem. Pomocí dalších pluginů do něj lze ale doplnit mnohé rozšířené funkce, např. funkci pro zobrazování obrázků a videí pomocí komponenty Lightbox. Obsahuje také volbu pro plynulé přepínání do režimu zdrojového kódu, díky čemuž jsme mohli článek v případě potřeby editovat v jazyce XHTML. Při práci na kurzu jsme využívali hlavně nástroje pro základní formátování textu. Pro jednotný vzhled jednotlivých článků kurzu byly definovány různé styly písma. Při vytváření úvodních rozcestníků kapitol jsme vkládali na potřebná místa kotvy, na které byly následně směrovány odkazy.

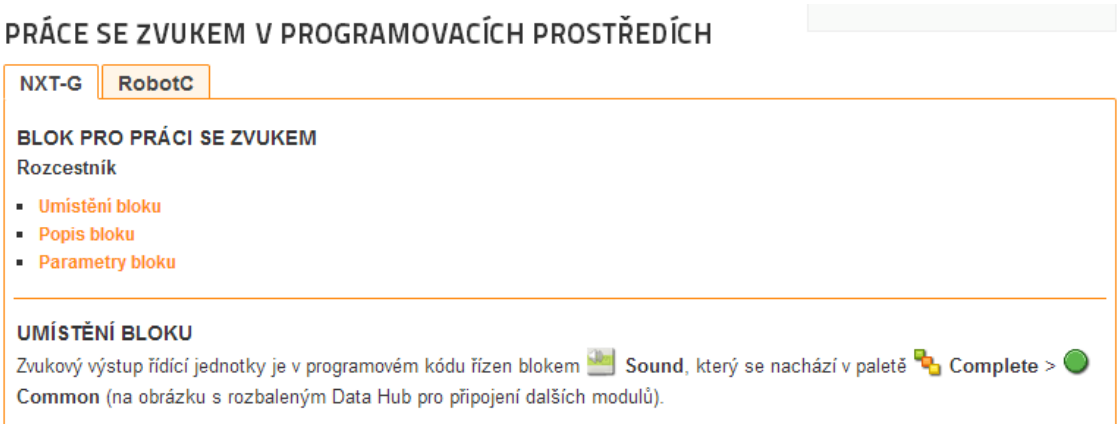


Obrázek 25 - WYSIWYG editor JCE

Web rozšíření: http://extensions.joomla.org/extensions/edition/editors/88

Rozdělení článku do tabulek

Každý článek se věnuje obecnému popisu modulu či programového konstruktu, následně popisuje jeho využití v NXT-G a v RobotC a na závěr ještě představuje praktické možnosti jeho použití. Články jsou tedy poměrně obsáhlé. Řešili jsme proto, jak je vzhledem k jejich délce zpřehlednit. Použili jsme plugin **Tabs & Sliders**, který umožňuje vytvořit tabulku se záložkami, která se podobá kartotéce. V horní části textu, který obsahuje, se zobrazí záložky, které představují jednotlivé oddíly článku. Tímto způsobem jsme docílili toho, že text věnující se stejné problematice, pouze aplikované do jednoho a posléze druhého programovacího prostředí, je umístěn na stejném místě a uživatel se pouze přepíná mezi jednotlivými záložkami.

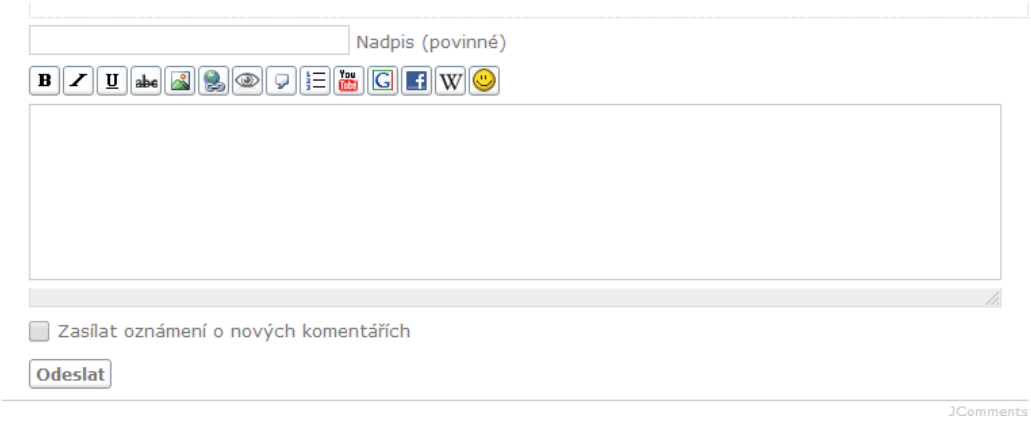


Obrázek 26 - Záložky vytvořené pomocí pluginu Tabs & Sliders

Web rozšíření: http://extensions.joomla.org/extensions/news-display/article-elements/articles-tabs/1046

Komentáře k článku

Při vytváření kurzu jsme si vytyčili, že bude dobré, abychom získávali zpětnou vazbu od návštěvníků webu. Pod každý článek jsme proto pomocí pluginu **JComments** vložili možnost vkládání komentářů. Tento plugin má několik funkcí. Umožňuje pracovat s jednotlivými uživatelskými oprávněními, díky čemuž jsme přidávání komentářů umožnili pouze registrovaným a přihlášeným uživatelům. Při odesílání komentáře si můžeme vyžádat zadání CAPTCHA kódu, jako ochranu proti spamu. Při vytváření komentáře se dají vkládat obrázky, odkazy, videa nebo citace. V nastavení nalezneme také volbu pro odeslání upozornění administrátorovi webu po každém přidání komentáře.

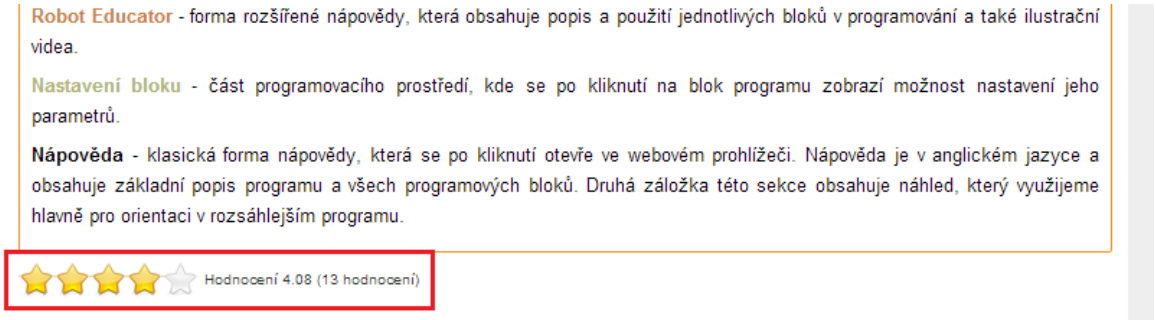


Obrázek 27 - Okno pro přidání komentáře k článku vytvořené pluginem JComments

Web rozšíření: http://extensions.joomla.org/extensions/contacts-and-feedback/articles-comments/9985

Hodnocení článků

Vkládání komentářů je sice nejlepší způsob zpětné vazby, ale ne každý uživatel bude ochoten jej vkládat. Pro rychlý přehled o kvalitě článku a jeho užitečnosti pro návštěvníky webu, bylo vloženo pod články ještě pětihvězdičkové hodnocení. Realizováno je pluginem **Extra Vote**. Ten má jednoduchou funkci. Pomocí pěti hvězdiček zobrazuje, jaké je průměrné hodnocení článku uživateli. Před hvězdičkami se ještě zobrazuje, kolik návštěvníků o kvalitě článku hlasovalo. Hlasování se nemusí používat pouze na konci článku, ale takřka na kterémkoliv místě webu.



Obrázek 28 - Zvýrazněné hodnocení vložené pod článek pomocí pluginu Extra Vote

Web rozšíření: http://extensions.joomla.org/extensions/clients-a-communities/ratings-a-reviews/5483

Vkládání zdrojového kódu do článku

U popisu programování v prostředí NXT-G jsme demonstrovali jednotlivé konstrukty pomocí pořizování screenshotů programových bloků nebo částí kódu. Pokud bychom to takto řešili i u RobotC, narazili bychom na několik problémů. Pro každou prezentovanou část kódu bychom museli vytvářet vlastní screenshot. Problém by navíc vyvstal u programů, které by nešly díky svojí délce celé zobrazit na monitoru pro pořízení jediného obrázku. Demonstrace by se tak musela skládat z několika obrázků. Při opravě chyb nebo editaci programu by bylo nutné vytvářet znovu kompletní obrázek a umisťovat jej na web. Návštěvníci webu by navíc v případě, že by si chtěli program sami vyzkoušet, museli zdrojový kód opisovat. Použili jsme proto plugin **GeSHi (Generic Syntax Higlighter)**. GeSHi slouží ke vkládání, formátování a následnému zobrazení zdrojového kódu na webové stránce. Vytvořením databáze příkazů či klíčových slov a definováním, jak mají být formátována, můžeme docílit totožného zobrazení, jako je tomu v programovacím prostředí. Díky tomu je možné celý zdrojový kód nebo pouze jeho část ze stránky jednoduše zkopírovat, bez nutnosti jej do programovacího prostředí opisovat. Jednoduchá je také jeho editace ze strany tvůrce, která je možná přímo ve webovém rozhraní administračního prostředí redakčního systému.

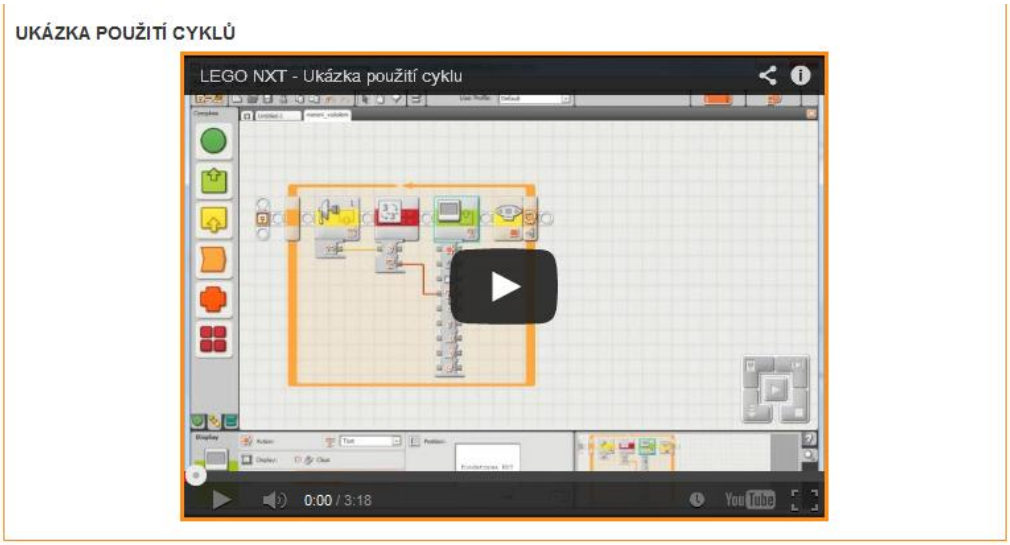
```
task main()
{
 nMotorEncoder[motorB] = 0; // Nulovani encoderu motoru B.
 nMotorEncoder[motorC] = 0; // Nulovani encoderu motoru C.

 nMotorEncoderTarget[motorB] = 180; // Nastaveni cilove pozice motoru B (otoceni o 180
stupnu).
 nMotorEncoderTarget[motorC] = 180; // Nastaveni cilove pozice motoru C (otoceni o 180
stupnu).

 motor[motorB] = 30; // Nastaveni rychlosti otaceni motoru B na 30%.
 motor[motorC] = 30; // Nastaveni rychlosti otaceni motoru C na 30%.

while (nMotorRunState[motorB] != runStateIdle || nMotorRunState[motorC] != runStateIdle)
 {
   // Tento cyklus ceka na to, az je dokonceno natoceni encoderu o pozadovany uhel.
   // Neprovadi se zadny dalsi prikaz.
 }

 motor[motorB] = 0; // Uvedeni motoru B do klidu.
 motor[motorC] = 0; // Uvedeni motoru C do klidu.
}
```

Obrázek 29 - Zdrojový kód zobrazený na webu díky pluginu GeSHi

Web rozšíření: http://qbnz.com/highlighter/

Vkládání videí

Na webu jsme použili některá videa uložená na serveru YouTube. Jejich vkládání na web jsme v redakčním systému realizovali pomocí pluginu **AllVideos**. Ten umožňuje vkládat videa nejenom ze serveru YouTube ale i například z Vimeo, Dailymotion, SoundCloud a dalších.



Obrázek 30 - Video vložené na web pomocí pluginu AllVideos

Web rozšíření: http://extensions.joomla.org/extensions/multimedia/multimedia-players/video-players-a-gallery/812

3.4 VYUŽITÍ VÝUKOVÉHO MATERIÁLU

Výukový materiál se zaměřuje na představení programovacích prostředí, programových konstruktů a modulů robotické stavebnice. Jeho struktura není koncipována jako výuka robotického programování od nejjednodušších úloh po složité. Nehodí se tedy příliš pro to, aby byl použit ve vyučování jako učební pomůcka pro samostudium nebo jiný způsob komplexního vzdělávání. Jeho využitelnost tkví v podpoře výuky. Hlavně v počátcích výuky robotického programování žáci a studenti naráží na mnohé problémy. Ty souvisí s počáteční neznalostí prostředí, programovacího jazyka, ale i samotné robotické stavebnice. Pro vyučujícího je při individualizovaném přístupu k žákům velmi obtížné reagovat na časté dotazy a řešit problémy. V těchto situacích může učitel žáky odkazovat k využití webových stránek a hlavně výukového materiálu. Před započetím programování si v něm mohou přečíst informace o základních funkcích některého z popisovaných

programovacích prostředí. Při řešení problému u vytváření programové úlohy si zde žáci mohou nejdříve přečíst informace o modulu stavebnice a následně si v kurzu dohledat, jakým způsobem by bylo nejvhodnější úlohu řešit a jaké programové konstrukty k řešení použít. V možnostech praktického využití navíc naleznou vzorovou úlohu, na níž si představí, jakým způsobem s modulem pracovat. V případě potřeby si mohou navíc programový kód zkopírovat, či kompletní program stáhnout v příslušném formátu programovacího prostředí. Vyučující tak v tomto materiálu získávají online pomůcku k podpoře výuky robotického programování a žáci studijní materiál, který by jim měl umožnit překonávat nastalé problémy. Jeho velkou výhodou pro žáky a mnohé učitele bude fakt, že materiál je v českém jazyce. Podpora robotického programování v českém jazyce totiž není příliš velká.

ZÁVĚR

Popularizace robotických stavebnic v posledních letech prudce narůstá. Dostávají se do povědomí mnoha uživatelů a značné množství škol je zařazuje do své výuky. Přesto, že jsou jejich možnosti velice široké, vyučující o jejich využívání nemají příliš mnoho informací. Cílem této práce tedy bylo představit možnosti jejího využití na různých stupních škol a porovnat možnosti jejího programování v programovacích prostředích NXT-G a RobotC.

Nejprve jsme stručně představili robotickou stavebnici, historii jejího vývoje a jednotlivé součásti. Následně jsme se zaměřili na možnosti využití robotické stavebnice LEGO Mindstorms NXT na různých stupních škol. Snažili jsme se vycházet ze senzorických a motorických schopností žáků v daném věku a z úrovně rozvinutí jejich myšlení. Doporučili jsme vhodný postup začlenění robotické stavebnice do výuky na různých stupních vzdělávání a také jsme popsali, v jakých předmětech by se dala využít.

Dále jsme se již zaměřili na samotná programovací prostředí NXT-G a RobotC. Obě prostředí jsme popsali, vysvětlili jejich funkce a způsob programování. Na představení následně navázalo jejich srovnání. Nejprve jsme si stanovili příslušná kritéria, která jsme popsali. Na jejich základě jsme sestavili přehledové tabulky. Výsledky v nich znázorněné jsme doplnili slovním komentářem.

V rámci diplomové práce vznikl také výukový materiál představující možnosti práce s jednotlivými moduly stavebnice či programovými konstrukty v programovacích prostředích NXT-G a RobotC. Jeho popisu se věnuje poslední kapitola. V té jsme shrnuli cíle, které jsme si pro jeho tvorbu vytyčili. Dále jsme popsali jeho postup tvorby a části, ze kterých se skládá.

Z výsledků srovnání NXT-G a RobotC vyplynulo, že každé z obou prostředí se hodí pro nasazení ve výuce jiné věkové skupiny uživatelů. Ikonické prostředí NXT-G obsahuje základní funkce a jeho ovládání je pro začátečníky jednodušší. Hodí se proto spíše pro výuku žáků mladšího školního věku. Pokročilejším programátorům už by mohly při jeho používání chybět některé pokročilé funkce. Ty obsahuje programovací prostředí RobotC. Jeho nasazení do výuky je proto vhodnější v dalších letech vzdělávání nebo pro zkušenější uživatele, kterým již zápis programového kódu v textové podobě nebude činit potíže.

RESUMÉ

The aim of this thesis is to introduce the possibility of using the robotic kit LEGO Mindstorms NXT at all the education levels and then to compare the possibilities of the robotic programming environments NXT-G and RobotC. One of the main parts of the thesis is the educational course based on the submitted curriculum. The educational course introduces basic programming constructs of both environments and explains the use of robotic kit modules.

The first chapter concerns of the general performance of the robotic kit and the history of its development. Then it follows by the description of the usage possibility in various educational levels. The chapter focuses on the kinetic abilities, thinking and other dispositions of students. Based on these dispositions we will describe how and in what subjects should be used the robotic kit during the teaching at the particular level of education.

The second chapter presents the both, NXT-G and RobotC, programming environments. It deals with the way of creating the source code of the program. It describes their characteristics and focuses on the user´s support extent. Finally, it compares both environments based on the selected criteria.

The last chapter is devoted to the teaching material that originated from this work. You will find out which targets we tried to fulfill and what environment the course will occur in. Its creation is also described. In conclusion, there is the information how they could be used in the education.

SEZNAM LITERATURY

1. MIT Media Lab. In: *LEGO's Mindstorms* [online]. [cit. 2013-10-02]. Dostupné z: http://www.media.mit.edu/sponsorship/getting-value/collaborations/mindstorms
2. MINDELL, D. et al. MIT - Massachusetts Institute of Technology. In: *LEGO Mindstorms - The Structure of an Engineering (R)evolution* [online]. 15. 12. 2000 [cit. 2013-10-08]. Dostupné z: http://web.mit.edu/6.933/www/Fall2000/LegoMindstorms.pdf
3. WILLIAMS, R. Faculty of Environment and Technology - University of the West of England. In: *Lego LEGO RCX, Mindstorms & NQC* [online]. 13. 9. 2007 [cit. 2013-10-12]. Dostupné z: http://www.cems.uwe.ac.uk/~rwilliam/RCX/LEGO_mindstorms_RCX.pdf
4. FERRARI, M. G. FERRARI a R. HEMPEL. *Buildings Robots with LEGO Mindstorms* [The ULTIMATE Tool for Mindstorms Maniacs!] [online]. 2nd ed. London: International Thomson, 2002 [cit. 2013-12-16]. ISBN 1-928994-67-9. Dostupné z: http://wikirobokomp.ru/images/1/1e/Building_Robots_with_LEGO_MINDSTORMS_(2002).pdf
5. MAIRI. LEGO Mindstorms RCX. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikipedia Foundation, 14. 1. 2006 [cit. 2014-02-09]. Dostupné z: http://en.wikipedia.org/wiki/File:LegoMindstormsRCX.jpg
6. LEGO. In: *Historie LEGO Robotics - robotického programu* [online]. 2013 [cit. 2013-11-02]. Dostupné z: http://www.lego.com/cs-cz/mindstorms/gettingstarted/historypage/
7. LEGO. In: *Hardware Developer Kit* [online]. 2006 [cit. 2013-11-04]. Dostupné z: http://www.lego.com/en-us/mindstorms/downloads/nxt/nxt-hdk/
8. *Metodika NXT* [online]. The LEGO Group, 2006 [cit. 2013-11-25]. Dostupné z: http://www.eduxe.cz/domain/prezentace/files/download/9797_lme_manual_cz.pdf
9. SOLDAAT, X. Comparing the NXT and EV3 bricks. In: *BOT BENCH* [online]. 8. 1. 2013 [cit. 2014-01-10]. Dostupné z: http://botbench.com/blog/2013/01/08/comparing-the-nxt-and-ev3-bricks/
10. LEGO. In: *Uživatelská příručka pro EV3* [online]. 2014 [cit. 2014-01-10]. Dostupné z: http://www.lego.com/cs-cz/mindstorms/downloads/user-guides/cs/
11. SULÍR, M. *Úvod do programování robotů 1* [online].. Orlová: Obchodní akademie Orlová, 2012 [cit. 2013-11-22]. 978-80-87477-04-5. Dostupné z: http://scholanova.obaka-orlova.cz/files/robot1.pdf
12. HERETOHELP. Lego Mindstorms kit. In: *Wikimedia Commons* [online]. 10. 11. 2010, 15:00 [cit. 2014-02-15]. Dostupné z: http://commons.wikimedia.org/wiki/File:Lego_Mindstorms_kit.jpg
13. VYSKOTOVÁ, J. a K. MACHÁČKOVÁ. *Jemná motorika: vývoj, motorická kontrola, hodnocení a testování*.. Praha: Grada Publishing, a.s. 2013. ISBN 978-80-247-4698-2.
14. LANGMEIER, J. a D. KREJČÍŘOVÁ. *Vývojová psychologie*. 2. aktualiz. vyd. Praha: Grada,

2006. ISBN 80-247-1284-9.

15. VÁGNEROVÁ, M. *Vývojová psychologie: dětství, dospělost, stáří*.. Praha: Portál, s.r.o. 2000. ISBN 80-7178-308-0.
16. V čem se soutěží? In: *Česká liga robotiky* [online]. 2013 [cit. 2013-11-25]. Dostupné z: http://www.ceskaligarobotiky.cz/o-fll/v-cem-se-soutezi
17. CMU ROBOTICS ACADEMY. FTC Getting Started. In: *Carnegie Mellon Robotics Academy* [online]. 2009 [cit. 2014-01-15]. Dostupné z: http://www.education.rec.ri.cmu.edu/content/events/ftc/common/docs/install_nxtg_lv.pdf
18. KELLY, JAMES FLOYD. *LEGO Mindstorms NXT-G programming guide*. New York: Apress, 2010 [cit. 2013-11-10]. ISBN 978-1-4302-2977-3. Dostupné z: http://www.ncdd.com.br/robotica/Lego%20Mindstorms%20NXT-G%20programming%20guide.pdf
19. BFEHER. Getting Started. In: *ROBOTC a C Programming Language for Robotics* [online]. 7. 5. 2012, 18:32, verze 13 February 2014 [cit. 2014-01-23]. Dostupné z: http://www.robotc.net/wiki/Tutorials/Getting_Started
20. BFEHER. Data Types. In: *ROBOTC a C Programming Language for Robotics* [online]. 23. 1. 2012, 18:40, verze 20 June 2012 [cit. 2014-02-20]. Dostupné z: http://www.robotc.net/wiki/Data_Types

SEZNAM OBRÁZKŮ A TABULEK

PŘÍLOHY

Příloha 1

Struktura studijní kurzu robotického programování pomocí NXT-G a RobotC

1. Programovací prostředí a jeho ovládání
   - založení a uložení nového programu,
   - orientace v prostředí,
   - export programu do robotické stavebnice,
   - správa paměti robotické stavebnice.
2. Základní vstupní senzory (získávání a interpretace dat, praktické využití)
   - dotykové,
   - světelné,
   - zvukové,
   - ultrazvukové senzory.
3. Základní výstupní moduly (zasílání dat pro řízení modulů, řízení pohybu, praktické použití)
   - motory,
   - práce s displejem,
   - zvukový výstup,
   - světelné diody.
4. Programové řízení (struktura, syntax a využití)
   - podmínky,
   - cykly (s podmínkou na začátku, s podmínkou na konci, s pevným počtem průchodů),
   - čekání na dokončení úloh,
   - zastavení programu.
5. Proměnné (práce s proměnnou)
   - datové typy,
   - přetypování,
   - čtení a zápis proměnné.
6. Matematické operace
   - logické matematické operace (kombinace podmínek),
   - kontrola intervalů,

- náhodná čísla,
- funkce.

7. Tvorba vlastních metod (bloků)
   - vytváření a použití vlastních metod.
8. Rozšiřující vstupní moduly (získávání a interpretace dat, praktické využití)
   - teplotní,
   - barevné,
   - gyroskopické,
   - kompasové,
   - akcelerace,
   - vyhledávač IR signálů,
   - multiplexování vstupů.
9. Rozšiřující výstupní moduly (zasílání dat pro řízení modulů)
   - lineární motory,
   - multiplexování výstupů.
10. Ostatní možnosti prostředí
    - zaznamenávání a vizualizace hodnot,
    - časovače.
11. Spolupráce více jednotek, Bluetooth komunikace
    - párování přístrojů,
    - zasílání zpráv,
    - příjem a vyhodnocení zprávy.