Metody pro výuku programování na základních školách

Mgr. Zita Bajúszová

Bakalářská práce
2015

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2014/2015

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Mgr. Zita Bajúszová**
Osobní číslo:	**A12585**
Studijní program:	**B3902 Inženýrská informatika**
Studijní obor:	**Informační a řídicí technologie**
Forma studia:	**kombinovaná**
Téma práce:	**Metody pro výuku programování na základních školách**
Téma anglicky:	**Methods for Teaching Programming in Primary Schools**

Zásady pro vypracování:

1. **Prostudujte didaktické metody pro výuku programování na základních školách u nás i v zahraničí.**
2. **Specifikujte uživatelské požadavky pro systém, popř. jazyk pro výuku programování pro děti od 9–ti let.**
3. **Analyzujte stávající systémy a hry pro výuku programování s ohledem na splnění jednotlivých uživatelských požadavků.**
4. **Pokud nebude v předchozím bodě nalezen vyhovující systém, který by splňoval uživatelské požadavky z bodu 1, vytvořte specifikaci nového systému.**
5. **Navrhněte sadu úloh, od jednoduchých po složité, které děti budou v navrženém systému programovat.**

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **VORDERMAN, Carol, Jon WOODCOCK, Sean MCMANUS, Craig STEELE, Claire QUIGLEY a Daniel MCCAFFERTY. Help your kids with computer coding: a unique step-by-step visual guide, from binary code to building games. First American edition. 224 pages. ISBN 978-146-5419-569.**
2. **KUBICA, Jeremy. Best Practices of Spell Design. CreateSpace Independent Publishing Platform, 2013. ISBN 978-1481921916.**
3. **KUBICA, Jeremy. Computational fairy tales. [North Charleston, S.C: CreateSpace], 2012. ISBN 978-147-7550-298.**
4. **MAYER, By Jake. A tale of friends, enemies and Minecraft. San Bernardino, CA: [Jake Mayer], 2013. ISBN 978-148-9574-091.**
5. **BELLANCA, James A a Ronald S BRANDT. 21st century skills: rethinking how students learn. Bloomington, IN: Solution Tree Press, c2010, xxxi, 375 p. Leading edge (Bloomington, Ind.), 5. ISBN 19-352-4990-8.**
6. **ROBINSON, Ken. Out of our minds: learning to be creative. Fully rev. and updated ed. Chichester: Capstone, 2011, xvii, 326 s. ISBN 978-085-7081-490.**
7. **KLEMENT, Milan, Jiří KLEMENT a Jan LAVRINČÍK. Metody realizace a hodnocení výuky základů programování. Olomouc, 2012. ISBN 978-80-87658-01-7.**
8. **VANÍČEK, Jiří. Informatika pro základní školy. Vyd. 1. Brno: Computer Press, 2005, 88 s. Učebnice (Computer Press). ISBN 80-251-0630-6.**

Vedoucí bakalářské práce: **Ing. Tomáš Dulík, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce: **6. března 2015**

Termín odevzdání bakalářské práce: **22. května 2015**

Ve Zlíně dne 6. března 2015

L.S.

doc. Mgr. Milan Adámek, Ph.D.
děkan

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové/bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně 22. 5. 2015

.....
podpis diplomanta

ABSTRAKT

Cieľom bakalárskej práce je zmapovať súčasnú situáciu výučby algoritmizácie na druhom stupni základných škôl v Českej republike, analyzovať požiadavky na systém pre výučbu programovania z pohľadu cieľovej skupiny, učiteľov a správcov systému a analyzovať existujúce, najčastejšie používané edukačné systémy. Následne na základe zistených skutočností špecifikovať požiadavky a zostaviť sadu úloh pre unikátny edukačný softvér, určený pre výučbu programovania na druhom stupni základných škôl v Českej republike.

Klíčová slova: deti, základy programovania, programovanie pre neprogramátorov, metodika vyučovania programovania, software pre výučbu programovania, programovacie prostredie, design orientovaný na užívateľa, gamifikácia

ABSTRACT

The aim of bachelor thesis is to map the current situation of algorithmic teaching at the 2nd stage of primary school in the Czech republic, to analyze system requirements for teaching programming from the viewpoint of target group, teachers and system administrators and analyze the most commonly used educational systems. Afterward, on the basis of the knowledges to specify the requirements and prepare a set of tasks for a unique educational software designed for teaching programming at the 2nd stage of primary school in the Czech Republic.

Keywords: Children, How Programming Works, Programming for Non-Programmers, Methods for Teaching Programming, Educational Software, Programming Environments, User Centered Design, Gamification

Rada by som poďakovala svojmu vedúcemu bakalárskej práce Ing. Tomášovi Dulíkovi, Ph.D. za odborné vedenie, podnetné rady, informácie a trpezlivosť, ktoré mi poskytoval počas práce na mojej bakalárskej práci.

OBSAH

ÚVOD

Motivácia

Predkladaná bakalárska práca sa zaoberá problematikou výučby základov programovania a algoritmizácie na druhom stupni základnej školy. Dôvodov, prečo sme si vybrali práve uvedenú tému, je hneď niekoľko.

Prvým dôvodom je fakt, že schopnosť samostatne riešiť problémy a hľadať vhodné spôsoby ich riešenia, je jednou z kľúčových kompetencií, ktorú by si mali žiaci základnej školy osvojiť pred vstupom na vyššie stupne vzdelávania. Táto kompetencia pritom v širšom zmysle slova nie je nič iné, ako osvojenie si procesu algoritmizácie

Ďalší dôvod sa týka masového využívania informačno komunikačných prostriedkov vo všetkých sférach života človeka, či sa už jedná o školu, prácu alebo voľných čas. Napriek tomu len veľmi malá časť užívateľov dokáže spravovať svoje zariadenia, prispôsobiť si existujúcu aplikáciu, naprogramovať si jednoduchý program, alebo mať aspoň hmlistú predstavu o tom, ako dané aplikácie pracujú. Na prvý pohľad sa môže zdať, že pre bežných užívateľov uvedené schopnosti nemajú zmysel. Myslíme si však, že dané vedomosti umožnia aj bežnému užívateľovi jasne a zrozumiteľne formulovať svoje požiadavky na novú funkcionalitu či softvér, čo je dnes v mnohých zamestnaniach veľmi žiadaná zručnosť.

Mnoho žiakov, ktorí sa chcú naučiť programovať, však často krát nevydrží viac, ako pár hodín, pretože sa im to zdá buď príliš zložité, prípadne veľmi nudné[1]. Prečo je tak ťažké naučiť sa programovať? Pretože riešenie úloh je potrebné vyjadriť a zapísať vo forme, ktorú vyžaduje počítač. Aby to žiak zvládol, je potrebné získať niekoľko vedomostí a rozvíjať zručnosti z rôznych druhov oblastí, ako napríklad už spomínanú algoritmizáciu, logické a abstraktné myslenie, základné matematické operácie, ale aj písanie na klávesnici alebo anglický jazyk. Rozvoj týchto oblastí a vedomostí dokáže rozvíjať vhodne zvolený spôsob výučby programovania, čo znamená kombináciu vhodne zvoleného programovacieho, resp. skriptovacieho jazyka, vývojového prostredia alebo edukačného softvéru.

[1] V prípade nevhodne zvoleného softvéru na výučbu programovania.

Napriek uvedeným skutočnostiam sa programovanie na základných školách vyučuje skôr okrajovo a stále mu nie je venovaná dostatočná pozornosť ani zo strany štátu, ani zo strany samotných učiteľov.

Cieľ

Cieľom bakalárskej práce je:

> na základe analýzy požiadaviek na systém pre výučbu programovania z pohľadu cieľovej skupiny, učiteľov a správcom systému a analýzy existujúcich, najčastejšie používaných systémov, špecifikovať požiadavky a zostaviť sadu úloh pre unikátny edukačný softvér, určený pre výučbu programovania na druhom stupni základných škôl v Českej republike.

Cieľová skupina

Cieľovou skupinou, na ktorú sa zameriavame v rámci našej bakalárskej práce, sú žiaci druhého stupňa základných škôl v Českej republike. Veková hranica cieľovej skupiny je teda od 10 do15 rokov. Dôvodom, prečo sme si vybrali práve danú vekovú hranicu je, že deti v tom veku majú často krát záujem dozvedieť sa, ako programy, hlavne hry vznikajú, ako by si mohli urobiť vlastnú hru a podobne. Zložitosť programovania ich môže ale veľmi rýchlo odradiť, pokiaľ sa im neposkytnú vhodné nástroje. Z aplikácií, ktoré využívajú sú zvyknuté, že tieto aplikácie sú graficky na vysokej úrovni, interaktívne a intuitívne. Pri výbere, resp. vytváraní edukačného softvéru je potrebné akceptovať túto špecifickú požiadavku cieľovej skupiny.

Všeobecné princípy designu softvéru

Cieľom ideálneho softvéru na výučbu programovania pre deti je, aby bol použiteľný, efektívny a intuitívny. Tieto požiadavky na návrh softvéru, o ktoré sa budeme opierať pri špecifikácii systému na výučbu programovania, korešpondujú s prístupom, resp. výskumnou oblasťou nazývanou Interakcia Človek – Počítač (Human-Computer Interaction, HCI[2]), ktorá je prienikom matematickej informatiky, behaviorálnych vied, dizajnu a mediálnych štúdií.

---

[2] http://en.wikipedia.org/wiki/Human%E2%80%93computer_interaction [cit. 2015-04-05].

Na dosiahnutie uvedeného cieľa HCI špecifikuje niekoľko princípov, ktoré môžeme zhrnúť nasledovne:

- „jednoduchá komunikácia s užívateľom - užívateľské rozhranie má byť tak jednoduché, ako je to len možné;
- rešpektovanie „jazyka“, ktorým hovorí užívateľ - užívateľské rozhranie má rešpektovať cieľovú skupinu, pre ktorú je určené;
- minimalizovanie pamäťovej náročnosti softvéru;
- dodržiavanie konzistentnosti – rovnaký príkaz alebo prvok spúšťa rovnakú udalosť naprieč celým softvérom;
- poskytovanie spätnej väzby zo strany sofvéru – užívateľ je v každom časovom okamihu práce so softvérom dostatočne informovaný, ako ho má používať a čo sa od neho očakáva;
- uvádzanie navigácie pre rýchlu orientáciu;
- poskytnutie možnosti kedykoľvek ukončiť prácu so softvérom – možnosti ako prácu uložiť a ukončiť musia byť jasne a viditeľne komunikované;
- uvádzanie jasných a užitočných chybových hlásení;
- dostatočné poskytovanie pomoci a podpornej dokumentácie.“ [1]

Prehľad členenia bakalárskej práce

Bakalárska práca je členená na dve časti a to Teoretickú a Praktickú. Teoretická časť sa zameriava na hlbšiu analýzu danej problematiky, vrátane zachytenia spoločenskej situácie, v ktorej je problematika ukotvená.

Teoretickú časť tvoria štyri kapitoly, konkrétne 1. Výučba programovania na základných školách v ČR, 2. Požiadavky na systém pre výučbu programovania z pohľadu cieľovej skupiny, 3. Užívateľské požiadavky na systém pre výučbu programovania a 4. Analýza existujúcich systémov. Prvá kapitola mapuje súčasnú situáciu v oblasti výučby informatiky jednak na úrovni štátu, jednak priamo na úrovni základných škôl, vrátane časovej dotácie, analýzy vybraných kurikúl a najčastejšie používaných programov na výučbu programovania v Českej republike. Druhá kapitola sa venuje požiadavkám na systém pre výučbu programovania z pohľadu žiakov 2. stupňa základnej školy. V rámci tejto kapitoly analyzujeme rôzne prístupy, nástroje a jazyky pre výučbu programovania a posudzujeme ich z hľadiska vhodnosti pre cieľovú skupinu. Záver kapitoly obsahuje odporúčania pre špecifikáciu

a návrh vhodného softvéru. Tretia kapitola sa zase venuje analýze požiadaviek na systém z pohľadu učiteľov a správcov tohto systému. Dôležitými kritériami sú v tomto prípade cena, požiadavky na hardvérové vybavenie a podpora systému. Štvrtá kapitola obsahuje podrobnejší opis najčastejšie využívaných systémov používaných na výučbu programovania vo svete. Na konci kapitoly sa nachádza zhrnutie vo forme tabuľky, ktoré prehľadnou formou zobrazuje vlastnosti daných systémov z pohľadu požiadaviek stanovených v kapitole 2. a 3. v zmysle spĺňa/nespĺňa, prípadne doplňujúcu poznámku.

Praktickú časť tvoria dve kapitoly, a to 5. Špecifikácia systému pre výučbu programovania a 6. Sada úloh. Špecifikácia je tvorená s ohľadom na splnenie požiadaviek užívateľov a cieľovej skupiny a obsahuje miminálne hardwarové požiadavky na databázový server, aplikačný server a klienta, uvádza zoznam operačných systémov a zariadení, s ktorými má byť daný softvér kompatibilný, navrhuje vhodné spôsoby riešenia, použité technológie, vlastnosti systému a pod.

Sada úloh obsahuje herné stratégie pre samotný softvér.

I. TEORETICKÁ ČASŤ

1 VÝUČBA PROGRAMOVANIA NA ZÁKLADNÝCH ŠKOLÁCH V ČR

1.1 Zmapovanie súčasnej situácie

Príprava školskej dokumentácie na úrovni vzdelávacej inštitúcie, teda školy ako takej, a na úrovni jednotlivých predmetov, upravuje v ČR od 1. 9. 2005 *Rámcový vzdělávací program pro základní vzdělávání*[3]. Tento program zaviedol novú, kurikulárnu politiku vytvárania dokumentácie, vďaka ktorej sa kurikulárne dokumenty vytvárajú na dvoch úrovniach – štátnej a školskej.

„Státní úroveň v systému kurikulárních dokumentů představují Národní program vzdělávání a rámcové vzdělávací programy (dále jen RVP). Národní program vzdělávání vymezuje počáteční vzdělávání jako celek. RVP vymezují závazné rámce vzdělávání pro jeho jednotlivé etapy – předškolní, základní a střední vzdělávání. Školní úroveň představují školní vzdělávací programy (dále jen ŠVP), podle nichž se uskutečňuje vzdělávání na jednotlivých školách.“ [2, s. 9]

Nás bude zaujímať časť C, ktorá okrem iného obsahuje definíciu kľúčových kompetencií a stanovuje rámcoví obsah pre jednotlivé vzdelávacie oblasti, kde sa nachádza aj *oblast Informační a komunikační technologie*.

Pod kľúčovými kompetenciami sa rozumie súhrn vedomostí, zručností, hodnôt a postojov, ktoré by si mal osvojiť každý žiak. Pre etapu základného vzdelania medzi kľúčové kompetencie patria (1) kompetencie k učeniu; (2) kompetencie k riešeniu problémov; (3) kompetencie súvisiace s komunikáciou; (4) kompetencie sociálne a personálne; (5) kompetencie občianske a (6) kompetencie pracovné.

Ako môžeme vidieť, už priamo v týchto kľúčových kompetenciách nájdeme potrebu naučiť žiakov samostatne a vhodným spôsobom riešiť problémy, čo v širšom zmysle slova nie je nič iné, ako proces algoritmizácie.

Vzdelávacia oblasť *Informační a komunikační technologie* (IKT) je charakterizovaná ako oblasť, ktorá umožňuje dosiahnuť žiakom základnú úroveň informačnej gramotnosti. Tiež

---

[3] Od roku 2013 je k dispozícii upravená verzia tohto programu.

je dané, že IKT je povinnou súčasťou základného vzdelania 1. a 2. stupňa základných škôl. Ako jedným z cieľov je uvedená schopnosť formulovať požiadavky a využívať pri interakcii s počítačom algoritmické myslenie. Keď sa však bližšie pozrieme na učivo určené pre našu cieľovú skupinu zistíme, že sa v ňom nenachádza ani zmienka o algoritmizácii a už vôbec nie o základoch programovania. Učivo, ktoré je stanovené v RVP, je zamerané len na vyhľadávanie, spracovanie a využitie informácií a najčastejšie je rozdelené na okruhy zamerané na základy práce s počítačom, grafický editor, programy zamerané na prezentáciu a internet.

Ešte doplníme, že logické úlohy, ktoré tiež môžu byť doplnkom algoritmizačných úloh, sa nachádzajú vo vzdelávacom obsahu vzdelávacieho odboru Matematika a jej aplikácie pre 2. stupeň v časti *Nestandardní aplikační úlohy a problémy*. V podstate sa jedná o okrajové učivo.

IKT tak isto nefiguruje ako Prierezová téma, hoci sa dotýka takmer všetkých vzdelávacích oblastí.

1.2 Časová dotácia

Minimálna časová dotácia, ktorú uvádza rámcový učebný plán, a ktorá je stanovená štátom, je stanovená na 1 vyučovaciu hodinu pre 2. stupeň s tým, že je možné ju doplniť o ďalšie hodiny z disponibilnej časovej dotácie. Táto disponibilnú časová dotácia je v súčasnosti stanovená na 24 vyučovacích hodín, z toho 6 je učených pre druhý cudzí jazyk. Inak je jej využitie v plnej kompetencii riaditeľa školy.

Už na prvý pohľad je zrejmé, že napriek významu, ktoré v súčasnosti informačné technológie majú a potenciálu rozvoja základných kľúčových kompetencií v rámci tohto vyučovacieho predmetu, sa podľa dotačnej schémy stále jedná o okrajový predmet.

Príklad učebného plánu pre 2. stupeň základnej školy, ktorý dodržiava pokyny RVP a predstavuje štandardný model[4], vyzerá bez rozdelenia disponibilnej časovej dotácie nasledovne:

[4] Rozdelenie hodín medzi jednotlivé vyučovacie predmety v rámci vzdelávacej oblasti sa môžu samozrejme líšiť. Rozdiel však bude len v prerozdelení medzi jednotlivými ročníkmi.

Tabuľka 1: Príklad učebného plánu pre 2. stupeň základnej školy

Vzdelávacia oblasť	**Vyučovací predmet**	**Ročník**				**Spolu**	**Časová dotácia v RVP**
		6.	7.	8.	9.		
Jazyk a jazyková komunikace	Český jazyk a literatura	3	4	4	5	15	15
	Anglický jazyk	3	3	3	3	12	12
Matematika a její aplikace	Matematika	4	4	3	4	15	15
Informační a komunikační technologie	**Informatika**	**1**	**–**	**–**	**–**	**1**	**1**
Člověk a společnost	Dějepis	2	2	2	2	11	11
	Výchova k občanství	1	1	1			
Člověk a příroda	Fyzika	1	1	2	2	6	21
	Chemie	–	–	1	2	3	
	Přírodopis	2	1	2	2	7	
	Zeměpis	2	1	1	1	5	
102	Hudební výchova	1	1	1	1	4	10
	Výtvarná výchova	2	2	1	1	6	
Člověk a zdraví	Tělesná výchova	2	2	2	2	8	10
	Výchova ke zdraví		1	1		2	
Člověk a svět práce	Svět práce	1	1	1		3	3

1.3 Stav českých škôl z pohľadu informatiky

Stav českých škôl z pohľadu informatiky nie je pozitívny. Okrem extrémne nízkej časovej dotácie, o ktorej sme sa zmienili v predchádzajúcej podkapitole, vidí Jiří Vaníček dôvody:

- „v dynamice oboru (během několika posledních let se ve společnosti objevuje řada nových témat souvisejících s technologiemi, jako sociální sítě, e-learning, cloudová řešení, webové aplikace, počítačová bezpečnost, a jsou snahy zařadit je co nejdříve do výuky);

- v nevyhraněnosti oboru (vzdělávací oblast nezahrnuje základní informatická témata, jako jsou algoritmizace, porozumění informacím);
- v roztříštěnosti informatického vzdělávání (v RVP ZV jsou některá informatická témata součástí vzdělávací oblasti matematika, počítačově technická v oblasti člověk a svět práce a ve vzdělávací oblasti ICT jsou to témata vesměs týkající se digitální gramotnosti, tedy uţivatelské zvládnutí vybraných především kanclářských aplikací a používání služeb Internetu);
- s tím souvisí i zaměření vzdělávací oblasti ICT na konzumní, nikoliv autorský nebo tvůrčí přístup k využití technologií, který vdůsledku nerozvíjí osobnost učícího se
- jiným problémem je neodbornost učitelů (absolventů učitelského směru informmatického zaměření mezi vyučujícími informatických předmětů je v českých školách výrazná menšina).“ [3, s. 15]

1.4 Používané metódy a programovacie jazyky

Zdrojom dát v podkapitole Používané metódy a programovacie jazyky je výskum, ktorý na základných školách v Českej republike zrealizoval v rámci svojej diplomovej práce *Výuka základů programování v prostředí Scratch* v roku 2014 Jan Krejsa. [4] Prieskum na základných školách bol realizovaný formou dotazníkov, čo je jedna z metód kvantitatívneho výskumu. Cieľovou skupinou prieskumu boli učitelia Informatiky. Ich výber nepodliehal žiadnym špeciálnym kritériám. Kontaktné údaje čerpal autor z internetovej stránky http://www.atlasskolstvi.cz/. Celkový počet respondentov bol 401.

Až 86% respondentov uviedlo, že v rámci Informatiky vyučujú aj programovanie. Veľmi málo z nich však začína už v 5. ročníku. Väčšina z nich začína základy programovania vyučovať až v 9. triede, čo je podľa nášho názoru pomerne neskoro a aj to s priemernou časovou dotácia iba 2 hodiny týždenne. Napríklad Rudolf Pecinovský uvádza [5], že žiaci sú schopní pochopiť základy algoritmizácie a vytvárať jednoduché programy už v prvej triede. Prekážkou v ďalšom vývoji je pre nich absencia abstraktného myslenia, preto odpo-

rúča začať s „oficiálnou“ výučbou programovania medzi treťou a šiestou triedou, čo je podstatne skôr, ako sa začína so základmi programovania na bežných základných školách[5].

Veľmi zaujímavé boli dôvody, ktoré uvádzali respondenti, ktorí nevyučujú programovanie. Najčastejšou odpoveďou bolo, že programovanie nie je uvedené v už spomínanom RVP. Ďalšie uvádzané dôvody boli:

- nedostatočná časová dotácia výučby,
- zložitosť výučby a chýbajúci software,
- podceňovanie žiakov, resp. podceňovanie významu programovania pre žiakov a pod.

Na základe vlastných skúseností z výskumu využívania Lego Mindstorms môžeme konštatovať, že podobné dôvody uvádzali učitelia aj na základných školách v Slovenskej republike. Dovolím si preto tvrdiť, že uvedené dôvody sú dôvody, prečo sa programovanie na základných školách všeobecne vyučuje v tak malej miere.

Medzi programovacie jazyky, resp. software, ktorý sú podľa uvedeného prieskumu zastúpené v najväčšej miere, patria Baltík a Imagine Logo. Podrobnejšie sa im budeme venovať v kapitole 4. Analýza existujúcich systémov. V štatisticky zanedbateľnom množstve boli uvedené ďalšie programy ako Karel a Lego Mindstorms, alebo programovacie, resp. skriptovacie jazyky ako PHP, Java, Perl, Python.

[5] Pod bežnou základnou školou myslíme školu, ktorá nie je určená pre nadané deti, ani nie je zameraná na vyučovanie informatiky.

2 POŽIADAVKY NA SYSTÉM PRE VÝUČBU PROGRAMOVANIA Z POHĽADU CIEĽOVEJ SKUPINY

Pri výbere systému pre výučbu programovania je potrebné mať na mysli ciele výučby programovania žiakov na základnej škole. Cieľom nie je naučiť ich vytvárať formálne dokonalé programy, alebo syntax jedného vybratého programovacieho, resp. skriptovacieho jazyka. Cieľom je:

- vedieť jasne definovať problém;
- určiť stratégiu riešenia problému rozdelením na podproblémy;
- stanoviť postupnosť krokov, ktoré budú viesť k riešeniu podproblémov, ako aj k riešeniu problému – tento proces je známy pod pojmom algoritmizácia;
- overiť navrhnuté riešenie, v prípade potreby vedieť nájsť a odstrániť chybu, či už logického, alebo syntaktického charakteru.

Pri výučbe programovania má učiteľ na výber niekoľko nástrojov a metód, ktoré si môže zvoliť v závislosti od zloženia skupiny a požadovanej náročnosti. Jednou z možností je vybrať si jeden z programovacích jazykov, druhou zvoliť si edukačný softvér.

Z didaktického hľadiska je vhodné dodržať nasledujúci postup:

- „čo najskôr umožniť tvorbu programov, aby si žiak mohol všetko sám vyskúšať;
- nepredbiehať, nepoužívať prvky jazyka, ktoré ešte neboli vysvetlené. Výklad by mal byť usporiadaný, každý nový prvok jazyka by mal byť vysvetlený;
- informácie podávať po malých častiach. Čím menej nových poznatkov žiakom predložíme na vyučovacej hodine, tým väčšia je nádej na to, že si ich žiaci zapamätajú a naučia sa ich používať;
- úlohy musia byť zaujímavé. Najlepšie úlohy sú také, keď žiak vie, ako by problém riešil sám, a toto riešenie potrebuje formalizovať tak, aby riešenie zvládol aj počítač;
- riešenia nemajú byť príliš neprehľadné. Časť programu, ktorá rieši zadanú úlohu, a tým aj precvičuje preberanú konštrukciu a prvok jazyka, sa nemá strácať v záplave príkazov vstupu a výstupu a rôznymi pomocnými operáciami.
- od začiatku dbať na zásady moderného programovania. Ak chceme žiakov naučiť riešiť aj zložitejšie problémy, treba od samého začiatku riešiť problémové úlohy rozložením na niekoľko jednoduchších úloh a tie riešiť samostatne.

- riešiť netriviálne problémy. Schopnosť riešiť zložité úlohy je tou najcennejšou zručnosťou, ktorú si žiaci pri výučbe programovania môžu osvojiť. Vyučujúci by sa mal sústrediť na to, aby žiakov učil riešiť zložitejšie úlohy, aby si žiaci osvojili zásady moderného programovania.“[6]

Okrem vyššie spomenutých princípov je potrebné mať na mysli aj špecifiká skupiny žiakov. Pravdou je, že v rámci štandardnej vyučovacej hodiny môže byť táto skupina značne heterogénna. Väčšina žiakov však v súčasnosti používa rôzne aplikácie v rámci svojich mobilných zariadení alebo tabletov, ktoré sú na vysokej grafickej úrovni, sú interaktívne a intuitívne. Je preto veľmi pravdepodobné, že rovnaké vlastnosti budú očakávať aj od edukačného softvéru.

2.1 Nižšie vs. vyššie programovacie jazyky

Programovací jazyk je umelo vytvorený jazyk, ktorý bol vytvorený na zápis počítačového programu. Má presne definovanú syntax a sémantiku, čím pripomína prirodzený jazyk. Na rozdiel od neho však vznikol cielene a mení sa podľa toho, ako sa menia dané ciele. To znamená, že zmeny v rámci jazyka nie sú postupné, ale skokové. Každý program má svoj zdrojový kód, ktorý slúži na zápis algoritmu. Programovacie jazyky môžeme členiť z viacerých hľadísk a jedným z nich je práve členenie na nižšie a vyššie programovacie jazyky.

Za nižší programovací jazyk sa v informatike označuje jazyk, ktorý poskytuje malú alebo žiadnu abstrakciu od toho, ako funguje procesor počítača. Označenie "nižší" odkazuje na veľmi malý alebo žiadny rozdiel medzi daným programovacím jazykom a strojovými inštrukciami procesora (resp. jeho inštrukčnou sadou). Preto bývajú nízkoúrovňové programovacie jazyky označované ako "tesne späté s hardvérom".[7] Veľkým nedostatkom týchto programovacích jazykov z pohľadu výučby je ich závislosť na dodatočnom hardvéri a pomerne veľké nároky na programátora, preto je ich využitie skôr záležitosťou voľnočasových aktivít, keď vôbec.

---

[6] http://www.lackovaj.veveve.info/videonavod/Metodicky-den-2013.pdf [cit. 2015-04-05].

[7] http://cs.wikipedia.org/wiki/Ni%C5%BE%C5%A1%C3%AD_programovac%C3%AD_jazyk [cit. 2015-04-05].

Vyššie programovacie jazyky sú rozšírenejšie čo sa používania týka, takže sa s nimi žiaci majú možnosť stretnúť doslova na každom kroku, čo môže byť pre nich motivačným faktorom. Zároveň sa oproti nižším programovacím jazykom vyznačujú vyššou mierou abstrakcie, to znamená, že ich zápis má bližšie k tomu, ako problémy reálne človek spracováva. Sú teda lepšie zrozumiteľné, čo uľahčuje aj samotný vývoj programu a sú tým pádom vhodnejšie aj na výučbu programovania. Učiteľ tu má na výber z veľkého množstva jazykov, či už procedurálnych, alebo objektovo-orientovaných. Objektovo-orientované jazyky však vyžadujú pochopenie zložitejších vlastností, ako je napríklad polymorfizmus, zapúzdrenie, dedičnosť a pod. a vyššiu mieru abstrakcie, preto sú na výučbu programovania pre ne-programátorov náročnejšie. Na výučbu programovania na základných školách sa z tejto skupiny jazykov využíva najmä C[8], C++, PHP, Python, prípadne Java. Ich využitie je však do veľkej miery tiež záležitosťou voľnočasových aktivít, prípadne ako súčasť vyučovania programovania a algoritmizácie na školách pre nadané deti.

2.2 Výučbové vs. profesionálne nástroje

Každý učiteľ, ktorý začína učiť programovanie, stojí pred voľbou, aký nástroj zvoliť, či výučbový alebo profesionálny. Výber nástroja aj vývojového prostredia je do veľkej miery ovplyvnený skúsenosťami žiakov s programovaním.

Pre výučbu základov programovania sú vhodnejšie výučbové nástroje, resp. špecializované programovacie jazyky, ktoré boli navrhnuté tak, aby umožnili jednoduché pochopenie zásad programovania. Takých jazykov a na nich nadväzujúcich vývojových prostredí je vo svete celá rada. Väčšina z nich zavádza jednoduchý virtuálny svet, v ktorom žiaci riešia rôzne úlohy.[5] V rámci našej práce si v kapitole 4. Analýza existujúcich systémov bližšie predstavíme niektoré z nich. Nevýhodou väčšiny týchto výučbových nástrojov je však ich vizuálny, resp. ikonický charakter, prípadne vlastný skriptovací jazyk, čo do veľkej miery sťažuje prechod na klasické programovacie jazyky.

Pod profesionálnymi nástrojmi máme na mysli programovacie jazyky a vývojové prostredia, ktoré využívajú profesionálni programátori. Sem patria napríklad už vyššie spomínané jazyky C, C++, PHP, Python alebo Java.

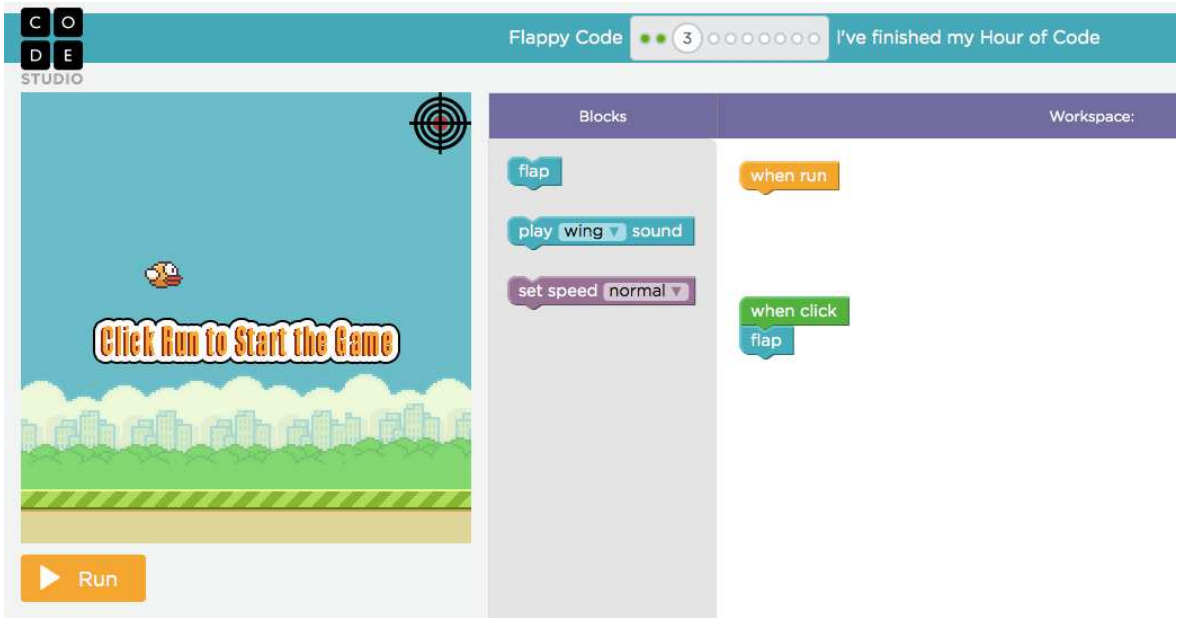
---

[8] Ukážka materiálov kurzu jazyka C pre žiakov kvarty a kvinty osemročného gymnázia: http://www.rirs.iedu.sk/Dokumenty/Ucebnice_Anino_Belan/C.pdf [6] [cit. 2015-04-05].

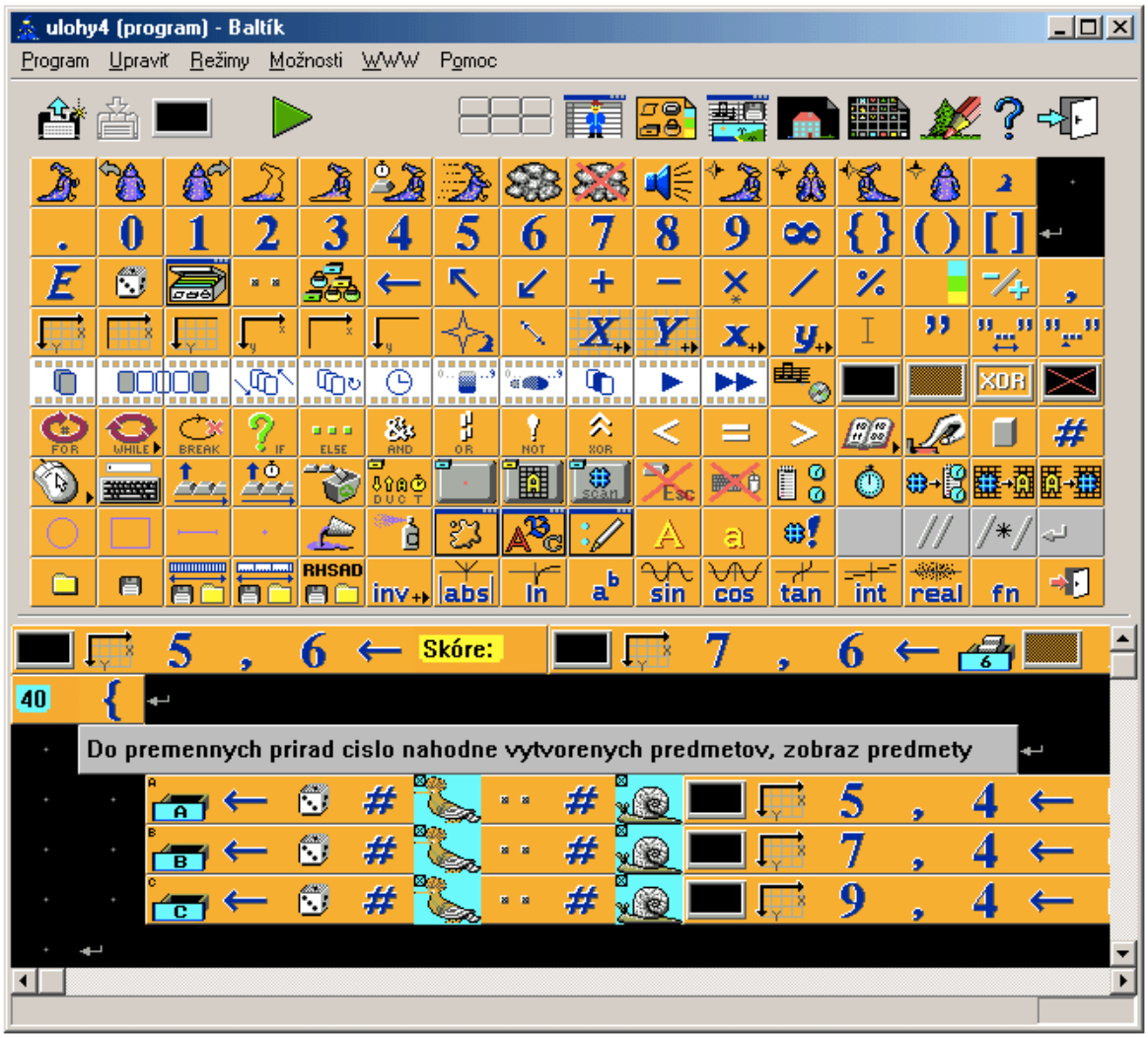
Našou snahou je vytvoriť edukačný softvér, ktorý spája oba tieto nástroje a predstavuje tak plynulý prechod od špecializovaného k programovaciemu jazyku. Z hľadiska syntaxe a sémantiky sme si zvolili programovací jazyk Python. Dôvody a bližší popis tohto programovacieho jazyka uvádzame v praktickej časti v kapitole 5. Špecifikácia systému pre výučbu programovania.

2.3 Vizuálne vs. textuálne nástroje

Väčšina výučbových nástrojov využíva na komunikáciu so svojimi užívateľmi grafické rozhranie. Inými slovami, zadávanie príkazov prostredníctvom príkazového riadku, tzn. textu, nahradili ikony, alebo iné grafické prvky. Zástancovia vizuálnych programovacích jazykov často argumentujú, že zníženie alebo odstránenie textu v programovaní umožní užívateľom sústrediť sa na podstatu problému, keďže nie sú rozptyľovaní problematikou syntaxe. Žiadna štúdia však tento predpoklad nepotvrdila. Výhodu použitia vizuálnych nástrojov pre vyučovanie na základných školách vidíme v tom, že v nižších ročníkoch ešte nemusia mať deti dostatočné zručnosti v písaní na klávesnici. Vizuálne nástroje by však mali byť postupne vo výučbe vo vyšších ročníkoch nahrádzané textuálnymi.



Obrázok 1: Ukážka zadávania príkazov pomocou grafických prvkov[9]



Obrázok 2: Ukážka zadávania príkazov pomocou ikon - Baltík[10]

2.4 Využitie herných princípov vo vyučovaní programovania

K tejto téme sme čerpali informácie najmä od Filipčíka [7]. Vo vyučovaní, obzvlášť pri dlhodobých a ťažkých aktivitách, kde si na výsledky svojej práce žiak musí počkať, ako to v prípade učenia sa programovania určite je, je veľmi dôležitým momentom motivácia. Žiak, ktorý má motiváciu, zvládne učivo rýchlejšie a nevzdá sa pri prvých ťažších úlohách.

[9] Zdroj: https://studio.code.org /[cit. 2015-04-05].
[10] http://baltik.infovek.sk/zakl_inf3.htm [cit. 2015-04-05].

Existuje viacero metód vonkajšej motivácie. V súčasnosti je v tejto oblasti často počuť o gamifikácii. Je to metóda, ktorá je postavená na využívaní herných princípov v neherných oblastiach. Náš navrhovaný edukačný systém je postavený ako hra, preto je logické, že na motiváciu bude využívať práve gamifikačné prvky, ktoré sú deťom, resp. žiakom dobre známe.

Existuje veľké množstvo techník, ktoré slúžia na motiváciu hráčov. Najčastejšie sa využívajú nasledovné:

- techniky postavené na čase, ako napríklad odpočítavanie, meranie rýchlosti a pod.;
- prvky postavené na náhode, šťastí, alebo prekvapení, prípadne tajomstve
- prvky, ktoré vyjadrujú dosiahnutie určitého stupňa, splnenie danej úlohy a pod. Do tejto skupiny patria odznaky, postup do ďalšej úrovne, získavanie špeciálnych bonusov atď.;
- interakcia s komunitou, čiže s ďalšími hráčmi, resp. spolužiakmi.

Navrhovaný edukačný systém bude z vymenovaných techník využívať prevažne prvky, ktoré vyjadrujú dosiahnutie určitého stupňa či splnenie danej úlohy a interakciu s komunitou.

Body

Udeľovanie bodov, ako odmena za vykonané aktivity, je povinnou súčasťou každej hry. Bodovací systém môže mať mnoho podôb a prevedení. My budeme využívať systém postupnej akumulácie bodov. Body budú verejne prístupné a zobrazované spolu s menom a avatarom užívateľa, resp. hráča. Získavať sa budú sa postup na ďalšiu úroveň, umiestnenie v súťažiach a splnenie extra úloh. Body majú okrem motivácie žiakov zmysel aj pre samotného učiteľa, ktorý tak priebežne môže sledovať aktivitu žiakov.

Obrázok 3: Príklad zobrazovania bodov v rámci aplikácie Zamzee[11]

Rebríčky

Rebríčky (angl. leaderboards) majú veľmi silný motivačný charakter a predstavujú jeden zo spôsobov, ako môžu jednotlivý hráči spolu interagovať. Bývajú zobrazené formou tabuľky, kde jednotlivé stĺpce reprezentujú meno hráča (často krát zobrazené spolu s avatarom), získané body, úroveň, prípadne ďalšie informácie o hráčovi, ako získané odznaky, čas strávený hraním, resp. riešením úloh a podobne.

Existujú dva hlavné typy rebríčkov. Prvý je tzv. neodradzujúci rebríček, ktorý zobrazuje hráča vždy v strede tabuľky. Hráč teda vidí len určitý počet hráčov pod ním a nad ním. Táto informácia mu stačí na to, aby videl, koľko bodov musí dosiahnuť, aby predbehol

[11] Zdroj: http://yukaichou.com/wp-content/uploads/2012/10/Zamzee-Leaderboard.png [cit. 2015-04-05].

hráča pred ním, nevidí však, aké skóre majú hráči na prvých miestach v rebríčku. Druhým typom je nekonečný rebríček, ktorý zobrazuje zoznam všetkých hráčov.

V rámci nášho systému budeme využívať druhý typ rebríčka.

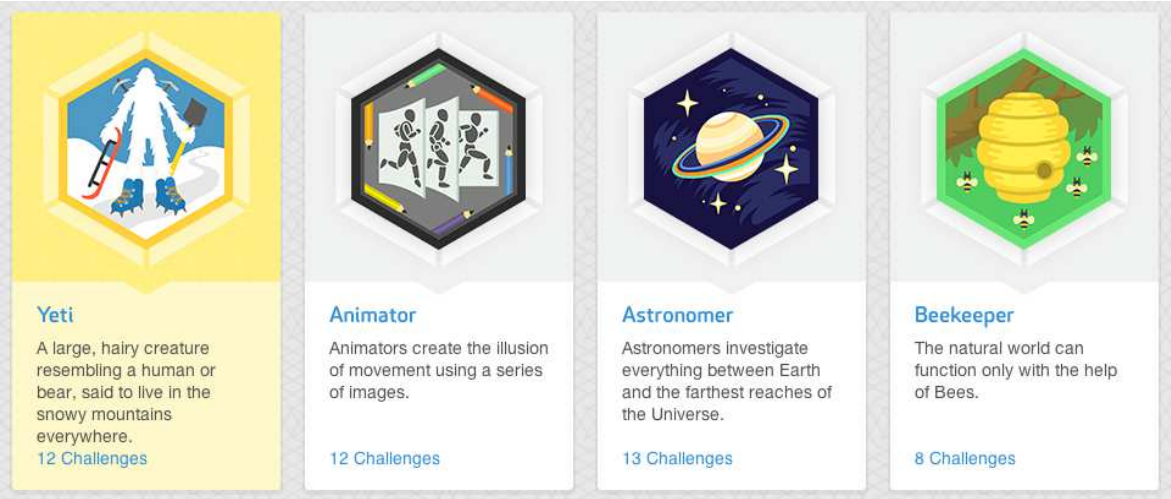
Úrovne

„Systém úrovní (angl. levels) našiel široké využitie predovšetkým vo videohrách. V hrách žánru RPG (Role Playing Game) hráč ovláda postavu, ktorá plnením úloh a vykonávaním množstva iných aktivít získava skúsenosti. Po dosiahnutí určitej hranice sa získané skúsenosti zameníme za novú, vyššiu úroveň. S navýšením úrovne postava zvyčajne získava i nové schopnosti a možnosti ako prechádzať hrou.“ [7, s. 7]

Úrovne sú zvyčajne naviazané na ďalšie prvky, najčastejšie na body. V prípade výučbových softvérov ide o pomerne jednoduchý a priamočiary spôsob zobrazenia úrovne žiaka.

Odznaky

Odznaky (angl. badges) predstavujú vizuálny spôsob zaradenia v rámci hierarchie. Systém prideľovania odznakov je známy aj z iných oblastí, ako napr. vyjadrenie hodnosti v armáde. Získavanie odznakov môže byť viazané na splnenie určitých úloh, na postup na ďalšiu úroveň a pod. Spracovanie a obsah sa samozrejme líši od aplikácie k aplikácii, jedno majú ale vždy spoločné a tým je grafická príťažlivosť. Získavanie odznakov bude v prípade nášho systému viazané na plnenie určitých výziev, podobne, ako je uvedené na obrázku č. 4.



Obrázok 4: Príklad grafického spracovania odznakov pre účely gamifikácie[12]

Súťaže, výzvy a úlohy

Súťaže, výzvy (angl. challenges) a úlohy (angl. missions) podporujú súťaživosť a dávajú možnosť učiteľovi zadávať náročnejšie úlohy, ktorými si žiaci precvičujú prebraté učivo na dobrovoľnej báze. Súťaže, výzvy alebo úlohy bývajú zvyčajne časovo ohraničené, čo zvyšuje ich exkluzivitu. Ako sme už uvádzali vyššie, v našom systéme bude práve získavanie odznakov závisieť od množstva absolvovaných súťaží a splnených výziev, resp. úloh.

[12] Príklad grafického spracovania odznakov uvádzame zo stránky https://diy.org/. [cit. 2015-04-05]. Je to aplikácia určená práve deťom a jej snahou pomocou gamifikácie rozvíjať nové zručnosti a vedomosti. Okrem odznakov je vidieť aj počet výziev, ktoré je potrebné k získaniu jednotlivých odznakov.

3 UŽÍVATEĽSKÉ POŽIADAVKY NA SYSTÉM PRE VÝUKU PROGRAMOVANIA

Pre potreby tejto kapitoly budeme pod pojmom užívateľ systému rozumieť učiteľa, resp. správcu systému.

3.1 Požiadavky na vybavenie

3.1.1 Hardvér

Našou snahou je vytvoriť taký edukačný softvér, ktorý bude mať minimálne nároky na hardvérové vybavenie. Akékoľvek prídavné zariadenia obmedzujú použitie na úzku skupinu užívateľov a tiež obmedzujú použitie len na školské prostredie.

Medzi dôležité požiadavky na hardvér patrí monitor s rozlíšením minimálne 1366 pixlov na 768 pixlov, grafická a zvuková karta, myš, klávesnica a pripojenie na Internet. Jedná sa o štandardné vybavenie, ktorým v súčasnosti disponuje každá počítačová učebňa.

3.1.2 Softvér

Softvérové požiadavky zahŕňajú požiadavku na taký tip licencie, ktorý umožní využívanie daného softvéru v čo najširšom rozsahu a nebude sa obmedzovať iba na školské zariadenia. Dôvodom je, aby deti mali možnosť riešiť zadané úlohy kedykoľvek a kdekoľvek. Druhou požiadavkou je minimálna starostlivosť o zvolený softvér. Podľa našich znalostí školského prostredia je na školách k dispozícii často iba jeden správca systému, ktorý väčšinou zastupuje v rámci školy aj ďalšie funkcie.

3.1.2.1 Licencie

S distribúciou softvéru je pevne zviazaná problematika licencií. Licencia predstavuje právo užívateľa používať daný softvér, a zároveň upravuje aj spôsob jeho používania. Jeden, softvér môže byť užívateľovi poskytovaný pod rôznymi druhmi licencií, za rôzne ceny. V prípade edukačného softvéru zohráva spôsob uplatnenia licencií a ich cena kľúčovú úlohu pri rozhodovaní o tom, ktorý softvér si nakoniec škola ako užívateľ vyberie, resp. zakúpi. Najjednoduchšie rozdelenie softvéru podľa tipov licencií je rozdelenie na slobodný a open-source softvér a proprietárny softvér.

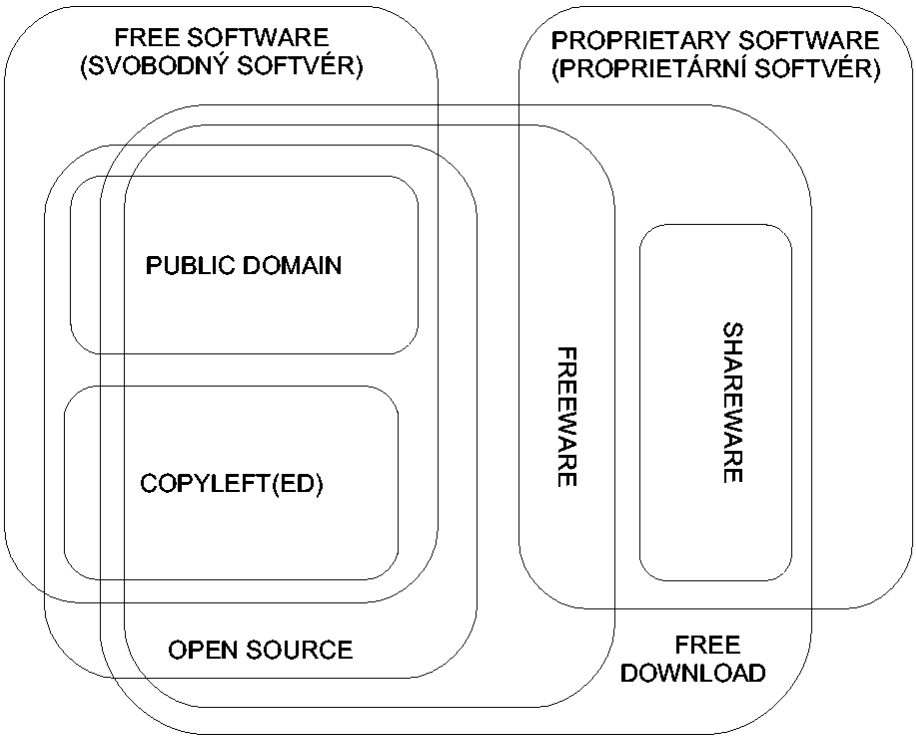
Proprietárny softvér

Jedná sa o komerčný tip licencie, určený pre bežný predaj. Vo väčšine prípadov sú pre školy a žiakov poskytované zľavy. Nevýhodou je, že takýto softvér býva limitovaný pre školské zariadenia. Žiaci si ho nemôžu inštalovať na svoje zariadenia a tým pádom je práca s nimi obmedzená iba na priestory školy.

Slobodný a open-source softvér

Softvér, zaradený medzi free alebo open- source, spadá väčšinou do dvoch kategórií: (1) tie, ktoré dovoľujú neobmedzené používanie a úpravu kódu aj za účelom ďalšieho použitia v proprietárnom softvéri (tzv. permissive licences – najznámejšie sú BSD alebo MIT licencia) a (2) tie, ktoré dovoľujú neobmedzené používanie a úpravu kódu, s tým, že všetky ďalšie vytvorené programy musia byť ďalším užívateľom poskytnuté zdarma (najznámejšou licenciou tohto druhu je GNU GPL).

Výhodou oproti predchádzajúcemu tipu licencií je samozrejme obstarávacia cena, ktorá je v prípade druhého typu nulová. Nevýhodou je často krát slabá podpora pre užívateľov a pomerne veľké riziko, že softvér sa nebude ďalej vyvíjať.



Obrázok 5: Mapa základných tipov softvérových licencií[13]

3.1.2.2 Model nasadenia

Základné modely nasadenia, ktoré má užívateľ k dispozícii, sú dva, a to SaaS (softvér ako služba - Software as a Service) a on-premises softvér.

SaaS

SaaS je model nasadenia softvéru, kedy samotnú aplikáciu prevádzkuje priamo poskytovateľ služby. Služba je poskytovaná prostredníctvom Internetu. Výhodou je, že škola v tomto prípade nemá náklady spojené s nákupom a prevádzkou softvéru ani hardvéru. Užívateľ, a teda aj žiak, sa do aplikácie dostane z akéhokoľvek zariadenia na akomkoľvek mieste. Jedinou podmienkou je pripojenie na Internet.

On-premises softvér

On-premises softvér je presný opak modelu SaaS. Jedná sa o model, kedy si konečný užívateľ inštaluje softvér do svojho zariadenia.

[13] Zdroj: http://upload.wikimedia.org/wikipedia/commons/4/43/Softverove_licence.png [cit. 2015-04-05].

3.2 Obstarávacie náklady

Náklady na obstaranie softvéru, resp. služby, sú dôležitým faktorom pri výbere edukačného nástroja a je zrejmé, že každý užívateľ by si prial obstarať takýto nástroj za čo najnižšiu cenu.

Pri on-premises softvéri je potrebné rátať minimálne s nasledujúcimi rozpočtovými položkami:

- náklady na nákup licencií;
- poplatok za správu a podporu systému na pravidelnej báze.

K týmto nákladom môžu ešte pribudnúť náklady na školenia užívateľov, upgrade softvéru a pod.

Pri SaaS sú základné funkcionality často krát poskytované zdarma. Pripláca sa za odomknutie ďalších funkcionalít a to buď na úrovni užívateľa, alebo inštitúcie.

3.3 Podpora užívateľa

Podpora užívateľa predstavuje komplexnú podporu a pomoc pri používaní systému a tiež pomoc pri riešení prevádzkových problémov. V prvom rade sa jedná o riešenie incidentov spojených s prihlasovaním sa do systému a so správnym fungovaním ponúkaných funkcionalít. Užívateľ by mal mať k dispozícii rozhranie na nahlasovanie takýchto incidentov a možnosť zistiť, v akom stave sa riešenie incidentu nachádza.

Medzi ďalšie štandardné typy podpory pre užívateľov patrí dokumentácia a FAQ.

Dokumentácia

Pod dokumentáciou rozumieme dokumentáciu, ktorej cieľom je uľahčiť prácu užívateľovi s novým systémom.[14] V prípade on-premises by mala obsahovať aj inštalačnú príručku. Užívateľská príručka, alebo manuál, býva najčastejšie zostavená z používateľských obrazoviek, ktoré sú zoradené podľa jednotlivých funkcionalít a logickej postupnosti. Obrazovky opisuje jednoduchý text.

[14] Iným typom je dokumentácia určená programátorom a jej úlohou je poskytovať informácie o technických riešeniach ako systému, tak jednotlivých funkcionalít.

Dokumentácia môže byť distribuovaná buď formou papierovej dokumentácie, alebo vo forme on-line príručky.

FAQ

Často kladené otázky (Frequetly Asked Questions) je v podstate forma dokumentácie, ktorá obsahuje zoznam otázok a odpovedí, ktoré užívatelia daného systému často kladú. V súčasnosti sú FAQ bežnou súčasťou internetových stránok, webových portálov a aplikácií. Sú veľmi užitočné, keďže jednoduchou a rýchlou formou zoznámia začiatočníkov so základným súborom pravidiel užívania. Môžu byť ďalej členené na rôzne oblasti (napr. všeobecné otázky, otázky týkajúcie sa ochrana súkromia a pod.), čo napomáha lepšej orientácii.

Ako príklad uvádzame FAQ nástroja na výučbu programovaia Scratch[15], ktorý si podrobnejšie predstavíme v kapitole 4. Analýza existujúcich systémov:

1. Čo je Scratch a ako ho môžem používať?
2. Ako môžem pomocou Scratch-u vyrobiť hru alebo animáciu?
3. Môžem video vyrobené pomocou Scratch-u umiestniť na YouTube?

Blog

Blog je pomerne nová a zatiaľ málo využívaná forma podpory užívateľov. Podľa definície predstavuje blog diskusnú alebo informačnú stránku umiestntnú na internete, ktorá pozostáva zo samostatných článkov (v blogerskej komunite nazývané „posts“), umiestnených v chronologickom poradí.[16] Blog nemá žiadnu predpísanú formu a jeho obsah sa líši podľa cieľovej skupiny, na ktorú sa chce autor zamerať.

V prípade edukačného softvéru predstavujú cieľovú skupinu užívatelia, tzn. v prevažnej miere pedagogickí pracovníci. Pre túto cieľovú skupinu predstavuje blog možnosť sprostredkovať im najnovšie poznatky z oblasti vzdelávania, príkladov dobrej praxe, plánovaných funkcionalít a pod.

[15] https://scratch.mit.edu/help/faq/, [cit. 2015-04-05].
[16] http://en.wikipedia.org/wiki/Blog, [cit. 2015-04-05].

3.4 Lokalizácia

Lokalizácia softvéru je proces, v rámci ktorého sa daný softvér prispôsobuje okrem jazyka prispôsobuje aj kultúrnym alebo sociálnym zvykom cieľovej krajiny a skupiny. Nejedná sa teda len o obyčajný jazykový preklad, proces lokalizácie zastrešuje aj prispôsobenie používaných prvkov, ako sú:

- písmo,
- kalendár,
- formát dátumu a času,
- formát čísel,
- symboly
- sprievodné videá apod.

Preklad musí byť jazykovo správny a samozrejme aj terminologicky konzistentný.

Samotný proces lokalizácie softvéru pozostáva z úpravy:

- užívateľského rozhrania softvéru – tu je nevyhnutná súčinnosť programátora, keďže sa jedná o úpravu ovládacích prvkov aplikácie, úpravu formátov dátumu a času apod. V prípade, že sa jedná o on-premises softvér, sa v konečnej fáze pripravuje aj inštalačný balíček,
- elektronickej dokumentácie – náročnosť úpravy elektronickej dokumentácie závisí od jej obsahu. V prípade, že obsahuje okrem textu aj multimediálny obsah, ako sú napr. prezenácie, videá, obrázkové návody a pod., si úprava elektronickej dokumentácie vyžaduje okrem práce profesionálneho prekladateľa aj prácu grafikov.
- všetkých tlačených súčastí.

Ako môžeme vidieť, lokalizácia softvéru pozostáva z úloh pre tím prekladateľov, programátorov, grafikov, prípadne iných špecialistov. Je to náročný proces ako po pracovnej, tak po finančnej stránke.

Pre potreby českých škôl je žiaduce, aby bol vybraný edukačný softvér k dispozícii v českom jazyku a s miestnymi reáliami. Mnoho vynikajúcich softvérov je však, ako budeme vidieť v kapitole 4. Analýza existujúcich systémov, k dispozícii iba v anglickom jazyku. Lokalizácia je, ako sme uviedli vyššie, finančne aj manažérsky náročný proces. Okrem samotnej lokalizácie je potrebné získať aj práva od poskytovateľov služby. Otázkou teda

je, či stojí za to pustiť sa do lokalizácie existujúceho systému, alebo navrhnúť vlastný, ktorý bude od začiatku vytváraný s ohľadom na cieľovú krajinu a skupinu.

4 ANALÝZA EXISTUJÚCICH SYSTÉMOV

Analýza existujúcich systémov je dôležitým krokom v príprave špecifikácie navrhovaného systému. Na jednej strane nám pomôže pochopiť, ako tieto systémy fungujú, čo užívateľom ponúkajú a čo je žiadané zo strany užívateľov, na strane druhej sa vďaka nej dozvieme, čo naopak na trhu chýba a čím by nový nástroj mohol prispieť.

Na existujúce systémy sme sa pozerali cez prizmu zistených skutočností v predchádzajúcich kapitolách. Samotná analýza je preto rozdelená na niekoľko častí, ktorými sú:

- požiadavky na vybavenie;
- spôsoby obstarania a náklady;
- dostupná podpora;
- lokalizácia;
- funkčná analýza programu;
- využitie herných princípov.

Medzi prílohami bakalárskej práce uvádzame ako Prílohu P I: Prehľad analyzovaných edukačných systémov, kde sú zistené skutočnosti zobrazené v prehľadnej tabuľke. Ako môžeme vidieť, ani jeden z analyzovaných systémov nespĺňa naše požiadavky. Najčastejším dôvodom je chýbajúca lokalizácia do českého jazyka a zložitý prechod od vizuálneho k textovému vývojovému prostrediu.

4.1 Hakitzu[17]

Hra Hakitzu pochádza z dielne londýnskeho developerského štúdia, zameraného na tvorbu hier.

Požiadavky na vybavenie

Hra je dostupná pre zariadenia s iOS a OS Android, teda pre smartfóny a tablety.

Spôsoby obstarania a náklady

Je možné ju stiahnuť zdarma na App Store alebo Google play.

[17] http://www.kuatostudios.com/games/hakitzu-elite/, [cit. 2015-04-05].

Dostupná podpora

Pre učiteľov je k dispozícii „Teacher's Pack"[18], ktorý predstavuje stručný úvod do prostredia Hakitzu a zároveň ponúka návody, akým spôsobom je možné Hakitzu využívať vo vyučovaní.

Manuály su k dispozícii na stiahnutie na cloudovom úložisku (za týmto účelom využívajú Dropbox).

Stránka zároveň obsahuje blog a odkazy na Facebook[19], Twitter[20] a YouTube[21].

Lokalizácia

Hra, ako aj stránka a dokumentácia, sú k dispozícii len v anglickom jazyku.

Funkčná analýza programu

Hakitzu je strategická hra. Jej cieľom je zoznámiť žiakov so základmi JavaScriptu a základmi programovania.

Hra je určená pre dvoch hráčov. Po výbere „Multiplayer" v hlavnom menu môže hráč pozvať do hry spoluhráča pomocou mailu alebo Facebooku. Hráč má tiež možnosť hrať s náhodným spoluhráčom.

Hra vychádza zo šachu. Hráči začínajú hru na opačných koncoch hracej dosky s programovateľným robotom a s objektom, nazývaným „Power Core". Cieľom hry je zničiť „Power Core" spoluhráča pomocou programovateľného robota. Tomu, komu sa to podarí ako prvému, vyhráva.

Hráč má možnosť postupne dokupovať výbavu pre svojho robota. Nákupnou menou sú kredity, ktoré postupne získava na základe úspechov dosiahnutých v hre.

Hra je rozdelená na štyri stupne obtiažnosti, a to začiatočník, junior programátor, programátor a hacker. Pre postup na ďalší stupeň musí hráč získať potrebný počet bodov. Začiatočník začína „programovať" pomocou ikon, to znamená, že sa od neho nevyžaduje, aby písal kód. Ten sa mu generuje automaticky. Od stupňa obtiažnosti junior programátor hráč

[18] https://www.dropbox.com/s/doj9fug23eymsgf/Hakitzu%20Teachers%20Guide.pdf, [cit. 2015-04-05].
[19] www.facebook.com/Hakitzu [cit. 2015-04-05].
[20] www.twitter.com/kuatostudios, [cit. 2015-04-05].
[21] www.youtube.com/kuatostudios, [cit. 2015-04-05].

pohybuje robotom prostredníctvom písaného kódu s tým, že na začiatku má možnosť využívať Auto-Complate systém. Napr. po napísaní písmena „m“ mu systém automaticky doplní „move“. Táto možnosť sa mu postupne redukuje a na vyšších stupňoch obtiažnosti už nie je prístupná vôbec. Hráč má tiež k dispozícii funkcionalitu, ktorá mu vyznačuje syntaktické chybu a ponúka mu nápovedu, ako chyby odstrániť.

Po vizuálnej stránke je hra spracovaná na vysokej úrovni.

Využitie herných princípov

Okrem zbierania bodov a postupovania na vyššie stupne obtiažnosti má hráč možnosť plniť výzvy, v rámci ktorých si testuje svoje programátorské zručnosti a znalosti.

Obrázok 6: Hakitzu - náhľad bojovej arény[22]

[22] Zdroj: http://292fc373eb1b8428f75b-7f75e5eb51943043279413a54aaa858a.r38.cf3.rackcdn.com/technology_08_0_temp-1373969143-51e51af7-620x348.jpg, [cit. 2015-04-05].

Obrázok 7: Hakitzu – strategický náhľad[23]

4.2 Lego Mindstorms[24]

Lego Mindstorms je jedným z výrobkom, ktoré poskytuje firma Lego.

Spôsoby obstarania a náklady

V súčasnosti je na trhu verzia Lebo Mindstorms EV3. Základný balíček obsahuje okrem stavebnicového systému Lego aj:

- programovateľný blok EV3 – ten obsahuje 4 vstupné porty, 4 výstupné porty, 1 Mini USB PC port, USB hostiteľský port, port mikrto SD karty a vstavaný repto-ruktor;
- senzory;

[23] Zdroj: http://i.ytimg.com/vi/pWFhIPExztQ/maxresdefault.jpg, [cit. 2015-04-05].
[24] http://www.lego.com/en-us/mindstorms/?domainredir=mindstorms.lego.com, [cit. 2015-04-05].

- motory;
- ďalšie príslušenstvo.

Obrázok 8: Sada Lebo Mindstorms[25]

Zo sady Lego Mindstorms je možné zostaviť rôznych robotov, ktoré sa dajú pomocou špeciálneho softvéru naprogramovať tak, aby chodili, vyhýbali sa prekážkam, hovorili a pod.

Výhradným dodávateľom pre Českú republiku je firma Eduxe s.r.o. Základná súprava je v ponuke za cenu 10 803 Kč s DPH. Za doplnkovú sadu je potrebné si priplatiť ďalších 3 180 Kč s DPH.

Na programovanie robotov slúži softvér EV3. V prípade inštitúcie, ktorá chce inštalovať program na viacerých počítačoch, je potrebné zakúpiť si licenciu. Licencia, ktorá umožňuje inštalovať softvér na neobmedzený počet počítačov, stojí 11 977 Kč s DPH.

Požiadavky na vybavenie

Softvér EV3 je dostupný pre OS X aj OS Windows.

---

[25] Zdroj: http://cache.lego.com/r/www/r/mindstorms/-/media/franchises/mindstorms%202014/products/in%20the%20box/inthebox_bricks_landscape.jpg?l.r=-1550870471, [cit. 2015-04-05].

Systémové požiadavky pre PC s OS Windows

Windows XP (32 bitový) a Vista (32/64 bitový) s výnimkou Starter Edition - s najnovšími aktualizáciami. Windows 7 (32/64 bitový) a Windows 8 desktop vrátane Starter Edition - s najnovšími aktualizáciami.

Ďalšie požiadavky: dual core processor 2.0 GHz alebo viac, pamäť RAM minimálne 2GB, 2GB miesta na hard disku, displej XGA (1024 x 768 pixelov), 1 voľný USB port.

Systémové požiadavky pre Apple Macintosh s OS X

Mac OS 10.6, 10.7 a 10.8 s najnovšími aktualizáciami.

Ďalšie požiadavky: dual core processor 2.0 GHz alebo viac, pamäť RAM minimálne 2GB, 2GB miesta na hard disku, displej XGA (1024 x 768 pixelov), 1 voľný USB port.

Dostupná podpora

Zákaznícky servis je aj pre Českú republiku na stránke k dispozícii iba v anglickom jazyku. Poskytuje napr. možnosť kontaktovať výrobcu formou mailu alebo telefonicky, ďalej obsahuje FAQ a objednávkový formulár. Otázky, resp. požiadavky, je možné na uvedené kontaktné údaje zasielať aj v českom jazyku. Preklik na zákaznícky servis presmeruje užívateľa na stránku Lego, čo pôsobí mätúco.

Lokalizácia

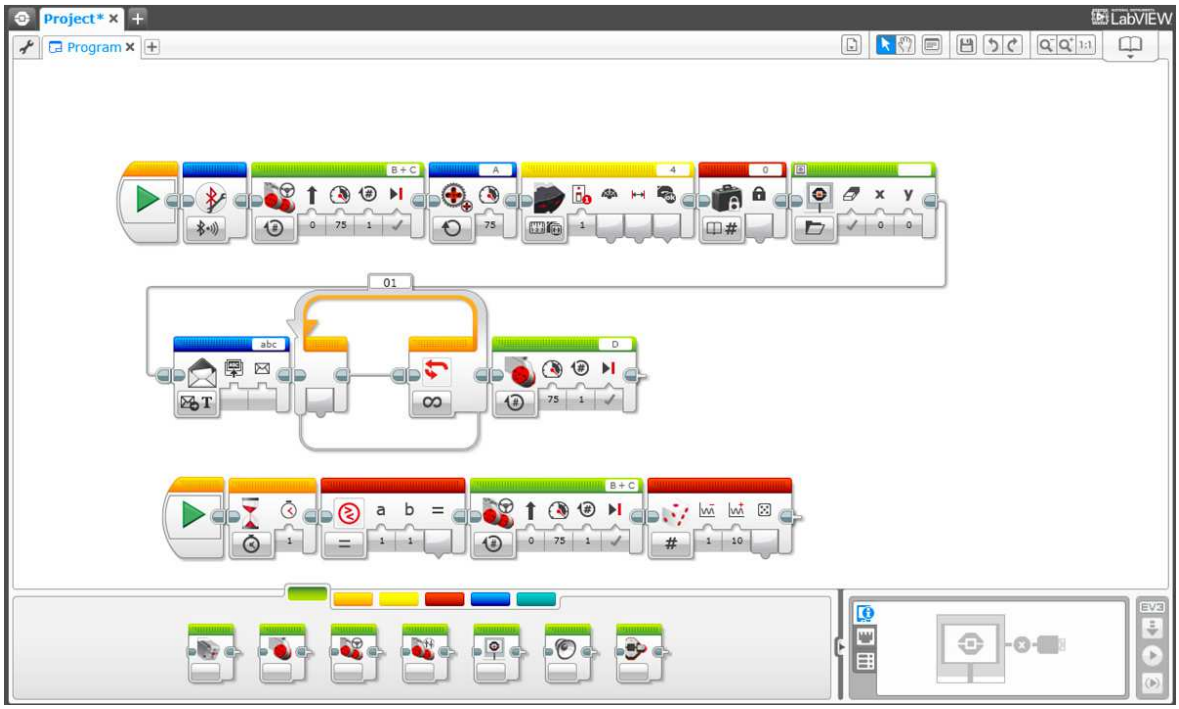
Stránka, užívateľská príručka ako aj ďalšie návody sú k dispozícii aj v českom jazyku.

Videá sú k dispozícii v anglickom jazyku.

Funkčná analýza programu

Pre ľahkú orientáciu v programe slúži Úvodná obrazovka, ktorá obsahuje možnosť pridať projekt, misia stavby robota, otvoriť naposledy spustené, rýchle spustenie, novinky a ďalší roboti (možnosť stiahnuť si ďalšie misie stavba robota z internetu).

Samotné programovanie sa realizuje prostredníctvom intuitívneho ikonového programovacieho prostredia.



Obrázok 9: Užívateľské rozhranie softvéru EV3[26]

Komunikácia s robotom prebieha prostredníctvom programovateľnej kocky EV3. Táto komunikácia sa riadi prostredníctvom stránky hardvéru, kde je tiež možné pozrieť sa, ktoré motory a senzory robota sú zapojené, a na ktorom mieste.

Okrem toho sú k dispozície aplikácie pre múdre zariadenia, vďaka ktorým je možné robota ovládať aj pomocou telefónu, alebo tabletu.

Využitie herných princípov

Užívateľ v prípade Lego Mindstorms neinteraguje s ostatnými užívateľmi, ani nezbiera body, prípadne odznaky.

Samotných robotov je však možné použiť na plnenie rôznych výziev[27].

[26] Zdroj: http://the-gadgeteer.com/wp-content/uploads/2014/06/mindstorm_ev3-programming.jpg [cit. 2015-04-05].
[27] Oficiálna stránka súťaže First Lego League v ČR: http://www.ceskaligarobotiky.cz/item/22-first-lego-league. [cit. 2015-04-05].

4.3 Scratch[28]

Scratch je projektom Lifelong Kindergarten Group, ktorá pôsobí v rámci MIT Media Lab. Umožňuje programovať interaktívne príbehy, hry a animácie a zdieľať ich v rámci komunity.

Spôsoby obstarania a náklady

Scratch je momentálne dostupný v dvoch, resp. troch verziách, všetky je možné používať zdarma.

K dispozícii je stále staršia verzia 1.4. Túto verziu je potrebné stiahnuť a nainštalovať priamo do počítača.

Nová verzia 2.0 je k dispozícii online, čiže ako SaaS, a zároveň ako offline editor (zatiaľ beta verzia).

Požiadavky na vybavenie

Verzia 1.4. je kompatibilná s Mac OSX 10.4 a vyššie, Windows 2000, XP, Vista, 7 a 8 a tiež s Ubuntu 12.04 a vyššie. Projekty vytvorené vo verzii 2.0 nie je možné otvoriť vo verzii 1.4.

Pre online verziu 2.0 je potrebné mať aktualizované verzie prehliadača Chrome, Firefox a Internet Explorer s Adobe Flash Player verzie 10.2 alebo vyššie. Offline verzia je k dipozícii pre Mac OS X, Mac OS X 10.5 a staršie, Windows a Linux. S povolením užívateľa sa všetky aktualizácie sťahujú automaticky.

Dostupná podpora

Stránka má veľmi dobre prepracovaný systém podpory.

Okrem samostatnej položky Help v hlavnom menu, kde užívateľ nájde návody, sprievodné videá a FAQ, umožňuje stránke aj diskusiu a nahlasovanie chýb.

Napriek tomu, že stránku je možné prepnúť do českého jazyka, väčšina uvedených materiálov je dostupných iba v anglickom jazyku.

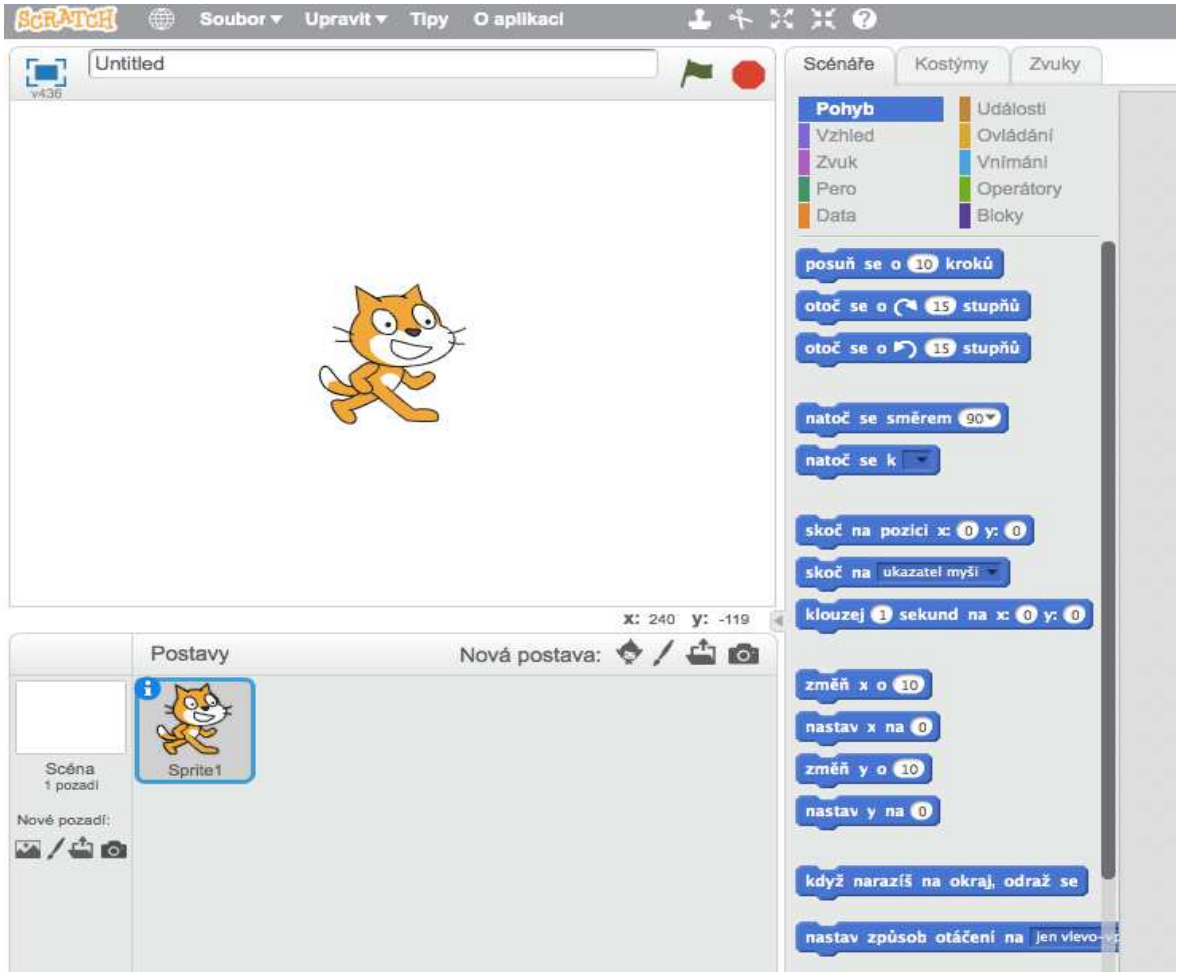
[28] https://scratch.mit.edu/ [cit. 2015-04-05].

Lokalizácia

Všetky verzie softvéru a stránka sú dostupná v českom jazyku. Väčšina dokumentácie je však k dispozícii len v anglickom jazyku.

Funkčná analýza programu

Scratch je jednoduché programovacie prostredie, určené na výučbu základov programovania. Je intuitívne a jednoducho sa ovláda, preto je vhodné aj pre nižšie ročníky. Programuje sa výhradne pomocou blokov príkazov, ktoré do seba zapadajú podobne ako puzzle.



Obrázok 10: Vývojové prostredia programu Scratch

Využitie herných princípov

Scratch priamo neponúka žiadne gamifikačné prvky.

Užívateľ má však možnosť umiestniť vytvorené projekty na stránku, kde môžu ostatní členovia komunity projekt komentovať, pridať si ho medzi obľúbené, alebo ho označiť, že sa im páči.

4.4 Tynker[29]

Spôsoby obstarania a náklady

Pre vzdelávacie inštitúcie je Tynker plateným riešením. Základná verzia je síce zdarma, ale pokiaľ chce učiteľ využívať všetky možnosti, ktoré Tynker ponúka, sú na výber dve možnosti. Prvá možnosť Classroom je k dispozícii za 399 dolárov a obsahuje kurikulum pre jeden vzdelávací stupeň. Druhá možnosť School je určená pre celú školu a nákupná cena je od 2 000 dolárov.

Aplikácie dostupné na App Store a Google play sú taktiež platené. Aplikácia určená pre Android je dostupná za 3,67 eur, pre iOS od 1,99 dolára.

Požiadavky na vybavenie

Tynker je online program, postavený na webových štandardoch (HTML5, JavaScript, CSS) a je teda dostupný na všetkých aktuálnych webových prehliadačoch.

Tiež je k dispozícii vo forme aplikácie pre Android od verzie 4.0 vyššie a pre iOS od verzie 6.0 vyššie.

Dostupná podpora

Pre učiteľov obsahuje stránke kompletne vypracované kurikulum rozdelené podľa jednak podľa ročníkov, respektívne stupňov a jednak podľa úrovní začiatočník, stredne pokročilý a pokročilý. Tiež obsahuje prehľad o tom, ktoré kurzy rozvíjajú ďalšie oblasti v rámci medzipredmetových vzťahov.

Na stánke je v pravej časti stále vidiľné tlačidlo Feedback, vďaka ktorému je možné poslať Tynker tímu spätnú väzbu.

Stránka zároveň obsahuje blog, odkazy na Facebook[30], Twitter[31], YouTube[32] a Google+[33] a možnosť prihlásiť sa na odber noviniek.

---

[29] https://www.tynker.com/?t=reset, [cit. 2015-04-05].

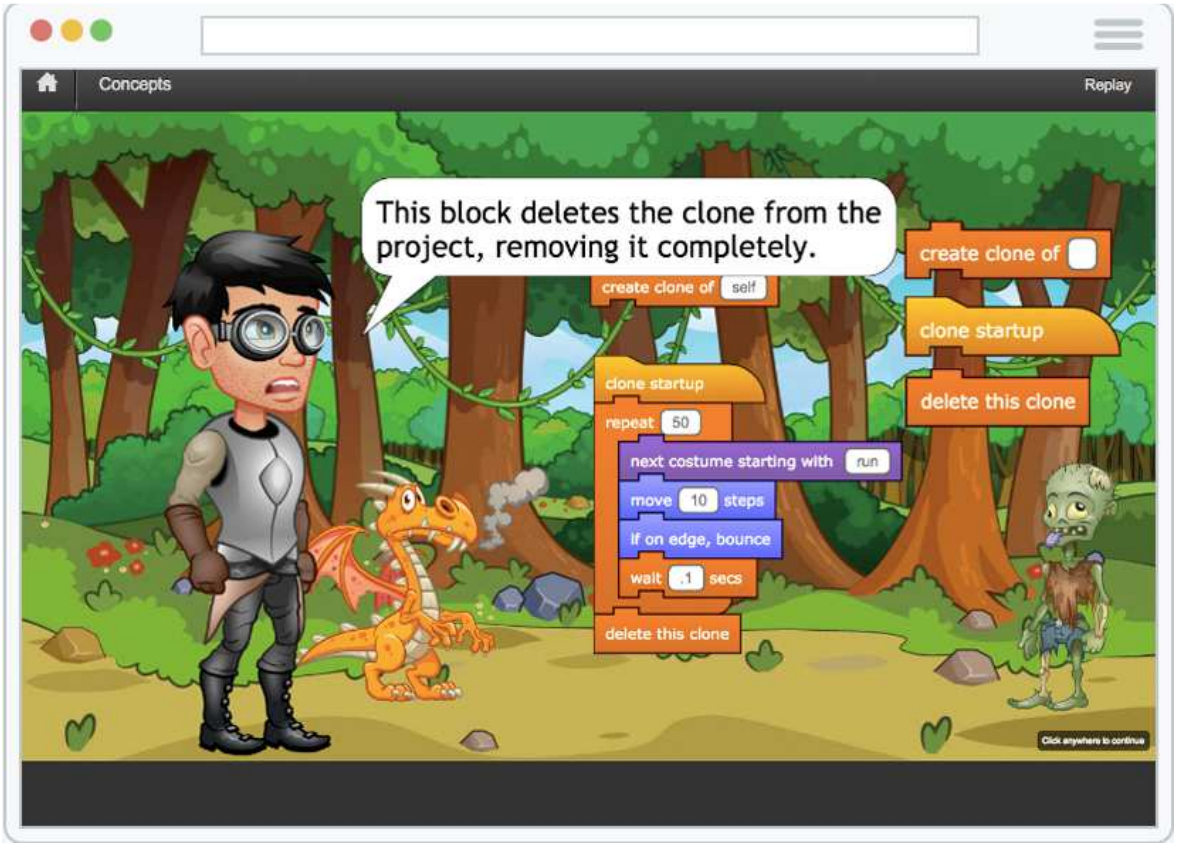
Lokalizácia

Hra, ako aj stránka a dokumentácia, sú k dispozícii len v anglickom jazyku.

Funkčná analýza programu

Tynker je online dostupné prostredie, v rámci ktorého sa žiaci pomocou skladania vizuálnych blokov učia základné princípy programovania.

Kurzy sú rozdelené do modulov, každý modul má stanovený cieľ, čo sa má žiak počas neho naučiť. V rámci modulov sa nachádzajú rôzne úlohy. Na konci modulu je k dispozícii kvíz.

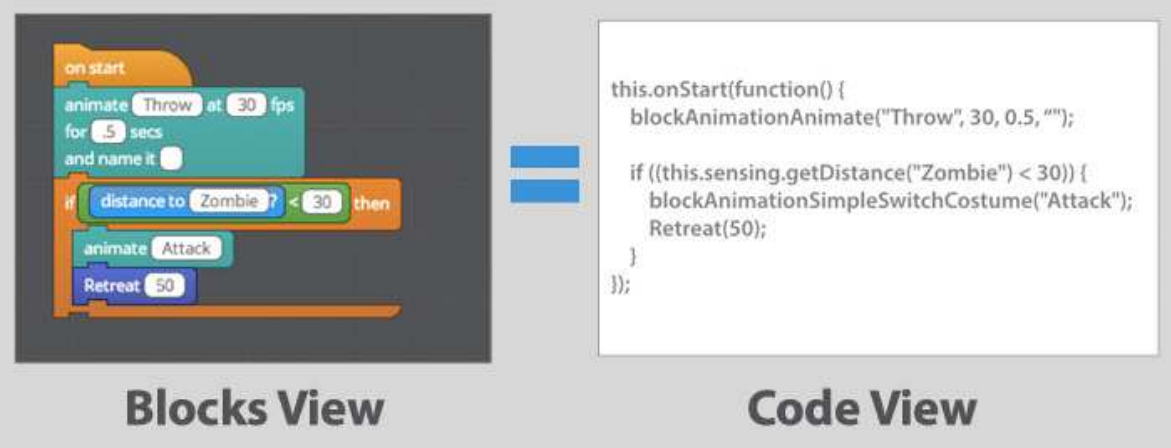


Obrázok 11: Tynker – vývojové prostredie[34]

[30] https://www.facebook.com/GoTynker, [cit. 2015-04-05].
[31] https://twitter.com/gotynker, [cit. 2015-04-05].
[32] https://www.youtube.com/channel/UC2MAKe5X7pohhiMZ4nzdInA?sub_confirmation=1 [cit. 2015-04-05].
[33] https://plus.google.com/111189514752913165927[cit. 2015-04-05].
[34] Zdroj: https://www.tynker.com/image/product/student/img-student-direction-01.jpg[cit. 2015-04-05].

Po zakúpení licencie pre školy získava učiteľ možnosť využívať funkcionality, vďaka ktorým dokáže manažovať vzdelávanie žiakov.

Tynker je jedným z mála edukačných nástrojov, ktorý umožňuje plynulý prechod od vizuálneho programovania, resp. programovania prostredníctvom blokov, k programovaniu pomocou príkazového riadku. Tynker to umožňuje prepínaním sa medzi Code View a Block View. V prípade, že si užívateľ zvolí možnosť Code view, môže vidieť obe okná súčasne. Najprv tak môže iba sledovať, ako sa kód pridávaním blokov mení, neskôr môže začať sám zasahovať do kódu, čo naopak mení Block View. Vďaka tomu vie dieťa ľahko zistiť, keď urobí chybu v kóde a pomocou už známych blokov ju identifikovať a opraviť.



Obrázok 12: Tynker - Blocks View vs. Code View[35]

Využitie herných princípov

Za postup do ďalších úrovní ja možné získať odznaky.

Obrázok 13: Tynker - odznaky za postup na vyššie úrovne[36]

[35] http://images.tynker.com/blog/wp-content/uploads/code-view.jpg [cit. 2015-04-05].
[36] Zdroj: https://www.tynker.com/image/school/features/badges.png [cit. 2015-04-05].

4.5 Alice[37, 38]

Spôsoby obstarania a náklady

Program Alice je dostupný zdarma. Ako je uvedené na stránke, je to darček od Carnegie Mellon University. Súbory na stiahnutie sú k dispozícii na domovskej stránke programu.

Požiadavky na vybavenie

Momentálne je dostupné vydanie Alice 3. Softvérové požiadavky pre túto verziu sú nasledovné:

- Windows (XP, Vista 32-bit, Vista 64-bit, Windows 7 32-bit, Windows 7 64-bit, Windows 8 32-bit, Windows 8 64-bit)
- Mac OS X 10.6 a vyššie, procesor Intel
- Ubuntu, Red Hat – minimálne 250MB voľného miesta, OpenGL ovládače a Java Runtime Environment.

Alice 3 využíva Java JDK. Pokiaľ Java JDK nie je k dispozícii, je potrebné ju nainštalovať pred inštaláciou Alice 3.

Hardvérové požiadavky:

- stolný počítač alebo notebook;
- 1 GB RAM (odporúčaná veľkosť pamäte je 2 GB alebo viac, ale nie je to podmienkou);
  grafická karta schopná vysokého rozlíšenia farieb (32 bitov) a s rozlíšením aspoň 1024x768 pixelov (3D grafická karta poskytuje vyšší výkon, ale nie je nutná);
- zvuková karta;
- myš.

Dostupná podpora

Alice poskytuje svojim užívateľom veľmi dobrú podporu. Pre učiteľov sú k dispozícii pracovné materiály a tutoriály a samozrejme FAQ. Pre správcov a programátorov sú to inštalačné návody a zoznam najčastejších chýb spolu s návodom, ako je možné ich riešiť.

---

[37] http://www.alice.org/index.php [cit. 2015-04-05].
[38] Okrem Alice existuje veľmi podobný nástroj na tvorbu 3D animácii Looking Glass z dielne Washington University in St. Louis: https://lookingglass.wustl.edu/ [cit. 2015-04-05].

Okrem toho je na stránke uvedený zoznam literatúry, ktorá sa venuje výučbe programovania pomocou Alice.

Lokalizácia

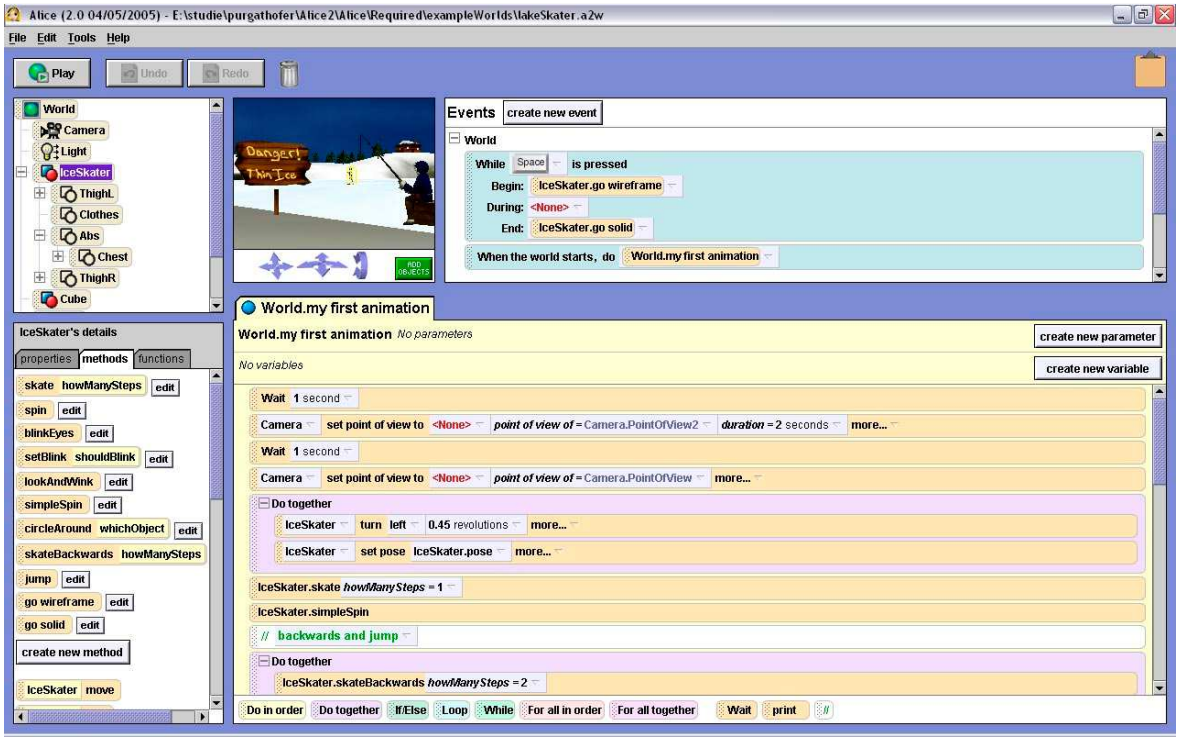
Hra, ako aj stránka a dokumentácia, sú k dispozícii len v anglickom jazyku.

Funkčná analýza programu

Alice je 3D programovacie prostredie určené na výučbu základov objektovo orientovaného programovania, ktoré umožňuje vytváranie vlastných príbehov prostredníctvom animácií, hranie interaktívnych hier, alebo vytváranie videí, ktoré je možné zdieľať na webe.

V interaktívnom rozhraním žiaci vytvárajú program pomocou umiestňovania grafických blokov. Inštrukcie zodpovedajú štandardným funkciám v OOP, ako napr. v Jave, C++ alebo v C#. Alice vizualizáciou kódu umožňuje žiakom pochopiť vzťah medzi ním a správaním sa objektu v animácii.

Alice tiež umožňuje prechod od vizuálneho k textovému programovaniu. Užívateľ si musí ale najprv stiahnuť a nainštalovať NetBeans IDE a následne si tam program vytvorený v Alice importovať (na stránke sa nachádza podrobný video návod). Prechod od vizuálneho k textovému programovaniu sa teda nedeje v už známom rozhraní.



Obrázok 14: Vývojové prostredie programu Alice[39]

Využitie herných princípov

Alice nevyužíva žiadne gamifikačné prvky.

4.6 Karel

Spôsoby obstarania a náklady

Robot Karel je k dispozícii zdarma. Zdroje na inštaláciu je však potrebné hľadať a inštalácia nie je práve priateľská k užívateľom.

Požiadavky na vybavenie

Počítač s OS Windows, prípadne s niektorou linuxovou distribúciou, napr. Ubuntu.

Niektoré implementácie sú dostupné aj online, prípadne ako aplikácia pre iPad. Aplikácia je dostupná na Apple Store za cenu 1,99 dolára. Požadovaná verzia iOS je minimálne 6.0.

[39] http://upload.wikimedia.org/wikipedia/commons/5/57/Alice-2-screenshot.jpg, [cit. 2015-04-05].

Dostupná podpora

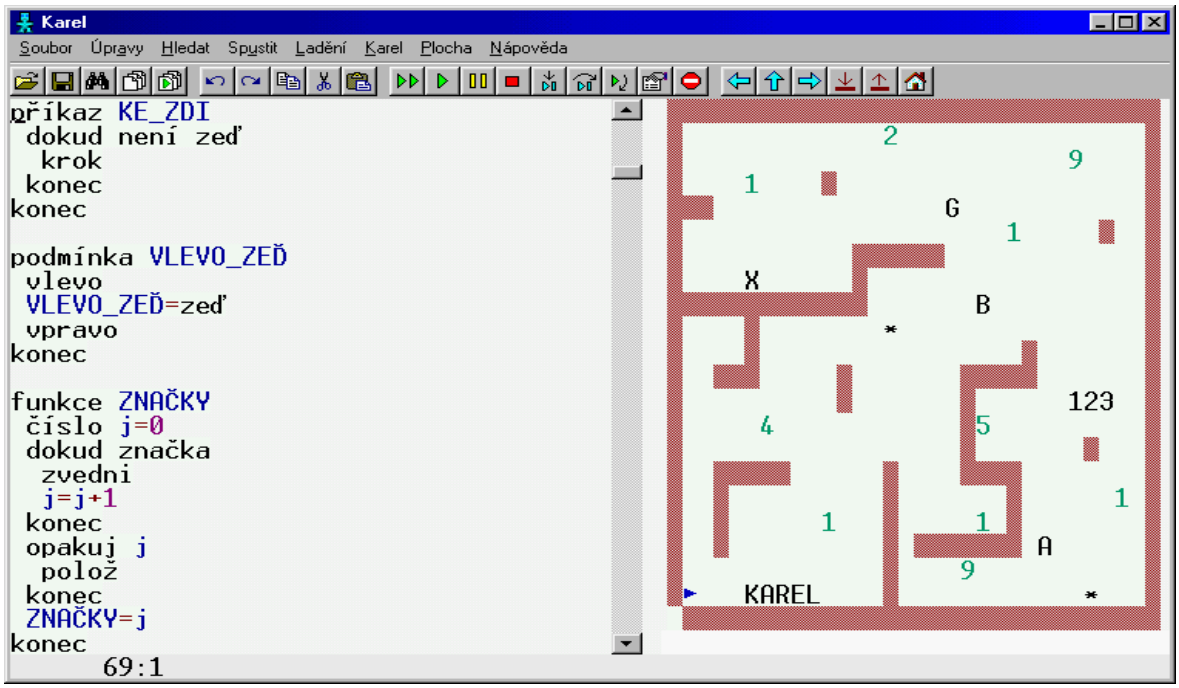
Keďže sa jedná o mnoho implementácii jednej myšlienky, je ťažké hovoriť o dostupnej forme podpory. Existuje mnoho stránok, ktoré sa venujú problematike výučby v tomto prostredí a ponúkajú rôzne manuály aj v českom jazyku.

Lokalizácia

Prostredie, ako aj dokumentácia, sú k dispozícii aj v českom jazyku.

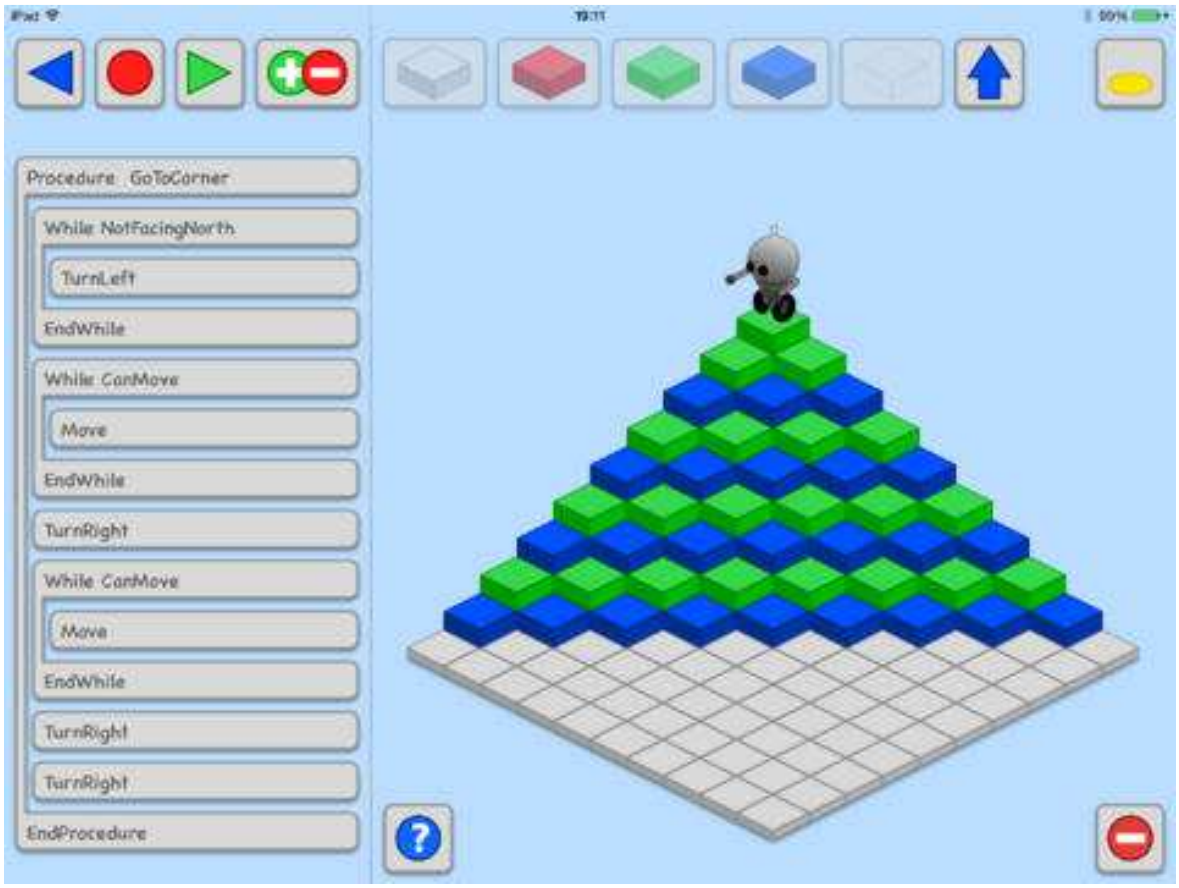
Funkčná analýza programu

S myšlienkou Robota Karla prišiel na prelome 70. a 80. rokov Richard E. Pattis, ktorý ho používal pri výučbe základov programovania na Stanfordovej univerzite. Podstatou je snaha o ovládanie robota, ktorý sa pohybuje v rámci stanoveného priestoru, ktorý predstavuje mesto. Robot ovláda malý počet jednoduchých príkazov, a to krok, vľavo bok, položiť a zdvihnúť značku. Všetky, aj zložitejšie povely, sa vytvárajú kombináciou týchto príkazov. Jazyk podporuje základné algoritmické štruktúry, ako je blok príkazov, podmienky a cykly, je tiež možné použiť princíp rekurzie.



Obrázok 15: Vývojové prostredie pre program Robot Karel[40]

[40] Zdroj: http://petr.lastovicka.sweb.cz/img/karelWin.png, [cit. 2015-04-05].



Obrázok 16: Vývojové prostredie pre program Robot Karel - iPad verzia[41]

Využitie herných princípov

Karel nevyužíva žiadne gamifikačné prvky.

4.7 Baltík[42]

Spôsoby obstarania a náklady

Baltíka vyvíja aj priamo predáva firma SGP. Na ich stránke je možné stiahnuť a nainštalovať si demo verziu zdarma. Takáto verzia programu je plne funkčná, neumožňuje však ukladanie súborov na disk.

[41] Zdroj: http://a1.mzstatic.com/us/r30/Purple/v4/f7/d7/72/f7d772ed-5e74-68c5-5c74-9b0fd734c1b0/screen480x480.jpeg, [cit. 2015-04-05].
[42] http://www.sgpsys.com/cz/Default.asp, [cit. 2015-04-05].

Firma ponúka tri verzie a to Baltík 3, ktorý obsahuje programovacie a kresliace nástroje – ikony (syntax jazyka C), Baltie 4 C#, ktorý obsahuje programovacie a kresliace nástroje – ikony, C#, 3D, OOP, objektový browser a debugger a hru Gems, ktorá je naprogramovaná v Baltie 4 C#.

Podľa cenníka, uvedeného na stránke, je možno objednať každú z týchto verzií zvlášť. Ročná multilicencia SGP, ktorá obsahuje aj Baltík 3, aj Baltie 4 C#, je pre celú školu aj žiakov domov k dispozícii za 300 eur.

Výrobca odporúča programu zakúpiť aj učebnicu programovania Baltík 3, ktorá je dostupná za 10 eur.

Požiadavky na vybavenie

Program je potrebné nainštalovať na každý počítač alebo na server (je potrebné vlastniť sieťovú licenciu Baltíka). Nie sú uvádzané žiadne špeciálne požiadavky na hardvérové ani softvérové vybavenie. Program však nie je k dispozícii pre počítače s OS X.

Dostupná podpora

Podpora je pomerne rozsiahla. Pre učiteľov sú k dispozícii metodiky výučby, pre správcov návody na inštaláciu.

Lokalizácia

Stránka, dokumentácia aj program sú k dispozícii v českom jazyku.

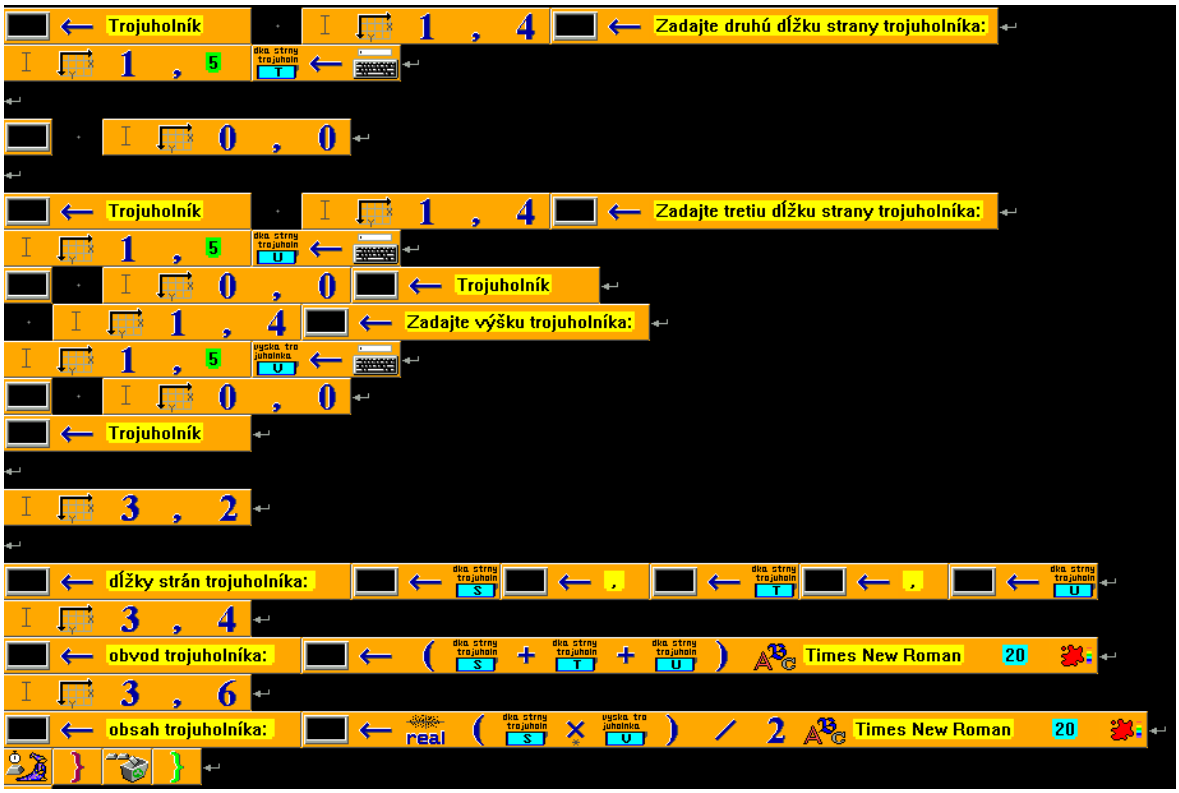
Funkčná analýza programu

Baltík je multimediálny výučbový programovací a kresliaci nástroj, v ktorom je možné tvoriť prezentácie, výučbové programy a hry. Baltík využíva štandardnú syntax neobjektových programovacích jazykov, ako napr. C, tzn. že obsahuje príkazy ako if, for, while, do-while atď., je možné používať premenné, konštanty, polia dátových typov, procedúry a funkcie. Všetky príkazy sú však namiesto textu zobrazené vo forme ikon.

Program má tri základné úrovne:

1. Skladanie scény – žiak sa učí pracovať s myšou a klávesnicou a zoznamuje sa s pracovnou plochou. Môže si skladať scény alebo kresliť nové predmety
2. Čarovanie scény – žiak ovláda postavičku čarodejníka Baltíka, dáva mu príkazy a ten ich vykonáva. Úlohou tejto úrovne je zoznámiť sa s procesom algoritmizácie

3. Programovanie – táto úroveň je ešte rozdelená na časť pre začiatočníkov a pokročilých.



Obrázok 17: Ukážka zdrojového kódu programu Baltík 3[43]

Program je vizuálne nezaujímavý a programovanie pomocou ikon pomerne neintuitívne. Je ťažké identifikovať, čo ktorý symbol znamená. Pre žiakov nižších ročníkov je čítanie nápovedy kontraproduktívne a môže ich ľahko odradiť.

Využitie herných princípov

Baltík nevyužíva žiadne gamifikačné prvky.

4.8 Imagine Logo[44]

Imagine Logo je nová generácia programovacieho jazyka a prostredia Logo.

Spôsoby obstarania a náklady

[43] http://www.flatulent.szm.com/baltik/obr/07_a_3.png, [cit. 2015-04-05].
[44] http://www.pf.jcu.cz/imagine/index.php, [cit. 2015-04-05].

Imagine v českej verzii je možné zakúpiť v nakladateľstve Computer Press. Neobmedzená školská licencia je k dispozícii za 7 616 Kč. Neobmedzená školská licencia je určená školám a umožňuje inštalovať Imagine Logo na ľubovoľnom počte školských počítačov. Učitelia a žiaci môžu program inštalovať aj na počítačoch mimo školy pre svoju osobnú potrebu.

K programu bolo možné dokúpiť aj učebnicu Imagine Logo – učebnice programování pro děti. Kniha vyšla v Computer Press v roku 2006, a momentálne už nie je dostupná.

Požiadavky na vybavenie

Program je určený pre OS Windows.

Ďalšie požadavky: minimálna konfigurácia PC - Procesor Pentium II 400 MHz a rýchlejší, 128 MB RAM, 60 MB voľného priestoru na disku, CD mechanika, myš, zvuková karta.

Dostupná podpora

Existujú rôzne stránky, na ktorých je možné nájsť didaktické materiály, príklady dobrej praxe, prípadne ukážkové projekty.

Priama podpora užívateľov ale nie je dostupná.

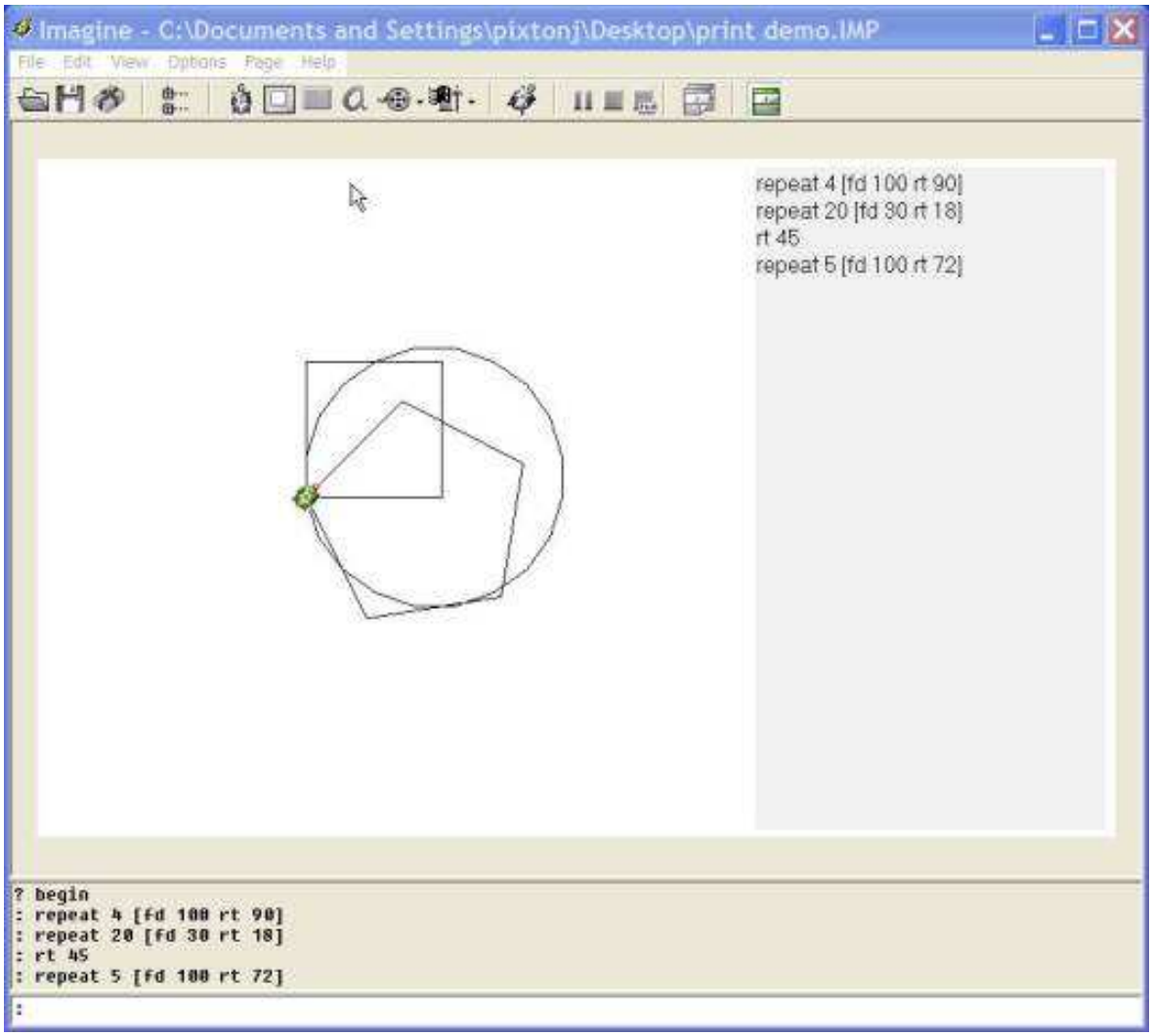
Lokalizácia

Program, prípadne ďalšie materiály, sú k dispozícii v českom jazyku.

Funkčná analýza programu

Prostredie Imagine Logo spoločne s modulom Logo Motion je určené na osvojenie si základov procedurálneho programovania a programovanie riadeného udalosťami a prácu v rastrovom grafickom editore. Tiež podporuje multimédiá a možnosť prezentácie vytvorených projektov na webe.

Základnou súčasťou programu Imagine Logo je grafické prostredie, v ktorom sa pohybuje korytnačka. Pomocou jednoduchých príkazov môžeme korytnačkou manipulovať, pričom po jej pohybe zostáva na ploche stopa. Základné príkazy sú v češtine dopředu (do) a vzad (vz) a pohybujú korytnačkou o daný počet krokov určeným smerom, pomocou príkazov vpravo (vp) a vlevo (vl) môžeme zase korytnačkou otáčať. Medzi základé príkazy ešte patrí barvapera! (Bp!) - Tloušťkapera! (Tp!) - Peronahoru (pn) - perodolů (pd) – zmaž, vyplň a domov.



Obrázok 18: Ukážka vývojového prostredia Imagine Logo[45]

Prostredie obsahuje ďalšie objekty, ktoré rozširujú možnosti programovania, ako sú ďalšie korytnačky, textové polia alebo prehrávače zvukových súborov a video súborov. Prostredie sa ovláda pomocou textových príkazov písaných do príkazového riadku, písaním procedúr alebo zaškrtávaním z ponuky do jednotlivých objektov.

Prostredie umožňuje tiež pracovať s grafikou: korytnačku je možné napr. zmeniť podľa obrázku.

Imagine umožňuje objektové programovanie, využíva napr. hierarchiu objektov, objektové premenné, dedenie vlastností atď. Umožňuje tiež spúšťať paralelné procesy (niekoľko

[45] http://www.ecolenet.nl/best/imagine_files/image002.jpg, [cit. 2015-04-05].

objektov vykonáva príkazy spúšťané nezávisle na sebe), programovanie založené na udalostiach (korytnačky reagujú na udalosti, napr. na kliknutie myšou alebo na na stlačenie klávesnice) a vytvárať sieťové projekty (programy, ktoré bežia súčasne na viacerých počítačoch v rámci siete a spolupracujú).

Využitie herných princípov

Imagine Logo nevyužíva žiadne gamifikačné prvky.

II. PRAKTICKÁ ČASŤ

5 ŠPECIFIKÁCIA SYSTÉMU PRE VÝUKU PROGRAMOVANIA

Špecifikácia systému pre výučbu programovania je pripravené s ohľadom na analýzu prostredia, ktorá je súčasťou bakalárskej práce a je obsahom časti I. Praktická časť.

Na základe analýzy sú požiadavky na systém nasledovné:

- vzhľadom na cieľovú skupinu a jej vek začať výučbu základov programovania pomocou grafických prvkov;
- umožniť plynulý prechod od využívania grafických prvkov k programovaniu pomocou príkazového riadku v rámci toho istého prostredia;
- pri prechode k programovaniu pomocou príkazového riadku zaradiť do výučby aj úlohy zamerané na základy písania znakov na klávesnici;
- pri výučbe programovania pomocou príkazového riadku vybrať reálne existujúci programovací jazyk;
- zabezpečiť dostupnosť programu pri štandardnom hardvérovom vybavení, tzn. počítač, klávesnica a myš;
- umožniť využívať aj neplatenú verziu programu, ako platenú verziu poskytovať iba špeciálne balíčky, ktorých absencia nebude obmedzovať základnú funkcionalitu;
- zabezpečiť lokalizáciu do českého jazyka;
- umožniť nastaviť rôzne sady úloh pre rôzne skupiny;
- využívať herné princípy, ako bodovanie, rebríčky, odznaky a pod.

Ako vizuálne prvky budú využívané grafické bloky, ktoré považujeme za intuitívnejšie ako ikony.

Programovacím jazykom, využívaním na vyšších úrovniach, bude Python. Dôvodov je niekoľko[46]:

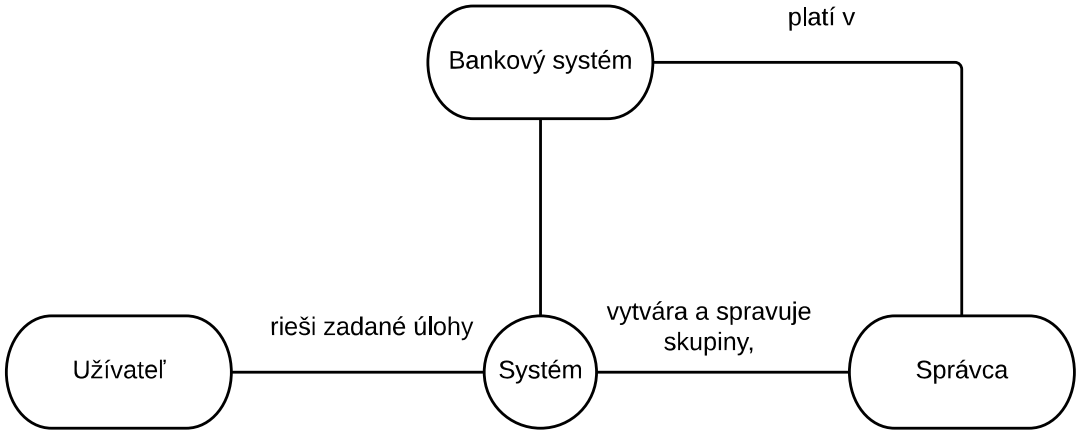
- Python je jednoduchý, resp. má jednoduchú syntax. Príkazy sú ukončené znakom konca riadku, blok je označený odsadením. Tento minimalizmus umožňuje vyvarovať sa začiatočníckych syntaktických chýb;
- využíva garbage collection, takže tu nehrozia problémy s únikom pamäte, ako napr. v C;

[46] http://www.ceskaskola.cz/2004/11/john-zelle-prvni-jazyk-python.html, [cit. 2015-04-05].

- podporuje OOP;
- je to reálny programovací jazyk, ktorý sa dá využiť v praxi.

5.1 Opis navrhovaného riešenia

Kontextový diagram zobrazuje pohľad na okolie systému. So systémom pracuje užívateľ, ktorý rieši zadané úlohy a tým postupuje na vyššie úrovne a správca, ktorý vytvára a spravuje skupiny. Bankový systém umožňuje online platbu. Iný spôsob platby ako online



5.3 Používateľské rozhrania

Tabuľka 3: Používateľské rozhrania

PR – 1	Používateľské rozhranie bude vytvorené pomocou web aplikácie.
PR – 2	Používateľské rozhranie bude rozdelené na frontend (užívateľská časť) a backend (správcovská časť).
PR – 3	Budú k dispozícii dve používateľské rozhrania – jedno pre samostatných užívateľov, druhé pre vzdelávacie inštitúcie.
PR – 4	Pre zobrazenie niektorých častí obsahu, ako aj pre zobrazenie úloh, bude potrebné prihlásiť sa.

5.4 Softvérové rozhrania

Tabuľka 4: Softvérové rozhranie

SSR – 1	Edukačný nástroj bude web aplikácia postavená na Open Source technológiách PHP, JavaScript a MySQL.
SSR – 2	Používateľské rozhranie musí korektne fungovať v prehliadačoch Internet Explorer 10+ bez režimu spätnej kompatibility, Google Chrome, Firefox 24+, Opera 12+ a Safari 7+.

Pre vývoj v PHP budeme využívať framework Yii[49], ktorý v súčasnosti patrí medzi najpoužívanejšie PHP frameworky. Yii je framework určený pre vývoj rozsiahlych webových aplikácii. Medzi jeho vlastnosti patrí:

- „rozdelenie aplikačnej a prezentačnej logiky na základe architektonického princípu MVC, ktorý umožňuje relatívnu nezávislosť prezentačnej, aplikačnej a databázovej vrstvy aplikácie a teda ich lepšiu správu;
- prístup k zdrojom dát pomocou Database Access Objects (DAO) a Active Record;
- integruje jQuery, obľúbenú javascriptovú knižnicu;

[49] Výber frameworku bude opäť prediskutovaný v čase vývoja aplikácie. Je teda možné, že konečný vývoj aplikácie bude prebiehať v inom frameworku.

- jednoduché a bezpečné spracovanie formulárov a validácia vstupných dát;
- autentizácia a autorizácia - kontrola prístupu na základe hierarchických rolí (RBAC);
- podpora motívov a rýchle vizuálne zmeny aplikácie;
- webové služby - automatické generovanie špecifikácií a spracovanie požiadaviek WSDL služieb;
- lokalizácia (L10N) a internacionalizácia (I18N) - preklad správ, formátovanie čísel a údajov o dátume a čase;
- vrstvená cache pre dáta, stránky, fragmenty stránok a dynamický obsah, ktorá významne znižuje reakčný čas aplikácie;
- spracovania, archivácia a filtrácia chýb;
- zabezpečenie a odolnosť aplikácie voči rôznym druhom útoku;
- generovanie validného XHTML kódu.“[50]

5.5 Triedy používateľov

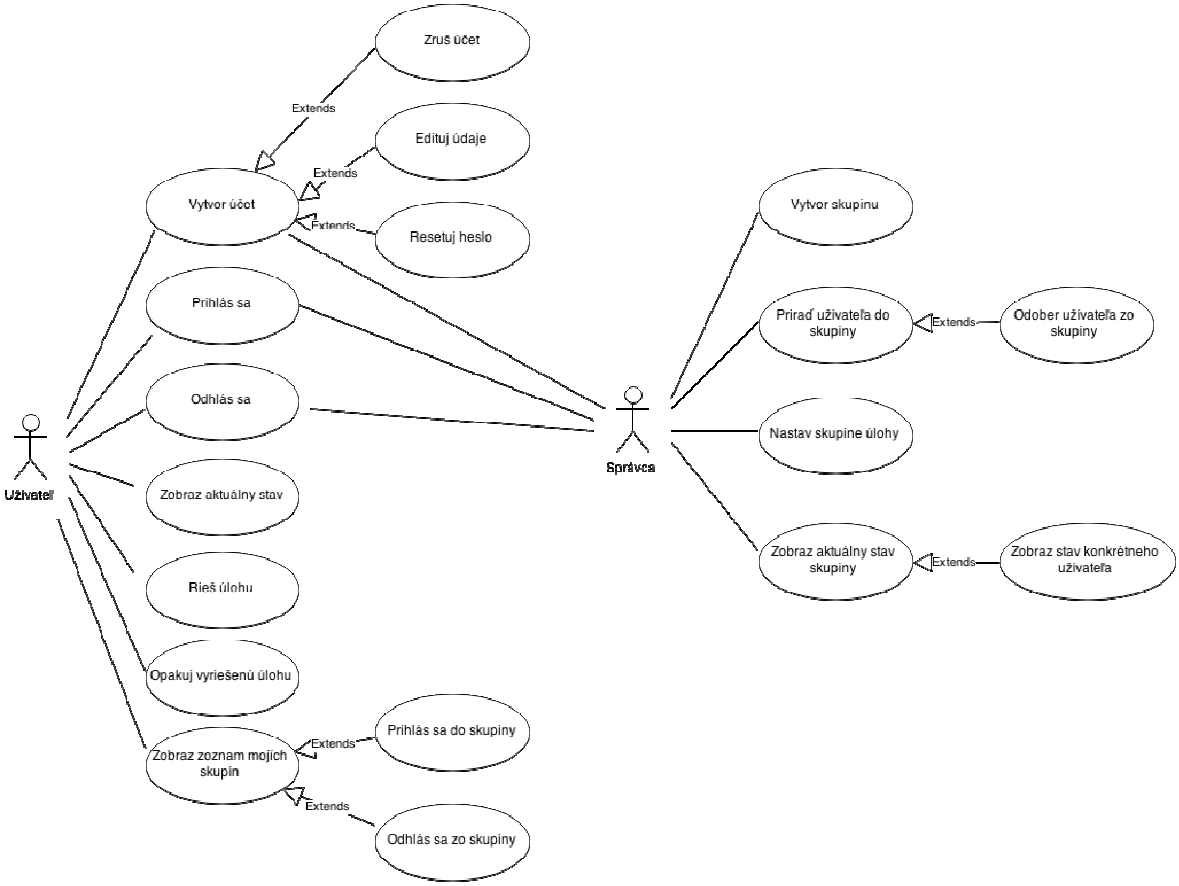
Z pohľadu používateľov bude systém obsahovať dve triedy, a to správcu a užívateľa.

Tabuľka 5: Triedy používateľov

Trieda	**Popis**
správca	Správca je fyzická osoba, ktorej úlohou je zakladanie a spravovanie skupín. Jedna osoba môže mať založený ľubovoľný počet skupín. Skupina musí mať najmenej dvoch členov. Skupina ako entita môže využívať špeciálne funkcionality určené pre skupinu.
užívateľ	Užívateľ je fyzická osoba, ktorá vie po vytvorení účtu prechádzať jednotlivými úrovňami a plniť výzvy, za ktoré získava odznaky. Po pridaní do skupiny dokáže interagovať s ostatnými členmi skupiny a plniť zadané skupinové úlohy, za ktoré získava skupinové body.

[50] http://cs.wikipedia.org/wiki/Yii [cit. 2015-04-05].

5.6 Diagram prípadov použitia



Obrázok 20: Diagram prípadov použitia

6 SADA ÚLOH

V súlade s teoretickými poznatkami bolo našim zámerom vypracovať ukážkovú sadu úloh tak, aby spĺňala požiadavky cieľovej skupiny a bola zároveň kompatibilná s požiadavkami na systém, ktoré sme uviedli v kapitole 5. Ukážková sada úloh obsahuje tri prvé úlohy, na ktorých sa snažíme ilustrovať spôsob zadávania úloh a práce s edukačným nástrojom. Všetky úlohy obsahujú na začiatku interaktívny tútoriál, ktorú uvádza žiakov do danej problematiky. Učiteľ si má možnosť zvoliť, či ho využije, alebo zvolí inú formu výučby.

Celkovo by sada úloh mala napĺňať nasledujúce všeobecné a konkrétne ciele.

Všeobecné ciele:

- rozvíjať algoritmické myslenie;
- naučiť sa vyjadrovať algoritmy pomocou grafických blokov a zvoleného programovacieho jazyka;
- realizovať zadané úlohy.

Konkrétne ciele:

- žiak dokáže navrhnúť algoritmus problému zadaného v úlohe;
- žiak dokáže overiť správnosť zvoleného algoritmu, resp. identifikovať a odstrániť vzniknutý problém;
- žiak dokáže navrhnutý algoritmus zapísať pomocou programovacieho jazyka (či už graficky alebo textovo);
- žiak rozumie, čo je príkaz a cyklus s daným počtom opakovaní;
- žiak rozumiem, čo sú udalosti, paralelizácia a nekonečný cyklus;
- žiak rozumie, čo sú premenné a pozná základné dátové typy;
- žiak rozumie, čo sú procedúry a funkcie;
- žiak dokáže používať základné príkazy jazyka Python.

6.1 Prostredie

Dôležitým prvkom je vývojové prostredie, v ktorým žiaci prechádzajú a v ktorom plnia zadané úlohy. Navrhovaný edukačný nástroj bude obsahovať vývojové prostredie, ktoré sa na prvý pohľad nelíši od herného prostredia. Žiak, hráč, prechádza príbehom, ktorý sa vyvíja tým, že plní zadané úlohy. Úlohy sú rozdelené do tematických celkov, ale aj tými prechádza žiak plynule, bez vytrhnutia z príbehu, alebo prostredia. Na konci každého tema-

tického celku je krátky vedomostný kvíz. Okrem toho sú k dispozícii výzvy, po splnení ktorých je možné získať odznaky.

Edukačný nástroj je vlastne adventúra. „Adventúra je druh počítačovej hry, v ktorej hlavný hrdina rieši rozličné úlohy. Hráč postupuje „kľukatým" príbehom k vzdialenému cieľu, pričom je kladený dôraz na rôznosť riešení zápletiek a hádaniek. Adventúra je žáner, ktorý je rozvláčny a príbehovo býva zaujímavý a spletitý, pretože hádanky a príbeh v tomto žánri podmieňujú zábavu. Štruktúrne je vytýčený tzv. „Main Quest" (hlavný cieľ / úloha), ktorý rozprestrie onú kľukatiacich úloh, po ktorých sa hráč prepracováva používaním najmä svojej fantázie a logického myslenia k naplneniu hlavného cieľa. Teda splnenie hlavného cieľa sa má nápaditým zadosťučinením za virtuálne patálie spôsobené hlavnému hrdinovi."[51] Štandardne sa adventúra ovláda formou point-and-click (voľný preklad ukáž a klikni), my túto formu ovládania nahradíme a náš hrdina sa bude pohybovať pomocou správnej kombinácie grafických blokov, resp. textových príkazov na vyšších leveloch.

Veľkou motiváciou ako po grafickej stránke, tak po stránke naratívnej, je pre nás adventúra Machinárium z dielne firmy Amanita Design. Okrem toho nás zaujal aj dvojúrovňový systém nápovedy. Hráč ma jednak k dispozícii možnosť získať drobnú dobrú radu, ako pokračovať ďalej. Zároveň je k dispozícii aj kompletný sprievodca hrou. Edukačnou hrou by tak mohli prechádzať aj menej zdatní žiaci, ktorí nie sú schopní riešiť logické úlohy, ale zároveň by mohli postupovať v základoch programovania rovnako, ako ich šikovnejší spolužiaci. Tým sa hra stáva zaujímavou aj pre využitie v bežnom vyučovacom procese, nie len v rámci voľnočasových aktivít.

[51] http://cs.wikipedia.org/wiki/Adventura [cit. 2015-04-05].

Obrázok 21: Ukážka z adventúry Machinárium[52]

6.2 Tvorba úloh

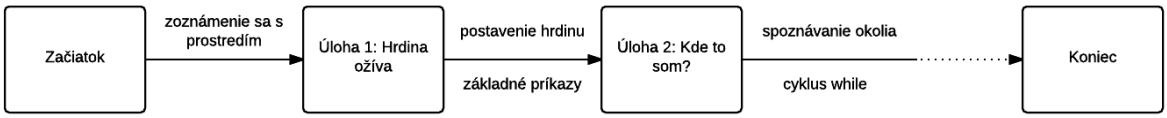
Pri tvorbe úloh sme prihliadali na postupnosť, tzn. že sú radené od jednoduchých až postupne po tie náročnejšie. Dôležité je, že žiak musí pri riešení každej úlohy dodržať logickú postupnosť.

Plnenie úloh nie je časovo obmedzené. Tým, že hra bude dostupná online, nie sú žiaci obmedzovaní vyučovacou jednotkou. Je ale dobré, keď žiakov pred každou úlohou, ktorá obsahuje nové príkazy, alebo prvky, uvedie do problematiky učiteľ.

6.3 Štruktúra sady úloh

Štruktúra sady úloh je navrhnutá tak, aby si žiak v každej úlohe prehlboval znalosti, ktoré získal na jej začiatku a zároveň si opakoval znalosti, ktoré získal v predchádzajúcich úlohách.

[52] Zdroj: https://lh3.ggpht.com/MaCL9ZRIRi9qNEiqpBR9FT-jc0-SlJpLrDWsrqPrv9lD0RsXg_hGAxryBU8Chk7DC8M=h900 [cit. 2015-04-05].



Obrázok 22: Štruktúra sady ukážkových úloh

Začiatok

Začiatok slúži na zoznámenie sa s prostredím. Žiaci sa naučia, z akých častí sa prostredie skladá, aká je „Main Quest“ a ako sa v prostredí majú pohybovať.

Úloha 1: Hrdina ožíva

Prvou logickou úlohou, ktorej budú užívatelia čeliť, je postavenie vlastného hrdinu. K dispozícii budú mať viacero kúskov, ktoré bude potrebné poskladať v správnom poradí. Na to im bude slúžiť základná sada príkazov, ktorá bude obsahovať príkazy typu „choď x krokov“, „otoč sa o 90^o/180^o stupňov“ a pod. Sada príkazov bude rozdelená do niekoľkých sekcií, napr. pohyb, cykly alebo premenné. Sady budú postupne pribúdať, ako bude žiak postupovať na vyššie stupne.

Úloha 2: Kde som to?

Keď hrdina ožije, snaží sa preskúmať svoje okolie – chodí a obzerá sa, kým nenarazí na prekážku, alebo sa nedostane k veci, ktorú si potrebuje odložiť na ďalšie použitie. Veci, ktoré je potrebné pozbierať, musí najprv na základe príbehu identifikovať (napr. ak mu bolo v na začiatku povedané, že jeho úlohou je zachrániť princeznú, potrebuje pozbierať veci, ktoré bude na ceste za splnením úlohy potrebovať – meč, vodu, jedlo a pod.)

6.4 Zadanie a metodické pokyny

Zadanie jednotlivých úloh obsahuje stručný popis úlohy, ciele, kľúčové pojmy, zadanie a pokyny pre učiteľa.

6.4.1 Začiatok

Popis úlohy

Na začiatku bude pripravené sprievodné, interaktívne video, ktoré oboznámi žiakov s prostredím a zároveň ich bude vyzývať, aby si vyskúšali základné formy ovládania hlavného hrdinu.

Ciele

Žiak:

- zoznámi sa s príbehom;
- pochopí, ako sa v prostredí pracuje, tzn. ako pomocou grafických blokov príkazov dokáže pohybovať hrdinom;
- bude vedieť používať nápovedu.

Kľúčové pojmy

- vývojové prostredie;
- grafické bloky príkazov;
- nápoveda.

Zadanie

Na začiatku je potrebné absolvovať interaktívny tútoriál. K tútorialu je možné kedykoľvek sa vrátiť a pozrieť si ho ľubovoľne veľa krát. V rámci tútoriálu je potrebné vyriešiť drobné úlohy, vďaka ktorým bude zrejmé, že žiak dokáže samostatne pracovať s prostredím.

Pokyny pre učiteľa

Aj keď bude celý tútoriál postavený tak, aby ním žiak mohol bez problémov prejsť sám, je na začiatku potrebné uviesť problematiku programovania, vysvetliť žiakom, čo sa od nich čaká a ako budú s daným nástrojom na hodine pracovať. Prípadne je možné tútoriál pozerať spolu.

Pozornosť je treba venovať najmä tomu, či žiaci pochopili, čo sa od nich očakáva, či sa dokážu vo vývojom prostredí samostatne pohybovať a či dokážu používať grafické bloky príkazov. Tiež je potrebné vysvetliť im systém získavania bodov a odznakov.

6.4.2 Úloha 1: Hrdina ožíva

Popis úlohy

Žiaci začínajú putovať krajinou a plniť „Main Quest“. Najprv však musia poskladať a oživiť hrdinu.

Ciele

Žiak:

- dokáže formulovať problém;

- vie, ako má postupovať pri riešení problému;
- dokáže zapísať jednoduchý algoritmus pomocou základnej sady príkazov.

Kľúčové pojmy

- algoritmizácia;
- základné príkazy;
- zápis algoritmu.

Zadanie

Kde sa stratil hrdina? Aby bolo možné splniť hlavnú úlohu, musíme nájsť nášho hrdinu. To ale nie je také jednoduché. Náš hrdina je totiž výtvorom starého vedca, ktorý vyrobil jednotlivé časti. To ho tak vyčerpalo, že si musel ísť zdriemnuť. My však nemôžeme čakať, kým sa zobudí. Vidíme, kde má hrdina nohy, ostatné časti sú ale stratené medzi inými nepotrebnými vecami v dielni nášho vedca. Dokážeme ich nájsť a pozbierať ich v správnom poradí?

Pokyny pre učiteľa

Úlohou učiteľa je uviesť žiakov do problematiky algoritmizácie a ilustrovať túto problematiku na zadanej úlohe.

6.4.3 Úloha 2: Kde som to?

Popis úlohy

Po oživení sa hrdina opúšťa vedcovu dielňu a vydáva sa na cestu.

Ciele

Žiak:

- chápe, čo je to cyklus;
- dokáže používať cyklus do-while.

Kľúčové pojmy

- cyklus;
- konečný cyklus;
- podmienka;
- do-while.

Zadanie

Hrdina je už celý. Čo teraz? Je potrebné zistiť, ako ďalej. Na stole vidíme ležať napísaný odkaz. Prečítame si ho tak, že sa k nemu priblížime. Na odkaze si prečítame naliehavú misiu, určenú pre vedca. Ten však spí, preto musíme konať. Na takú náročnú misiu sa však musíme pripraviť. V dielni sa povaľuje veľa vecí, ktoré by sme mohli potrebovať a tiež sa nejako potrebujeme dostať von z dielne.

Pokyny pre učiteľa

Na začiatku je dobré si zopakovať problematiku algoritmizácie. Aj v tejto úlohe, podobne ako vo všetkých ďalších, je potrebné vyriešiť niekoľko problémov. Formulovať ich žiaci môžu spoločne, tiež môžu spoločne navrhnúť postup, ako tieto problémy riešiť. Nevadí pokiaľ sa nezhodnú, každý si môže vyskúšať svoju variantu a overiť si tak správny postup riešenia.

Okrem opakovania algoritmizácie sa žiaci pri riešení tejto úlohy dostávajú aj k problematike cyklov. Učiteľ môže využiť náučný interaktívny tútoriál, ktorý sa nachádza na začiatku každej úlohy.

Treba však počítať s tým, že úlohy sú pomerne časovo náročné, tzn. že nie je možné ich dokončiť počas jednej vyučovacej hodiny. Je preto si potrebné overiť, či majú žiaci doma prístup k počítaču a internetu. Na základe toho si treba rozvrhnúť pracovné tempo.

ZÁVER

Predkladaná bakalárska práca sa zaoberá problematikou výučby základov programovania a algoritmizácie na druhom stupni základnej školy. Za cieľ sme si stanovili zmapovať súčasnú situáciu v Českej republike, analyzovať požiadavky na systém pre výučbu programovania z pohľadu cieľovej skupiny, učiteľov a správcov systému a analyzovať existujúce, najčastejšie používané edukačné systémy. Na základe zistených skutočnosti následne špecifikovať požiadavky a zostaviť ukážkovú sadu úloh pre unikátny edukačný softvér, určený pre výučbu programovania na druhom stupni základných škôl v Českej republike.

V teoretickej časti sme najskôr zmapovali súčasnú situáciu na českých školách. Venovali sme sa témam, aký dopad má časová dotácia na výučbu informatiky a programovania, aký je stav škôl z pohľadu informatiky, ktoré programovacie jazyky sa využívajú, a aké sú používané metódy výučby. Zhodnotili sme, že IKT nefiguruje ako Prierezová téma, hoci sa dotýka takmer všetkých vzdelávacích oblastí. Rovnako, výučba programovania a algoritmizačné úlohy sú v podstate okrajové učivo. Vyplýva to aj z extrémne nízkej časovej dotácie. Opierali sme sa aj o publikované závery prieskumov na českých základných školách.

Ďalej sme sa venovali požiadavkám na systém pre výučbu programovania z pohľadu cieľovej skupiny. Pri výučbe programovania má učiteľ na výber niekoľko nástrojov a metód, ktoré si môže zvoliť v závislosti od zloženia skupiny a požadovanej náročnosti. Jednou z možností je vybrať si jeden z programovacích jazykov, druhou zvoliť si edukačný softvér. Učiteľ taktiež stojí pred voľbou, či si má vybrať výučbové, alebo profesionálne nástroje, vizuálne alebo textuálne nástroje. Väčší priestor sme venovali využitiu herných princípov vo vyučovaní programovania.

Ďalšiu kapitolu sme venovali užívateľským požiadavkám na systém pre výučbu programovania od technických parametrov hardvéru, cez modely nasadenia, obstarávacie náklady, až po opis rôznych spôsobov podpory užívateľa.

Analýzu existujúcich systémov považujeme za dôležitý krok v príprave špecifikácie navrhovaného systému. Na jednej strane nám pomôže pochopiť, ako tieto systémy fungujú, čo užívateľom ponúkajú a čo je žiadané zo strany užívateľov, na strane druhej sa vďaka nej dozvieme, čo naopak na trhu chýba a čím by nový nástroj mohol prispieť. Analýza ôsmych systémov uzatvára teoretickú časť zistením, že ani jeden z analyzovaných systémov nespĺňa naše požiadavky. Najčastejším dôvodom je chýbajúca lokalizácia do českého jazyka a zložitý prechod od vizuálneho k textovému vývojovému prostrediu.

V praktickej časti navrhujeme špecifikáciu systému pre výučbu programovania vrátane návrhu rozhrania, definovania tried používateľov a diagramu prípadov použitia. Od špecifikácie systému prechádzame k návrhu vývojového prostredia následne k tvorbe vzorových úloh. Ukážková sada úloh obsahuje tri prvé úlohy, na ktorých sa snažíme ilustrovať spôsob zadávania úloh a práce s edukačným nástrojom, vrátane pokynov pre učiteľa. Tým aj uzatvárame predkladanú bakalársku prácu, ktorej téme by sme sa radi venovali aj naďalej.

ZOZNAM POUŽITEJ LITERATÚRY

[1] PANE, F. John. A Programming System for Children that is Designed for Usability. School of Computer Science, Computer Science Department, Carnegie Mellon University Pittsburgh, PA. [cit. 2015-04-05]. Dostupné z: http://www.cs.cmu.edu/~pane/ftp/PaneThesis.pdf

[2] BALADA, J. a kol. Rámcový vzdělávací program pro základní vzdělávání (verze platná od 1.9.2013), Praha: MŠMT ČR 2013. [cit. 2015-04-05]. Dostupné z: http://www.pf.jcu.cz/ research/svp/rvp-zv-0905.pdf

[3] VANÍČEK, Jiří. Potenciální a skutečný dopad informatické soutěže do změn kurikula ICT v České Republice. [cit. 2015-04-05]. Dostupné z: http://www.bebras.org/sites/default/files/documents/publications/Vanicek-2012.pdf

[4] KREJSA, Jan. *Výuka základů programování v prostředí Scratch*. České Budějovice, 2014. Diplomová práce. Jihočeská univerzita v Českých Budějovicích, Pedagogická fakulta, Katedra informatiky. Vedoucí diplomové práce doc. PaedDr. Jiří Vaníček, Ph.D. [cit. 2015-04-05]. Dostupné z: http://www.ceskaskola.cz/2014/05/jan-krejsa-scratch-vyuka-zakladu.html

[5] PECINOVSKÝ, Rudolf. *Proč učit programování na základní škole*. Dostupné z: http://www.ceskaskola.cz/2001/09/rudolf-pecinovsky-proc-ucit.html

[6] BELAN, Anino. Kurz jazyka C. učebný text pre kvartu a kvintu osemročného gymnázia, 2. Vydanie. Bratislava: Škola pre Mimoriadne Nadané Deti a Gymnázium, Teplická 7, 831 02 Bratislava 2003 – 2011.

[7] FILIPČÍK, Richard. *Vyhodnocovanie aktivity a motivácia vo webovom výučbovom systéme*. Bratislava, 2014. Bakalárska práca. Ústav informatiky a softvérového inžinierstva, FIIT STU v Bratislave. Vedúci práce: prof. Ing. Mária Bieliková, PhD. [cit. 2015-04-05]. Dostupné z: http://www2.fiit.stuba.sk/~bielik/ courses/projects/pdf/bp2014-filipcik.pdf.

[8] VORDERMAN, Carol, Jon WOODCOCK, Sean MCMANUS, Craig STEELE, Claire QUIGLEY a Daniel MCCAFFERTY. Help your kids with computer coding: a unique step-by-step visual guide, from binary code to building games. First American edition. 224 pages. ISBN 978-146-5419-569.

[9] KUBICA, Jeremy. Best Practices of Spell Design. CreateSpace Independent Publishing Platform, 2013. ISBN 978-1481921916.

[10] KUBICA, Jeremy. Computational fairy tales. North Charleston, S.C: 2012. ISBN 978-147-7550-298.

[11] MAYER, By Jake. A tale of friends, enemies and Minecraft. San Bernardino, CA: 2013. ISBN 978-148-9574-091.

[12] BELLANCA, James A a Ronald S BRANDT. 21st century skills: rethinking how students learn. Bloomington, IN: Solution Tree Press, c2010, xxxi, 375 p. Leading edge (Bloomington, Ind.), 5. ISBN 19-352-4990-8.

[13] ROBINSON, Ken. Out of our minds: learning to be creative. Fully rev. and updated ed. Chichester: Capstone, 2011, xvii, 326 s. ISBN 978-085-7081-490

[14] KLEMENT, Milan, Jiří KLEMENT a Jan LAVRINČÍK. Metody realizace a hodnocení výuky základů programování. Olomouc, 2012. ISBN 978-80-87658-01-7.

[15] VANÍČEK, Jiří. Informatika pro základní školy. Vyd. 1. Brno: Computer Press, 2005, 88 s. Učebnice (Computer Press). ISBN 80-251-0630-6.

[16] Apps for Teaching Children Coding Skills | Edutopia. Edutopia 2013 [cit. 2015-04-05]. Dostupné z: http://www.edutopia.org/blog/7-apps-teaching-children-coding-anna-adam

[17] Resources for Teaching Kids How to Program & Code. [cit. 2015-04-05].. Dostupné z:http://www.apartmenttherapy.com/20-resources-for-teaching-kids-how-to-program-code-200374

ZOZNAM WEBOVÝCH SÍDIEL

https://studio.code.org/

http://www.sgpsys.cz/cz/

https://scratch.mit.edu/

http://www.kuatostudios.com/games/hakitzu-elite/

http://www.lego.com/en-us/mindstorms/?domainredir=mindstorms.lego.com

http://www.ceskaligarobotiky.cz

https://www.tynker.com/

http://www.alice.org/index.php

https://lookingglass.wustl.edu/

http://www.pf.jcu.cz/imagine/index.php

ZOZNAM POUŽITÝCH SYMBOLOV A SKRATIEK

BSD	Berkeley Software Distribution.
ČR	Česká republika.
FAQ	Frequently asked questions.
GNU GPL	GNU General Public License.
HCI	Human-Computer Interaction.
IDE	Integrated Development Environment.
IKT (ICT)	Informačno-komunikačné technológie.
MIT	Massachusetts Institute of Technology.
MS	Microsoft.
OOP	Objektovo-orientované programovanie.
OS	Operačný systém.
RPG	Role Playing Game.
RVP	Rámcový vzdelávací program.
SaaS	Software as a Services.
ZŠ	Základná škola.

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ZOZNAM PRÍLOH

PRÍLOHA P I: PREHĽAD ANALYZOVANÝCH EDUKAČNÝCH SYSTÉMO

Názov systému	Požiadavky na hardvér	Model nasadenia	Platený systém	Lokalizácia	Podpora systému	Využívanie gamifikačných prvkov	Vizuálny/ Textuálny nástroj	Používaný programovací jazyk	Možnosť plynulého prechodu na programovací jazyk
Hakitzu	smartfón, tablet	aplikácia	nie	anglický jazyk	áno	áno	áno/áno	JavaScript	áno
Lego Mindstorms	stavebnica	on-premises	áno	český jazyk	áno	áno	áno/nie	ikonický	nie
Scratch	počítač	on-premises, SaaS	nie	český jazyk	áno	áno	áno/nie	grafické bloky príkazov	nie
Baltík	počítač	on-premises	nie	český jazyk	áno	nie	áno/nie	ikonický	nie
Tynker	počítač	Saas	áno	anglický jazyk	áno	áno	áno/áno	grafické bloky príkazov, JavaScript	áno
Alice	počítač	on-premises	nie	anglický jazyk	áno	nie	áno/áno	grafické bloky príkazov, Java	áno
Robot Karel	počítač	on-premises	nie	český jazyk	nie	nie	nie/áno	vlastný	nie
Imagine Logo	počítač	on-premises	nie	český jazyk	áno	nie	nie/áno	vlastný	nie