Pedagogická fakulta
Faculty of Education
Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

Jihočeská univerzita v Českých Budějovicích
Pedagogická fakulta
Katedra informatiky

Bakalářská práce

Co vlastně děti programují ve Scratch?

Vypracoval: Matěj Hudičák
Vedoucí práce: doc. PaedDr. Jiří Vaníček, Ph.D.

České Budějovice 2017

Prohlašuji, že svoji práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě fakultou elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

Datum Podpis studenta

Poděkování

Děkuji vedoucímu práce panu doc. PaedDr. Jiřímu Vaníčkovi, Ph.D. za ochotu a trpělivost pří poskytování rad během tvorby této práce. Díky mu patří i za to, že právě v jeho hodinách jsem se se Scratch setkal poprvé.

Název práce: Co vlastně děti programují ve Scratch?

Autor: Matěj Hudičák

Katedra: Katedra Informatiky

Vedoucí diplomové práce: doc. PaedDr. Jiří Vaníček, Ph.D.

E-mail vedoucího: vanicek@pf.jcu.cz

ANOTACE

Tato práce se zabývá výzkumem tvořivé činnosti dětí v prostředí programovacího jazyka Scratch. Cílem je zdokumentovat, jakými tématy se děti zabývají a inspirují. Dále projekty analyzovat v rámci používaných příkazů a projekty hodnotit prostřednictvím vytvořených prvků. Výsledky a data zjistíme pomocí nalezených a dostupných nástrojů. Všechny výstupy analýzy budou zobrazeny v grafech. Seznámíme se také s příklady skriptů připojených k textu pomocí obrázků. Výstupy z této práce se mohou dále použít při tvorbě učebnic a výukových metod. Tato práce má za úkol zvýšit zájem o výuku ve Scratch nejen v hodinách programování, ale i v jiných předmětech.

KLÍČOVÁ SLOVA

Scratch, analýza, výuka, hodnocení, projekty, témata, skripty

Title: What children in fact program in Scratch?

Author: Matěj Hudičák

Department: Department of Informatics

Supervisor: doc. PaedDr. Jiří Vaníček, Ph.D.

Supervisor´s e-mail address: vanicek@pf.jcu.cz

ANOTATION

This thesis deals with the research of creative activities of children in the Scratch programming language environment. The aim is to document the themes that children engage. In addition, analyze projects within commands and projects to evaluate through the elements created. We find the results and data with the help of found and available tools. All analysis outputs are shown in the graphs. We also encounter examples of scripts linked to text codes or images. The knowledge found in this work can be further used in the creation of textbooks and teaching methods. This work is intended to increase the interest in teaching in Scratch.

KEYWORDS

Scratch, analysis, teaching, evaluation, projects, themes, script

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH
Fakulta pedagogická
Akademický rok: **2015/2016**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Matěj HUDIČÁK**
Osobní číslo:	**P15687**
Studijní program:	**B7507 Specializace v pedagogice**
Studijní obory:	**Technická výchova se zaměřením na vzdělávání**
	Informační technologie se zaměřením na vzdělávání
Název tématu:	**Co vlastně děti programují ve Scratch?**

Zadávající katedra: **Katedra informatiky**

Zásady pro vypracování:

Scratch je projektem skupiny Lifelong Kindergarten z Massachusettského technologického institutu. Jedná se o programovací prostředí, ve kterém si žáci základních škol osvojují základní znalosti programování. Jeho vizuální stránka, blokové programování s odstraněním chyb syntaxe a webové komunitní prostředí podporuje moderní výuku. Tvůrci programovacích projektů ve Scratch ukládají své projekty na cloudové uložiště. K měsíci březnu roku 2016 se v databázi nachází přes 14 milionů projektů od více než 11 milionů uživatelů. Nejvíce registrovaných uživatelů uvádí svůj věk mezi 10 - 16 lety.

Cílem práce bude zmapování těchto projektů z pohledu témat, použitých technik a nástrojů či programovacích metod. Na několika stech pečlivě vybraných projektů bude provedena analýza za účelem získání přehledu o tom, co a jak děti v tomto prostředí vytvářejí a jaké používají programovací techniky. Podle různých kritérií (např. struktura zdrojového kódu, grafické zpracování, zvuková prezentace) budou projekty hodnocené a následně tříděné tak, aby bylo možné a snadné poznat, na co se tvůrci projektů zaměřují. Další část analýzy bude zaměřena na sociální interakci mezi uživateli - remixované a upravené projekty, komentáře k projektům v komunitě programátorů. Dbáno bude také na kontaktování tvůrců projektů a účast v diskuzích, tak aby bylo dosaženo co nejrelevantnějších informací.

Tato sonda by měla poskytnout představu o tom, co programátory dětského věku zajímá programovat, kde berou inspiraci, jaké programování zvládají. Vzhledem k tomu, že velká většina projektů vzniká mimo výuku, výsledky výzkumu budou užitečné pro stanovování vzdělávacího obsahu výuky programování ve školách.

Rozsah grafických prací:	**CD ROM**
Rozsah pracovní zprávy:	**40**
Forma zpracování bakalářské práce:	**tištěná**
Seznam odborné literatury:	**viz příloha**

Vedoucí bakalářské práce:	**doc. PaedDr. Jiří Vaníček, Ph.D.** Katedra informatiky
Datum zadání bakalářské práce:	**20. dubna 2016**
Termín odevzdání bakalářské práce:	**28. dubna 2017**

Mgr. Michal Vančura, Ph.D.
děkan

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH 20

doc. PaedDr. Jiří Vaníček, Ph.D.
vedoucí katedry

V Českých Budějovicích dne 20. dubna 2016

Příloha zadání bakalářské práce

Seznam odborné literatury:

1. **Forsgren Velasquez, N., Fields, D. A., Olsen, D., Martin, H. T., Strommer, A., Sheperd, M. C., Kafai, Y. B. (2014). Novice programmers talking about projects: What automated text analysis reveals about online Scratch users' comments. In the Proceedings of the Annual Hawaii International Conference on System Sciences (HICSS). Waikoloa, Hawaii. IEEE.**

2. **Carini, G. (2012). Finding a Needle in a Haystack: New Ways to Search and Browse on Scratch. Masters Thesis, MIT Media Lab.**

3. **Hill, B M., Monroy-Hernández, A. (2013). The remixing dilemma: the trade-off between generativity and originality. American Behavioral Scientist, 57-5, Pp. 643-663.**

4. **Fields, D. A., Giang, M., Kafai, Y. B. (2013). Understanding collaborative practices in the Scratch online community: Patterns of participation among youth designers. In N. Rummel, M. Kapur, M. Nathan, & S. Puntambekar (Eds), CSCL 2013 Conference Proceedings, Volume 1. International Society of the Learning Sciences: Madison, WI, 200-207.**

5. **Lifelong Kindergarten Group at the MIT Media Lab. Scratch Statistics - Imagine, Program, Share. Scratch - Imagine, Program, Share. [online]. [2003] [cit. 2016-04-14]. Dostupné z: https://scratch.mit.edu/statistics/**

6. **Lifelong Kindergarten Group at the MIT Media Lab. Discuss Scratch. Scratch - Imagine, Program, Share. [online]. [2003] [cit. 2016-04-14]. Dostupné z: https://scratch.mit.edu/discuss/**

7. **Scratch Wiki Home. Scratch Wiki. [online]. [2008] [cit. 2016-04-14]. Dostupné z: http://wiki.scratch.mit.edu/wiki/Scratch_Wiki_Home**

8. **Lifelong Kindergarten Group at the MIT Media Lab. Scratch - Imagine, Program, Share. Scratch - Imagine, Program, Share. [online]. [2003] [cit. 2016-04-14]. Dostupné z: https://scratch.mit.edu/**

Obsah

1 Úvod

Tato bakalářská práce se zabývá programátorskou platformou Scratch. Konkrétně se práce věnuje tomu, jaké projekty zde vytvářejí děti. Znalosti získané při tvorbě této práce mohou být zužitkované v pedagogické praxi. Práce je zaměřením teoretická, je rozdělena do osmi kapitol. Výsledky práce, především její analytické části, jsou shrnuty v závěru.

1.1 Cíle práce

Cílem této práce je pochopení a shromáždění evidence o tom, jaké programy děti vytvářejí, jaké programovací prvky nejvíce využívají a jakým tématům se věnují. Tyto postřehy lze poté aplikovat při tvorbě učebnic a výukových postupů i při hodinách programování. V rámci moderní výuky, podpory kreativního a samostatného myšlení může být Scratch významnou inspirací pro pedagogy.

Při shromažďování dat bylo zkoumáno, jak vlastně funguje projekt Scratch, tedy jak vypadají příkazy pod grafickým rozhraním. O tom, v jaké podobě jsou projekty ukládány v databázi, se můžeme dozvědět ve 2. kapitole. V předběžném průzkumu byly zachyceny prvky, které se dají nazvat jako páteřní, determinující jednotlivé projekty. Tyto prvky nazýváme Interaktivita, Operátory, Vlastní příkazy, Řídící a Datové struktury, viz *kapitola 4*. Vyhledávání projektů a shromažďování dat bylo provedeno pomocí dostupných nástrojů představených v kapitolách 5, 6 a 7. Celkem se analyzovalo 300 projektů shromážděných v 6 tematických skupinách. Tyto skupiny si dále rozebereme v 8. kapitole.

1.2 Terminologie

Programovací jazyk Scratch

Scratch je bezplatný výukový programovací jazyk, který byl vyvinut v MIT skupinou Lifelong Kindergarten (Celoživotní mateřská školka). Je směřován na děti ve věku 8–16 let. Scratch je navržen tak, aby byl zábavný, edukační a jednoduchý. [1] Programuje se zde pomocí přetahování a skládání příkazů do sebe podobně jako puzzle. Tento typ programování je označován jako „drag – and – drop programování“. Základními kameny tohoto programovacího jazyka jsou výrazy Imagine, Program, Share; v překladu Vymysli, Programuj, Poděl se. Název je odvozený od tzv. scratchingu, který v informatice znamená opakované využívání programů nebo jejich částí, které lze kombinovat, sdílet a přizpůsobovat jiným účelům. [2]

Projekt

Projekt je naprogramované dílo, které je vytvořeno v tzv. Scratch program‘s editor. [3] Toto prostředí se využívá pro design a programování projektů. Díky různorodosti a velkému výběru možných příkazů pro ovládání programu je možné tvořit rozmanité projekty.

Scratch program editor

V tomto programovacím prostředí se vytvářejí projekty. Nalezneme zde scénu, panel s postavami a pozadími, paletu příkazů, skriptovací oblast, nástroje pro úpravu hudby a grafiky a další součásti vyšších operací s celým projektem. [4]

Kategorie příkazů

Každý příkaz, který můžeme ve Scratch použít při budování vlastního programu, patří do určité skupiny. Můžeme je řadit buď podle jejich tvaru a typu, nebo podle toho, do jaké patří kategorie. V této práci řadíme příkazy do kategorií, které se vymezují názvem a barvou. Tyto kategorie se nazývají Pohyb (modrá), Vzhled (fialová), Zvuky (purpurová), Pero (zelená), Data (žlutooranžová), Události (hnědá), Ovládání (žlutá), Vnímání (světle modrá), Operátory (zelenožlutá) a Vlastní příkazy (modrofialová). [5]

Scratch databáze

Zde se nachází všechny vytvořené projekty. Vyhledávat můžeme ovšem jen ty sdílené, ostatní jsou viditelné jen autorům a administrátorům. V květnu roku 2017 bylo v databázi evidováno přes 23 miliónů projektů. [6]

Skupina projektů

Skupinou projektů v této práci myslíme skupiny určitého tématu. V každé této skupině je 50 projektů souvisejících s určitým motivem. V této práci mluvíme o celkem šesti skupinách. Jsou to Animace, Hry, Návody, Umění, Hudba a Příběhy. Jednotlivě jsou popisovány v deváté kapitole.

Studio

Důvodem pro vytvoření studia je potřeba uskupení projektů, které se týkají jednoho tématu. Moderátoři Scratch vytvářejí například různé soutěže, ve kterých se shromažďují vytvořené projekty na dané téma. Studia se vytváří také z mnoha jiných důvodů. Jde například o školní či třídní studia, ve kterém učitel shromažďuje projekty žáků. Také se můžeme setkat se studii, která jsou plná návodů například na vytvoření simulace fyzikálních zákonů v projektech atd. [7]

2 Metodologie

První kroky této práce směřovaly k pochopení mechanismů běžících za grafickým rozhraním Scratch. Získali jsme přehled o tom, jak jsou projekty a všechny jejich části kódovány a v jaké podobě ukládány.

Dále jsme se zabývali oficiálními statistikami a čerpali z nich potřebné informace. Příhodně k tématu přišla myšlenka na vytvoření podobných statistik i v této práci.

Významnou inspirací se stal webový portál DrScratch. [8] Pomocí tohoto portálu je možné analyzovat projekt a podle různých kritérií dostat bodové ohodnocení. Na základě poznatků jsme vytvořili vlastní kritéria a metodu hodnocení. Naše hodnocení se týkalo zjišťování výskytu určitých prvků v projektu jako jsou Vlastní příkazy, Programové a řídící struktury, Interaktivita a jiné.

Pro potřeby této práce jsme využili též nástroj Hairball, pomocí tohoto programu lze například zjistit počet a názvy využitých příkazů a další zajímavé informace o projektu. V souvislosti s tímto vznikl požadavek na vyhledání projektů a zároveň jakým programem to nejlépe provést.

Chtěli jsme také získat různá data, proto jsme je rozdělili do několika tematických skupin, které by se mezi sebou mohly porovnávat. Nakonec bylo vybráno šest skupin projektů s názvy Animace, Hry, Umění, Hudba, Příběhy a Návody. Každá skupina obsahovala 50 projektů. Celkem bylo analyzováno 300 projektů, ve kterých se nacházelo 21840 příkazů. Ty jsou zapsány v tabulkovém procesoru Excel. Výsledkům této práce je věnována 10. kapitola a její podkapitoly.

3 Forma a podoba uloženého Scratch projektu

Tato kapitola vychází z informací nalezených na stránce s citační značkou [10].

Tento souborový formát je využíván k ukládání Scratch 2.0 projektů. V této kapitole je představen projekt, jak vypadá uvnitř v jeho textové podobě, tedy jak je zakódovaný. V prvé řadě jsou to objekty obsahující klíče a hodnoty. Klíče určují, co a hodnoty kde, nebo jak to má vypadat. [9]

Projekt stáhneme vždy ve formátu *název.sb2*. Název zde znamená název projektu. Pro jeho dekomprimaci a získání všech jeho částí musíme v souborovém manažeru tuto příponu změnit na *.zip* a rozbalit v příslušném programu. Po rozbalení máme k dispozici složku, která obsahuje informace o projektu zakódované v textově založeném formátu JSON a projektová média (zvuk, grafika).

3.1 JSON

JSON, zkratka pro JavaScript Object Notation, je datový formát pro ukládání scriptů a informací (např. odkazy na zvuk, grafiku a další připojené informace) Scratch 2.0 projektů. Tento formát reprezentuje objekty se syntaxí `{ "key" : value, …}`, kde *key* je uveden jako textový řetězec v dvojitých uvozovkách a znamená název klíče. Hodnota takového klíče se nachází za dvojtečkou s označením *value*. Hodnota může být jiný objekt, pole a datové typy: textový řetězec (uvádí se ve dvojitých uvozovkách), čísla (int 100, float 3.14), boolean (`true, false`) nebo `null`. [10]

3.2 Formát, jeho klíče a hodnoty

V této kapitole si představíme všechny klíče a hodnoty, kterých mohou nabývat, objevující se v souboru *project.json*. Tyto klíče v textové podobě popisují, jak má vypadat projekt, kde se v okamžiku uložení projektu na scéně nachází postava nebo display proměnné. Dále například na jakých souřadnicích se nachází osa postavy. Také jsou v jaké pozici uložené skripty a komentáře ve skriptovací části. Tyto klíče poté prochází nástroji Scratch Virtual Machine a Scratch GUI, my poté vidíme po otevření projekt tak, jak byl uložen.

3.2.1 Objekty scény

Tyto objekty jsou kořenové objekty projektu nacházející se v extrahované složce v souboru *project.json*. Následuje přehled klíčů (tučné značení) a hodnot, kterých mohou nabývat:

objName

Název scény. V českém jazyce defaultně "Scéna", hodnota záleží na tom, v jakém jazyce je Scratch spuštěn.

variables

Globální proměnné projektu zapsané jako pole objektů proměnných.

lists

Globální seznamy projektu zapsané jako pole objektů seznamů.

scripts

Všechny skripty projektu zapsané jako seznam množin kódu.

scriptComments

Komentáře u skriptů zapsané jako seznam množin komentářů.

sounds

Zvuky scény zapsané jako seznam množin zvuků.

costumes

Pozadí scény zapsané jako seznam kostýmů.

currentCostumeIndex

Číslo pozadí scény.

penLayerID

Číslo obrázkového souboru v ZIP archivu projektu. Tento obrázek obsahuje čáry pera na scéně z momentu, kdy byl projekt uložen.

penLayerMD5

The MD5 hash název pro obraz čar pera na scéně v momentě, kdy byl projekt uložen.

tempoBPM

Uložené tempo z momentu uložení projektu.

videoAlpha

Průhlednost videa z momentu uložení projektu.

children

Postavy a displeje na scéně v projektu; seznamy postav, displejů a seznam objektů. Objekty, které se objevují později, jsou výše než objekty, které se objevují dříve.

info

Zvláštní informace o uživateli a projektu zobrazené jako objekt informací.

3.2.2 Objekt postavy

V objektu postavy jsou uložené informace týkající se postavy. Nalezneme zde následující klíče a hodnoty:

objName

Název postavy.

variables

Proměnné, týkající se postavy, zobrazené jako seznam objektů proměnných.

lists

Pole, týkající se postavy, zobrazené jako seznam polí.

scripts

Scripty postavy zobrazené jako seznam množin scriptů.

scriptComments

Komentáře u postavy zobrazené jako seznam množin komentářů.

sounds

Zvuky postavy zobrazené jako seznam zvukových objektů.

costumes

Kostýmy postavy zobrazené jako seznam kostýmů.

currentCostumeIndex

Číslo kostýmu postavy.

scratchX

X-ová souřadnice pozice postavy relativně ke středu scény.

scratchY

Y-ová souřadnice pozice postavy relativně ke středu scény.

scale

Velikost postavy pomocí čísla, ve kterém se 1 rovná 100 %.

direction

Směr otočení postavy jako číslo uvedené ve stupních, měřeno po směru hodinových ručiček od 0 (ručička natočená směrem nahoru).

rotationStyle

Styl rotace postavy jako textový řetězec; hodnoty "normal", "leftRight", nebo "none".

isDraggable

True, pokud je možné postavu přetáhnout; false v opačném případě.

indexInLibrary

Číslo indikující seřazení postav v seznamu postav pod scénou.

visible

True, pokud byla postava odkryta v momentě uložení projektu; false, pokud byla postava skryta.

spriteInfo

Doplňkové informace o postavě. V současné době vždy v podobě prázdného objektu.

3.2.3 Objekty na monitoru scény

Objekt monitoru scény obsahuje informace o monitoru scény. Uchovává následující klíče a hodnoty:

target

Název scény nebo postavy, ke které displej zobrazený na scéně patří.

cmd

Druh displeje scény. Validní hodnoty jsou:

cmd	Description	param
"answer"	The answer.	null
"backgroundIndex"	The backdrop number of the stage.	null
"costumeIndex"	The costume number of the stage monitor's target.	null
"getVar:"	A variable monitor.	The variable name.
"heading"	The direction of the stage monitor's target.	null
"scale"	The size of the stage monitor's target.	null
"sceneName"	The backdrop name of the stage.	null
"senseVideoMotion"	The video motion on the stage monitor's target.	"*type*,*thing*" where *type* is motion or direction and *thing* is Stage or this sprite
"soundLevel"	The loudness.	null
"tempo"	The tempo.	null
"timeAndDate"	A value of the current () block.	"year", "month", "date", "day of week", "hour", "minute", or "second", depending on which one the monitor is showing
"timer"	The timer.	null
"volume"	The volume of the stage monitor's target.	null
"xpos"	The X position of the stage monitor's target.	null
"ypos"	The Y position of the stage monitor's target.	null

Obr. č. 1: Klíče a hodnoty displejů

param

Parametr příkazu displeje zobrazeného na scéně. Viz obrázek výše.

color

Barva pro displej zobrazený na scéně, hodnota klíče zapsána hexadecimálním kódem.

label

Popiskový text pro displej zobrazený na scéně.

mode

Mód displeje zobrazeného na scéně. Nabývá hodnot 1, 2 nebo 3.

sliderMin

Minimální hodnota na posuvníku zobrazeného displeje.

sliderMax

Maximální hodnota na posuvníku zobrazeného displeje.

isDiscrete

True, pokud displej povoluje pouze číselné hodnoty, false v opačném případě.

x

X-ová pozice zobrazeného displeje relativně od levého horního okraje scény.

y

Y-ová pozice zobrazeného displeje relativně od levého horního okraje scény.

visible

True, pokud je displej zobrazený. False, pokud je skrytý.


```
{
	"target": "Stage",
	"cmd": "getVar:",
	"param": "x",
	"color": 15629590,
	"label": "x",
	"mode": 1,
	"sliderMin": 0,
	"sliderMax": 100,
	"isDiscrete": true,
	"x": 5,
	"y": 5,
	"visible": true
}
```


Obr. č. 2: Displej proměnné

3.2.4 Objekt proměnné

Objekt proměnné ukládá informace o každé vytvořené proměnné. Obsahuje následující klíče a jejich hodnoty:

name

Název proměnné.

value

Hodnota proměnné.

isPersistent

True, pokud se jedná o cloudovou proměnnou. Hodnoty false nabývá v opačném případě.

Příklad:

```
{
      "name": "x",
      "value": 6,
      "isPersistent": false
}
```

3.2.5 Objekt seznamu

Objekt seznamu uchovává informace o seznamu a jeho displeji na scéně. Každý seznam má dva odpovídající objekty v souboru sb2. První se nachází v poli seznamů u postavy (používaný pro zobrazení listu u postavy, ke které patří) a druhý u "children" pole (používaný pro řádné zobrazení displejů a dalších objektů na scéně). Obsahuje tyto klíče a hodnoty:

listName

Název seznamu.

contents

Pole položek v seznamu.

isPersistent

True, pokud se jedná o cloudový seznam; v jiném případě false.

x

X-ová pozice zobrazeného displeje seznamu relativně od levého horního okraje scény.

y

Y-ová pozice zobrazeného displeje seznamu relativně od levého horního okraje scény.

width

Šířka zobrazeného displeje seznamu v pixelech.

height

Výška zobrazeného displeje seznamu v pixelech.

visible

True, pokud je displej pro seznam zobrazený na scéně. False v opačném případě.

Příklad:

```
{
	"listName": "souřadnice X",
	"contents": ["-100", "-60", "-20", "20", "60", "100"],
	"isPersistent": false,
	"x": 297,
	"y": 37,
	"width": 168,
	"height": 268,
	"visible": true
}
```

3.2.6 Množina skriptů

V množině skriptů je skript uložen jako pole zapsané v JSON. Každý skript je zapsán jako pole o třech členech ve formě [x, y, příkazy].

x

Vzdálenost mezi levým okrajem skriptu a levým okrajem skriptové oblasti.

y

Vzdálenost mezi horním okrajem skriptu a horním okrajem skriptové oblasti.

blocks

Příkazy ve skriptu, zobrazené jako pole množin příkazů.

Například tato skriptová množina:

```
[712,409,
      [["procDef", "doleva", [], [], false],
      ["lookLike:", "convertible2"],
      ["glideSecs:toX:y:elapsed:from:", 0.3,
      ["-", ["xpos"], 40], ["ypos"]
]]]
```

vytvoří tento grafický blok příkazů:

Obr. č. 3:Ukázka definovaného příkazu

a zobrazí se 712 pixelů zprava a 409 pixelů dolů od horního levého okraje skriptové oblasti v editoru.

3.2.7 Množina příkazů

Množina příkazů ukládá příkaz jako JSON pole. První element pole obsahuje operační kód: textový řetězec, který identifikuje příkaz (textová reprezentace příkazu). Zbývající elementy pole obsahují argumenty příkazu.

Znakové hodnoty v numerických, textových nebo rozbalovacích vstupních slotech jsou uloženy jako čísla nebo řetězce.

Například:

```
["wait:elapsed:from:", 1]
["say:", "Hello!"]
```

Obr. č. 4

Vnořené příkazy jsou prezentovány jako množiny příkazů. Například:

```
["wait:elapsed:from:", ["*", 3, 2]]
```

viz. obr. 5

Obsahy příkazů ve tvaru C jsou zapsány jako pole množin příkazů. Například:

```
 ["doForever", [
      ["comeToFront"]
]]
```

viz. obr. 6

```
 ["doIf", ["mousePressed"], [
      ["say:", "yes"]
]]
```

viz. obr. 7

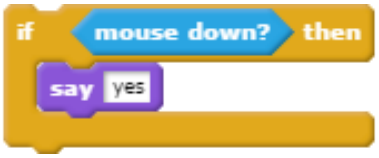
Když/jinak příkazy mají dvě pole, například:

```
 ["doIfElse", ["mousePressed"], [
      ["say:", "yes"]
], [
      ["say:", "no"]
]]
```

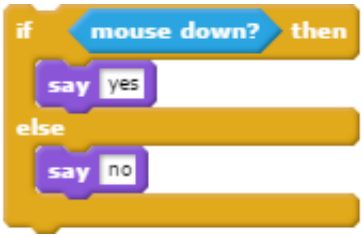
viz. obr. 8

Obr. č. 5

Obr. č. 6



Obr. č. 7



Obr. č. 8

3.2.8 Vlastní příkazy

Vlastní příkazy jsou interpretovány jako normální skripty. Procedura definování příkazu je daná formátem s klíči:

spec

Textový řetězec, který identifikuje popisky a vstupy do příkazu. Zřetelná slova produkují popisky v prototypu příkazu; %c vytváří otvor pro vstup určitého datového typu identifikovaného písmenem c; a %c.menuName přináší otvor pro vstup identifikovaného datového typu písmenem c a výběr možností typu identifikovaný díky menuName. Zde se nachází plný seznam vstupních datových typů:

inputNames

Pole názvů vstupů pro každý druh vstupního otvoru.

defaultValues

Pole výchozích hodnot pro každý druh vstupního otvoru.

atomic

True, pokud by měl příkaz probíhat bez obnovení scény. False v opačném případě.

Příklad:

```
["procDef", "pohyb %n %s %b", ["Number", "Text", "Boolean"], [1, "", false], true]
```

Obr. č. 9:Nově definovaný Vlastní blok se všemi parametry

Na obrázku můžeme vidět Vlastní příkaz nazvaný „pohyb“ se všemi defaultně pojmenovanými parametry typu číslo, text a booleovská hodnota.

3.2.9 Množiny komentářů

Množiny komentářů ukládají komentáře jako JSON pole. Každý komentář je zobrazen jako sedmičlenné pole. Forma takového pole:

```
[x, y, width, height, isOpen, blockID, text]
```

x

Vzdálenost mezi levým okrajem komentářového okénka a levým okrajem skriptové oblasti.

y

Vzdálenost mezi horním okrajem komentářového okénka a horním okrajem skriptové oblasti.

width

Šířka okna komentáře v pixelech.

height

Výška okna komentáře v pixelech.

isOpen

True, pokud je komentář zaplněný (rozšířený). False, pokud komentář není tak obsáhlý.

blockID

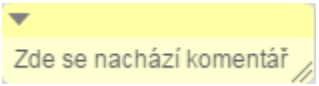
Index (v seznamu všech příkazů patřící k postavě nebo scéně, ke které komentář patří) v příkazu, ke kterému je komentář připoután. Pokud je hodnota tohoto klíče -1, jedná se o komentář, který není přichycen k žádnému příkazu.

text

Hodnota tohoto klíče obsahuje text komentáře.

Příklad:

```
"scriptComments": [[231, 24, 150, 50, false, -1, "Zde se nachází komentář"]]
```



Obr. č. 10: Ukázka komentáře

3.2.10 Objekt informací o projektu

Objekt informací obsahuje informace o projektu a studiu použitého pro jeho vytvoření. Nabývá těchto klíčů a hodnot:

scriptCount

Celkový počet skriptů v projektu.

videoOn

True, pokud je video zapnuté v momentu, kdy byl projekt uložen. Pokud je video vypnuté, hodnota klíče je false.

spriteCount

Celkový počet postav v projektu.

swfVersion

Verze Scratch studia, ve kterém byl projekt naposledy uložen.

flashVersion

Verze Flash Player pluginu, pomocí kterého bylo Scratch studio spuštěno.

hasCloudData

True, pokud projekt využívá cloudových dat. False, pokud projekt data nevyužívá.

userAgent

Textový řetězec znázorňující údaje o prohlížeči, ve kterém je Scratch studio spuštěno.

projectID

Podle tohoto Id můžeme najít projekt ve Scratch databázi.

savedExtensions

Doplňky, které byly importovány do projektu. Jsou znázorněné jako pole objektu rozšíření.

Příklad:

```
"info": {
	"videoOn": false,
	"flashVersion": "WIN 26,0,0,131",
	"hasCloudData": false,
	"userAgent": "Mozilla\/5.0 (Windows NT 6.1; Win64; x64
	"scriptCount": 9,
	"swfVersion": "v456",
	"projectID": "154650988",
	"spriteCount": 1
}
```

3.2.11 Objekt rozšíření

Objekt doplňku nese uložené informace o rozšíření. Obsahuje následující klíče a jejich hodnoty:

extensionName

Název rozšíření.

extensionPort

Port, na kterém běží rozšíření.

blockSpecs

Speciální příkazy z rozšíření uložené v poli.

3.2.12 Média

Mediální soubory (obrázky kostýmů, zvuky a obrázky pozadí) archivované v ZIP složce jsou pojmenovány sekvenčně od 0. Mají přípony .svg nebo .png pro obrázky a příponu .wav pro zvukové soubory.

Objekt kostýmu

V kostýmovém objektu se nachází speciální metadata pro kostým postavy nebo pozadí. Obsahuje následující klíče a hodnoty:

costumeName

Název kostýmu.

baseLayerID

Číslo odpovídající obrázkovému souboru v archivovaném ZIP souboru.

Poznámka: pokud není projekt stažen pomocí Scratch GUI, má hodnotu -1.

baseLayerMD5

Zmenšené informace o kostýmu pomocí MD5 hashovacího algoritmu, následované unicodovou značkou U+002E FULL STOP (.) a příponou souboru (obvykle png nebo svg).

bitmapResolution

V bitmapové verzi obrázku kostýmu; počet pixelů, které se nacházejí podél osy X samotného pixelu obrazu. Obvykle 1 nebo 2 když kostým zahrnuje i bitmapový text.

rotationCenterX

X-ová souřadnice centra kostýmové rotace. Relativní pozice k levému hornímu rohu obrázku.

rotationCenterY

Y-ová souřadnice centra kostýmové rotace. Relativní pozice k levému hornímu rohu obrázku.[{

```
      "costumeName": "convertible3",
      "baseLayerID": 1,
      "baseLayerMD5": "de371555505e068edeeaf55aede360de.svg",
      "bitmapResolution": 1,
      "rotationCenterX": 19,
      "rotationCenterY": 19
}]
```

Zvukový objekt

Zvukový objekt zahrnuje zvláštní metadata pro zvuk nacházející se u postavy nebo na jevišti. Najdeme zde následující klíče a hodnoty:

soundName

Název zvukové stopy.

soundID

Číslo odpovídající zvukovému souboru v archivovaném ZIP souboru.

Poznámka: pokud není projekt stažen pomocí Scratch GUI, má hodnotu -1.

sampleCount

Počet zvukových vzorků.

rate

Vzorkovací tempo zvuku.

format

Textový řetězec popisující zvukový formát. Obvykle prázdný řetězec.

4 Hodnocení projektů

Tato kapitola popisuje, jakým způsobem a postupy byly hodnoceny vybrané projekty. Hodnocení je vytvořeno z důvodu detekce vybraných prvků v projektu. Východiskem se mimo jiné stala hypotéza, že četnost a druh použitého příkazu často závisí na povaze a formě projektu. Zkoumané projekty poté můžeme podle hodnocených prvků lépe posuzovat a řadit.

Část hodnocení je zaměřena na pozorování výskytu logických a matematických operátorů. V další části pozorujeme využití řídících struktur (cykly, větvení), datových struktur a vlastních proměnných. V hodnocení jsou zařazeny pro jejich význam v programování. Znalosti a zkušenosti získané z práce s nimi se jistě uplatní v pozdější výuce programování a při tvorbě programů. Chceme zjistit, zda se v projektech tyto prvky objevují, popřípadě v jaké podobě a jak často.

Další část hodnocení je věnována tvorbě příkazů, které si autor projektu pro své potřeby vytvořil. Tyto příkazy si vytvoříme na paletě příkazů a posléze definujeme ve skriptovací oblasti. Pokud bychom vlastní příkazy přirovnali k některé části běžného programování, můžeme mluvit o funkcích, metodách a procedurách s nastavitelnými argumenty. [11] Zpravidla je jich v projektu využito za účelem omezení opakujícího se kódu a k jeho snadné úpravě. Důvodem zařazení mezi hodnocené prvky je, že toto počínání je pokládáno za obtížnější, vyžadující představivost a abstraktní myšlení.

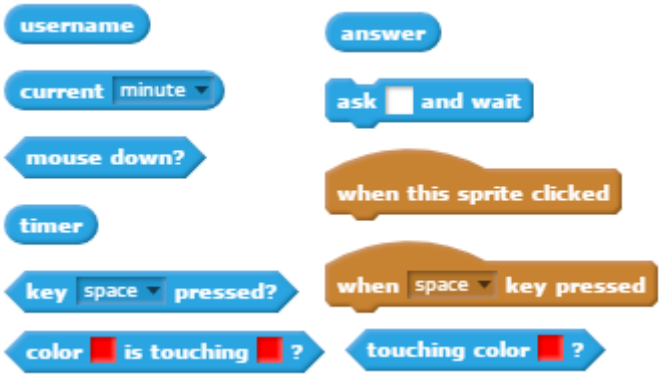
4.1 Interaktivita projektu

Jak často se můžeme setkat s použitím interaktivních příkazů?

V tomto hodnocení se věnujeme projektu po jeho interaktivní stránce. Interaktivitou rozumíme nejen vzájemnou komunikaci mezi uživatelem a projektem, ale také interaktivitu mezi objekty v programu. Uživatel, který zrovna používá program, může pomocí interaktivních prvků v projektu podnítit nové situace. Můžete použít klávesnici nebo myš, abyste rozpohybovali postavu, nebo můžete odpovídat na kladené otázky.

Hledané příkazy byly vybrány z kategorie Vnímání (světle modré) a Události (hnědé). Příkaz *username* oznamuje jméno uživatele, který zrovna program používá. *Current [minute,...]* vrací lokální časové hodnoty (sekundy, minuty, hodiny, měsíc, rok nebo datum) z uživatelova počítače. *Timer*, v českém překladu stopky, počítá reálný čas od jeho zapnutí. M*ouse down?* je příkaz detekující stisknutí tlačítka myši. Podobný je *key [space, ...] pressed?*, ten vnímá, zda je stisknuta vybraná klávesa. *Ask [] and wait* zobrazí text s políčkem pro odpověď, odpo-

věď se poté ukládá do proměnné *answer*. Příkazy z kategorie Události jsou určené pro zachycení události, která následně zapíná další skript. Zde jsou vybrány příkazy *when this sprite clicked* a *when [space, ...] key pressed*. Další příkazy jsou zobrazené na *obrázku č. 11*.

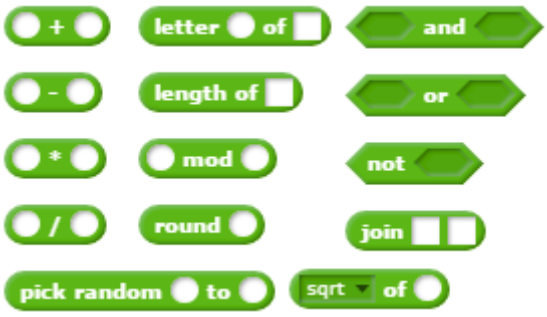


Obr. č. 11: Příkazy kategorie Interaktivita

4.2 Operátory

Jak často se můžeme setkat s použitím operátorů a logických příkazů?

Ovládání a vytváření programů nám značně zjednodušuje znalost matematických pojmů a možností jejich využití. Ve Scratch editoru máme při tvoření projektů na výběr z plné škály příkazů, které se dají považovat za hledané prvky. Těmi jsou příkazy z kategorie Operátory. Hlavním zájmem tohoto hodnocení je detekce těchto prvků v projektech a zjištění četnosti jejich využívání. Mezi Operátory patří sčítání, odčítání, násobení, dělení, modulo, zaokrouhlování, konjunkce, disjunkce, negace, náhodný výběr z množiny čísel, spojování řetězců, zjištění délky nebo pozice písmena řetězce a další matematické operace jako například mocniny.



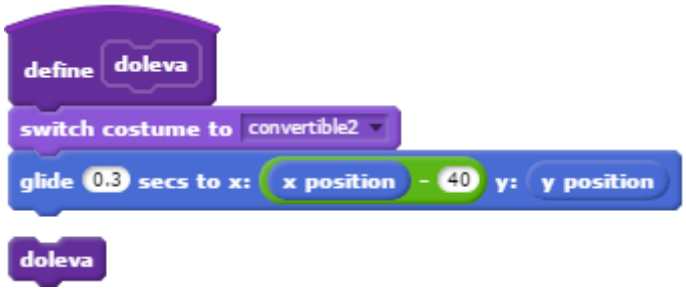
Obr. č. 12: Příkazy kategorie Operátory

4.3 Tvorba vlastních příkazů

Kolikrát si autoři vytvořili svou vlastní funkci, metodu, příkaz a k čemu ho využili?

Jako velmi zajímavá a užitečná možnost při tvorbě projektu se jeví vytváření vlastních příkazů. Je možné, že někdo tento prvek nepovažuje za příkaz, neboť spíše připomíná naprogramovanou metodu. Vlastní příkaz nám v první řadě pomáhá v lepší přehlednosti kódu a eventuálně v efektivitě jeho úpravy. Pokud například vytváříme více geometrických obrazců s pomocí cyklu, stačí nám definovat metodu n-úhelník a tu do daného cyklu vložit. Pokud bychom příkaz nedefinovali, značně by se snížila přehlednost kódu. Při nalezené chybě stačí, když ji upravíme jednou v našem příkazu. Tento nástroj by mohl posloužit ve výuce jako ukázka vnořených funkcí a vytváření vlastních metod a procedur.

Důvodem zařazení tohoto prvku do hodnocení je získání přehledu o četnosti a účelu jeho využití.



Obr. č. 13: Ukázka vlastního příkazu

4.4 Programovací prvky a řídící struktury

Každé seznamování s vybraným programovacím jazykem často prochází přes jeho syntaxi zápisu programovacích prvků a řídících struktur. Tyto prvky se dají označit jako nezbytné součásti tvorby programů. Z tohoto důvodu jsou zařazené mezi sledované a hodnocené části projektu. Řídícími strukturami myslíme cykly: nekonečné (*forever*), s pevným počtem opakování (*repeat*) nebo s podmínkou na začátku (*repeat until*). Další příkazy mohou skript pozastavit, dokud se nesplní podmínka (*wait until*) nebo ho zastavit i se všemi ostatními (*stop [all, ...]*). Dále můžeme využít klasického větvení (*if, if – else*). Řídící struktury patří do kategorie Ovládání, ve které nalezneme také příkaz umožňující klonování *when I start as a clone*.

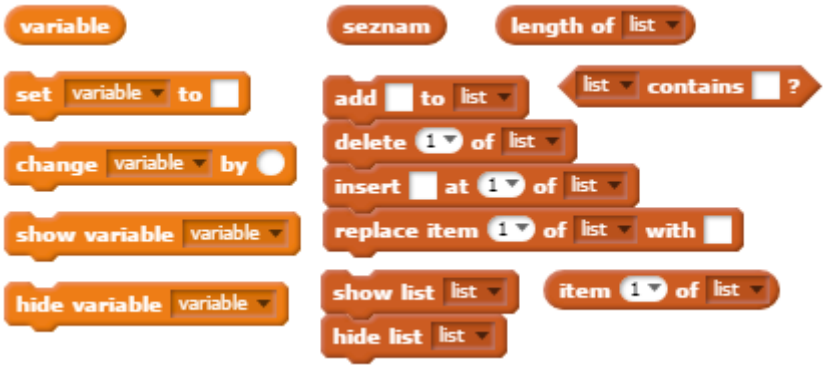
Obr. č. 14: Příkazy z kategorie Programovací prvky a řídící struktury

4.5 Využití datových struktur a vlastních proměnných

Jak moc ve zkoumaném vzorku autoři používají datových struktur (pole, seznamy) a vlastních proměnných?

Využití datových struktur a vlastních proměnných bylo pro hodnocení vybráno, neboť je důležité si osvojit tyto programové prvky a jejich znalost se může uplatnit v pozdější výuce programování a při tvorbě programů. Prostředky pro vytvoření těchto vlastních datových struktur nalezneme v kategorii příkazů s názvem Data. Při sledování využití těchto struktur nás zajímala jejich četnost a případy použití.

Příkazy, které se týkají tohoto prvku, si můžeme prohlédnout na *obrázcích 15* a *16*.



Obr. č. 15: Proměnná, list a ovládací příkazy

Obr. č. 16: Ostatní proměnné

5 Hairball – nástroj pro analýzu Scratch projektů

Automatický systém Hairball byl vyvinut týmem z University of California, Santa Barbara pro potřeby studentů i učitelů při hodnocení velkého množství Scratch projektů. Projekty jsou komplexní, složitá množina a při jejich hodnocení je zkoumáme nejen z programátorského pohledu, ale i pro jejich příběhový, grafický, vizuální a zvukový obsah. Tento systém se zaměřuje na jejich statickou analýzu. Dokáže studentovi oznámit, zda jeho kód obsahuje potenciální chyby nebo využívá programově nebezpečné praktiky. Pedagogovi asistuje při kontrolování implementace programů. Tím můžeme myslet bezpečné a silné programovací praktiky a také poskytnutí nástroje pro kontrolu bugů. Hairball je vhodný a užitečný nástroj především ve spojení s manuální analýzou. [12]

Kompletní zdrojový kód je dosažitelný jako open source podléhající zjednodušené licenci BSD, která umožňuje volné šíření obsahu, přičemž vyžaduje pouze uvedení autora a informace o licenci spolu s upozorněním na zřeknutí se odpovědnosti za dílo. Kód programu je hostován na webovém serveru Github na adrese *github.com/ucsb-cs-education/hairball*. Zde se nachází jeho popis a základní instrukce související s jeho instalací a použitím. Poslední verze tohoto programu je uvedena jako v0.3 a datována ke 4. srpnu 2014. [12],[13]

Zjednodušeně můžeme tento program nazvat analyzačním nástrojem, který prochází řádky zdrojového kódu projektu. V tomto kódu nachází příkazy použité v projektu, zpracovává je a podle funkce zadaného pluginu vrací na výstupu potřebná data.

5.1 Instalace

Tento aplikační rámec je napsán v jazyce Python verze 2.7. Je tedy nutné mít na svém zařízení nainstalovanou tuto platformu. Verze nainstalovaného Python prostředí musí obsahovat systém správy balíčků `pip`, poté se instaluje hairball jednoduše přes konzolovou řádku příkazem `pip install hairball`.

Pokud jsme si stáhli soubory se zdrojovým kódem a všemi ostatními přídavnými soubory, musíme se s příkazovou řádkou dostat do vnější složky hairballu, kde se nachází soubor *setup.py*. Instalaci poté spustíme příkazem `python setup.py install.` [13]

5.2 Spuštění a použití

Po úspěšné instalaci můžeme začít s analyzováním projektů. Na stránce nebo v souboru, kde je tento program uložen, otevřeme soubor s názvem *README.txt*. Zde nalezneme názvy

pluginů a jejich správný zápis. Spustíme si příkazový řádek, v něm nápovědu a informace k použití programu příkazem `hairball -h` nebo `hairball --help`. Tím zjistíme, jak má vypadat celý příkaz, který obsahuje spuštění programu Hairball, název pluginu a cestu k analyzovanému projektu.

V této práci je Hairball využit jako nástroj pro zpracování dat ze zkoumaných projektů. Vybrán byl z důvodu potřeby zjistit výčet použitých příkazů. K tomu byl zapotřebí plugin `blocks.BlockCounts`. Jedná se o automatickou statickou analýzu projektu a její výstupy jsou součástí několika bodů v hodnocení projektu.

Pro zjištění počtu všech použitých příkazů v projektu Hry.sb2 napíšeme tento příkaz:

```
$ hairball -C -p blocks.BlockCounts Hry1.sb2
Hry1.sb2
  1 stop %s
  2 touching %s?
  2 move %s steps
  2 change y by %s
  3 forever%s
  4 say %s
  4 say %s for %s secs
  4 touching color %s?
  5 when this sprite clicked
  5 repeat %s%s
  6 stop all sounds
  6 if %s then%s
  6 next costume
  7 switch backdrop to %s
  8 show
  9 switch costume to %s
  9 when @greenFlag clicked
 10 play sound %s
 10 go to x:%s y:%s
 10 when %s key pressed
 11 when backdrop switches to %s
 15 hide
 16 wait %s secs
 33 glide %s secs to x:%s y:%s
188 total
```

Obr. č. 17: Ukázka výstupu Hairballu v příkazové řádce

číslo zde označuje počet použití příkazu, následuje anglický název příkazu, %s označuje místo pro parametr v příkazu. Na konci výpisu se nachází celkový počet příkazů.

Ukázka všech možných zásuvných modulů:

- `blocks.BlockCounts` – vrací všechny příkazy použité v projektu a jejich počet
- `blocks.DeadCode` – plugin, který má zjistit nespustitelný script
- `checks.Animation` (není plně testováno) – tento plugin zjišťuje „komplexní animaci“, script animace má obsahovat cyklus, pohyb, časování a změny kostýmů
- `checks.BroadcastReceive` – zjišťuje, zda byla správně implementována komunikace prostřednictvím příkazů broadcast a receive

- `checks.SaySoundSync` (není plně testováno) – zjištění synchronizace mezi příkazem say a zvukovou stopou
- `duplicate.DuplicateScripts` – tento plugin detekuje duplikátní scripty
- `initialization.AttributeInitialization` – kontrola řádné inicializace modifikovaných atributů (pozadí, viditelnost, orientace, pozice, …), pokud program obsahuje více příkazů „po kliknutí na zelenou vlajku“ a oba upravují jeden atribut na jinou hodnotu, tento atribut je považován jako nespustitelný
- `initialization.VariableInitialization` (není plně testováno) – podobné jako předchozí, za atributy jsou zde považovány vytvořené proměnné.

Hairball můžeme použít na jeden projekt, nebo na celé sady projektů. Musíme však počítat s větší časovou prodlevou při parsování většího množství. [13]

6 Jak vyhledat a vybrat Scratch projekty?

Jak již bylo řečeno, velikost vzorku byla určena na celkem 300 projektů. Nejdříve bylo nutné vyhledat a seznámit se s nástroji používanými na výběr projektů. Jak jsme dosáhli z celkového množství 22 miliónů projektů relevance výsledků? Kromě odhlédnutí od trendových témat jsme využili nástroje ve Scratch a jeho sdruženém okolí. V následujících podkapitolách se zmíníme o dvou nejběžněji využívaných.

6.1 Scratch API

Díky rozhraní Scratch API máme přístup k datům týkajících se projektů, uživatelů, galerií a statistik bez použití webové stránky Scratch. Rozhraní je vytvořeno pro usnadnění přístupu programovacích jazyků k datům, které je poté dokáží zpracovat ve vytvořených programech. [14] Mohou to být například různé překladače a statistické nástroje. K informacím přistupujeme na stránce *https://scratch.mit.edu/api/[apiname]/[parameters],* v odkazu doplníme požadovanou akci místo *[apiname]* a její parametry do *[parameters]*.

Například tento odkaz vyhledá projekty s českou lokalizací:

https://api.scratch.mit.edu/search/projects?language=cs&offset=20

Problémem je, že zde nelze nastavit parametr popularity, a tak jsou projekty vyhledávané od nejpopulárnějšího. Z tohoto důvodu je tento nástroj pro naši práci nepoužitelný.

6.2 Scratch Explorer

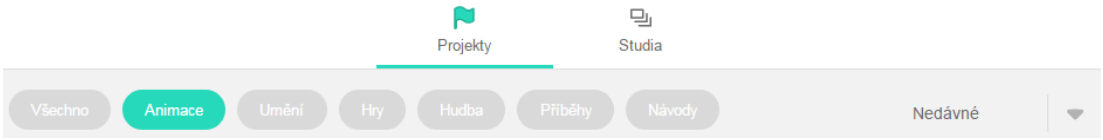
Druhým vyhledávacím nástrojem je Scratch Explorer, tento vyhledávač umožňuje uživateli bádat v nepřeberném množství projektů. Současná funkce je napájena službou Custom Search Google Engine (CSE) a na základě datové analýzy pojme stránka více než 154 milionů přístupů za měsíc. [15]

Důležitá je možnost jeho nastavení. Nastavení se týká parametrů a tematických skupin projektů. Tento průzkumník nám dává na výběr z 6 skupin / témat. Těmi myslíme Hry, Animace, Příběhy, Návody, Umění a Hudbu. [16]

Po výběru skupiny se přesouváme k dalším parametrům. Tyto parametry vyhledávání se týkají vlastností projektů a jsou celkem 3. První parametr má název Popular, ten vyhledá projekty od nejpopulárnějšího v daném tématu. Další parametr se nazývá Trending. S tímto nastavením se vyhledají parametry podle určitého trendu, který se dá vypozorovat. Jako poslední možnost nám zbývá parametr Recent, tedy nedávno sdílené projekty. Tento parametr můžeme označit

za jedinou známou možnost vyhledávání, při které zachováme relevanci výběru tvořených projektů, a proto byl pro tuto práci také zvolen. Tímto se vymezíme od projektů, které jsou tvořeny záměrně pro podporu komunity, moderovaných projektů a dalších pro tento výzkum nevhodných projektů.

Do vyhledávače lze zadávat hesla, názvy, tagy, témata a jiné výrazy. Po zadání hesel nebo názvů nelze nastavovat parametry.



Obr. č. 18: Scratch průzkumník

Při výběru bylo také nutno být obezřetný ohledně projektů, které neměly označení remixu, tedy odkaz na původní projekt v popisku, ale bylo zřejmé, že jsou pouze kopiemi. Na tyto neoriginální projekty můžeme v databázi narazit poměrně často.

Samotný výběr probíhal tak, že nejdříve byla zvolena tematická skupina. Výběr skupin byl motivován zkoumáním možností vyhledávání projektů. Bylo potřeba nalézt takové projekty, které by se daly tematizovat do několika uskupení. U těchto skupin byl zvolen parametr Recent. Poté se postupovalo shora dolů. Každý projekt, kromě zmíněných kopií a prázdných projektů (tyto projekty neobsahovaly žádný předmět výzkumu nebo byly v defaultním nastavení), byl posléze zpracován a hodnocen.

7 Zapisování dat

Jako nejvhodnější program pro zapisování dat byl zvolen Microsoft Excel. V každém projektu je možné využít až 145 příkazů z celkem 10 kategorií. Proto byla v Excelu vytvořena šablona, která obsahovala 10 skupin sloupců, obarvených jako kategorie příkazů ve Scratch a v nich se nacházelo 145 sloupců pojmenovaných jako příkazy. Dále se shromažďovala základní data o projektu jako je název, jméno autora, odkaz na projekt a název skupiny, ve které byl projekt vyhledán. Dále se zapisoval počet unikátních pozadí a postav v projektu. U postav se také zapisoval průměrný počet kostýmů. Zvláštní sloupce náležely kategorii *Data*, tedy v projektu vytvořených polí a jedinečných proměnných, a vlastním příkazům. U kategorie *Data* se udával počet proměnných, seznamů a zapisoval se počet použitých příkazů pro práci s těmito datovými strukturami. V kategorii vlastních příkazů se zapisovalo, zda byl vlastní příkaz vytvořen, popřípadě jejich počet.

8 Tematické skupiny zkoumaných projektů

Jak bylo zjištěno, ve Scratch se nachází 6 hlavních tematických skupin projektů.

Tyto skupiny mají za úkol zachytit charakter projektů tak, aby bylo možné je porovnávat jednotlivě uvnitř skupin, ale i vně s ostatními skupinami. Uvnitř skupin byl kladen největší důraz na zjištění nejvíce a naopak nejméně používaných příkazů, poté pozorování trendů a podobnosti bloků z těchto příkazů vytvořených, dále pak z jakých kategorií byly příkazy vybrány atd. Na základě těchto zjištění bylo možné porovnávat data jednotlivých skupin mezi sebou. Tyto porovnání nám poskytly náhled například na mediány sum použitých příkazů v jednotlivých skupinách projektů, podíl kategorií příkazů na celkovém počtu všech příkazů a také celkové sumy využití interaktivních příkazů ve zkoumaných skupinách projektů jakožto jedné části hodnocení projektu.

8.1 Animace

Tuto tematickou skupinu můžeme definovat projekty, ve kterých se autoři snaží o vizuální oživení věcí a obrazů. Projektů vytvořených ve smyslu animace je ve Scratch databázi velmi mnoho. Je to jeden z nejčastějších druhů projektů, lze v nich najít velké množství různých animací jako třeba 3D animace, animovaná hudební videa, rozpohybování obrazu pomocí jeho rozdělení na proužky tzv. Barrier Grid Animation, fázové animace a v neposlední řadě rychlé kreslení. [17]

8.2 Hry

Nejčastějším motivem projektů jsou hry. Definujme je jako program, který reaguje na činnost uživatele a vyžaduje jeho pozornost k dosažení cíle. Hry se dají zařadit mezi nejsložitější a nepropracovanější projekty. Díky již zmiňovanému velkému výběru příkazů lze ve Scratch tvořit nové a originální hry, ale také parodie a emulované legendární hry jako je například Mario, Pacman nebo Pong. V projektech patřících do této skupiny se můžeme setkat například i s aplikovanou herní fyzikou a speciálními enginy.

Nejběžnější typy her jsou tzv. platformer hry, ve kterých se postavička, kterou ovládáme, pohybuje po platformách. Rolovací hry, které dokáží zvětšit hrací plochu pomocí předem nakreslených částí, které se vedle sebe skládají jako při rozkládání mapy. Jako další můžeme zmínit RPG hry. U tohoto druhu nejčastěji ovládáme postavičku, která musí splnit nějaký úkol nebo zneškodnit nepřátele. Oblíbená je také tvorba bludišť, labyrintů a tycoon her. [18]

8.3 Umění

V projektech ze skupiny Umění můžeme pozorovat, že zde nejde tolik o umění programovat, jsou založené spíše na propagaci grafické tvorby autorů a projekty fungují jako jejich galerie. Obrazy v galerii pak prohlížíme pomocí šipek na klávesnici. Autoři zde pozadí používají jako plátno a kreslí na něj pomocí bitmapové nebo vektorové grafiky. Zdá se, že se pokoušejí o maximální využití kreslících nástrojů ve Scratch a soupeří mezi sebou o nejlepší dílo. Tato skupina má mnoho společného se skupinou Animace. [19] Odpovídají tomu i tagy u projektů, které většinou nesou názvy těchto dvou skupin.

8.4 Hudba

Na příkazové paletě v editoru se nachází mnoho pokynů, které se dají použít při tvorbě a ovládání zvukové stopy. Tvůrci mohou využívat zvukovou banku MIDI, která je nainstalovaná přímo ve Scratch editoru. Obsahuje 128 nástrojů, které lze dále přizpůsobovat pomocí tempa a hlasitosti. Jiné možnosti jsou import vlastní zvukové stopy a také nahrání zvuku pomocí mikrofonu. V této skupině najdeme projekty, ve kterých je nám umožněno ovládat celý orchestr pomocí kláves, komponovat vlastní hudbu nebo si pouštět výběr importovaných písniček. [20] Jedná se o jedny z nejinteraktivnějších projektů, které byly v této práci zkoumány. I to je jeden z důvodů, které tuto skupinu řadí mezi první v oblíbenosti a množství vypracovaných projektů.

8.5 Příběhy

Do této skupiny řadíme tzv. příběhové projekty, ve kterých jsme svědky příběhu, vypravování a historek. [21] V této skupině se můžeme setkat s mnoha způsoby vytváření a druhy projektů. Jako hlavní je třeba zmínit interaktivní příběhy, které nás vtáhnou do děje pomocí interaktivních bloků, tážou se nás na otázky a sami si volíme cestu příběhem. Také se nás postavy mohou ptát na otázky a pro další postup v příběhu musíme správně odpovědět. Často se s postavami posouváme do jiných oblastí pomocí změn pozadí. Za další druh projektů v této skupině můžeme označit talk show, kdy spolu postavy v projektu komunikují. Spouštět komunikaci je možné po kliknutí na postavu, předmět či vlajku. Posledním z hlavních typů příběhů jsou knižní příběhy a krátké povídky. U těchto projektů se setkáváme s textem na pozadí a stránky obracíme pomocí šipek nebo jiných kláves či předmětů. Takovéto projekty jsou často doplňovány hudbou na pozadí a průběžnými animacemi. Celkově projekty z této skupiny řadíme mezi méně složité a příkazy jsou poskládány často sekvenčně.

8.6 Návody

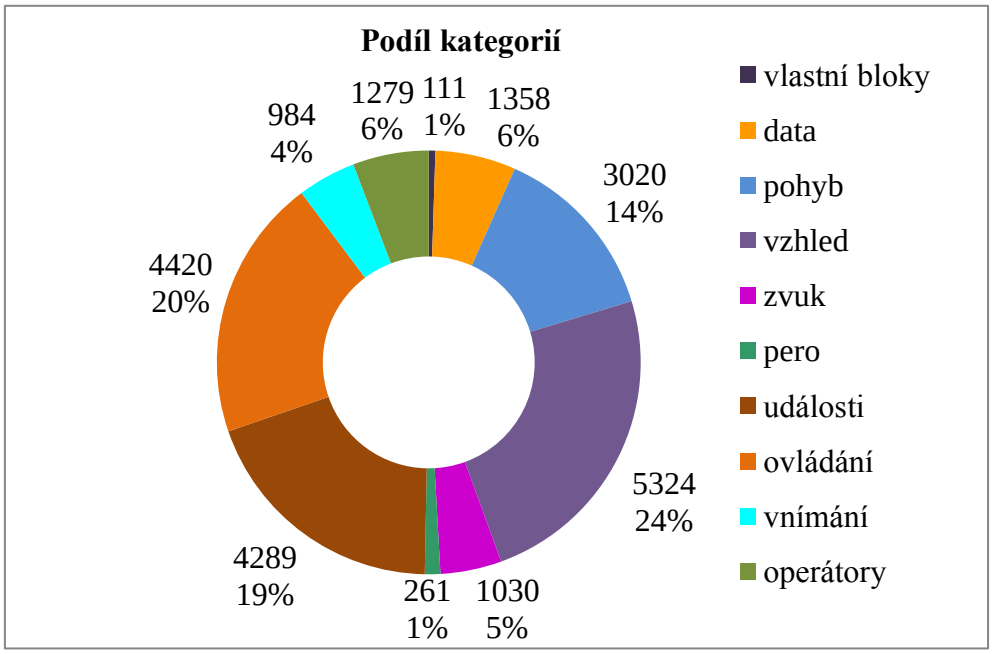
Projekty z této skupiny jsou navrhované jako příručky a uživatele učí jak provést určité úkoly a zadání nebo něco vytvořit. Návody se mohou, ale nemusí, týkat výhradně Scratch. Můžeme se setkat například s instrukcemi pro využívání různých technik kreslení. Na ty navazují příručky pro grafické vytváření figur. Další návody se mohou týkat technik programování a využívání příkazů ve Scratch. Jejich součástí jsou často předpřipravené části kódu a šablony pro další využití. [22] Poměrně rozšířené jsou projekty věnující se návodům k vytvoření úvodních pasáží projektů. Při vysvětlování objemného tématu je běžným jevem jeho rozdělení do více instruktážních projektů a jejich opětovné sloučení v jednom studiu. Příkladem takového počínání nám mohou posloužit studia pro vysvětlení funkcí příkazů, výuka cizích jazyků, vytváření rozmanitých typů her, animací, rozličných enginů (padající vločky, gravitace apod.) a návody pro mnoho dalších činností, které můžeme pomocí Scratch provádět.

9 Výsledky práce

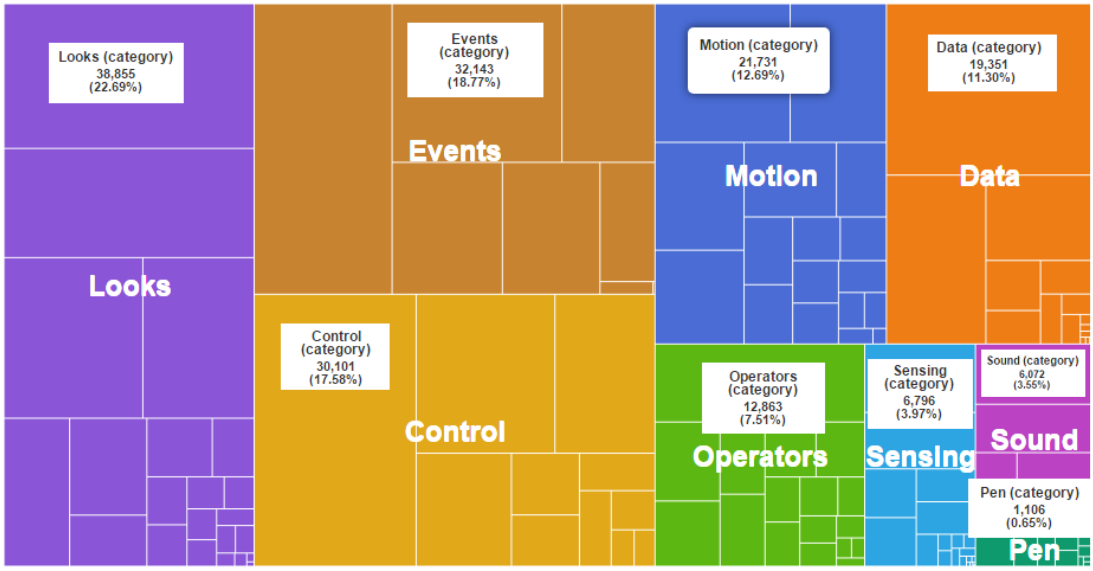
Do této kapitoly jsou zařazeny všechny významné výsledky. V první podkapitole můžeme vidět zastoupení, procentuální podíl a počet příkazů všech kategorií v celkovém množství. Pod tuto kapitolu dále patří podkapitoly *20 nejvíce a nejméně používaných příkazů*. Další významné výsledky nacházíme v podkapitole *Hodnocení*, která se dále dělí na výsledky s názvy všech hodnocených témat (*Interaktivita, Využití operátorů, Tvorba vlastních příkazů, Programovací prvky a řídící struktury, Proměnné a datové struktury*). V závěru se nachází podkapitola s názvem *Dílčí výsledky*. V té nalezneme grafy, které znázorňují, jaká je střední hodnota nevyužitých příkazů ve skupinách a dále například střední hodnotu sum příkazů použitých v projektech zkoumaných v rámci tematických skupin.

9.1 Srovnání celkových výsledků

Jedna z důležitých otázek, kterou jsme si na začátku tvorby této práce položili, zněla, do jaké míry budou korespondovat data této práce s oficiálními statistikami? Pozitivním výsledkem práce je, že výsledky opravdu korelovaly. Pro srovnání zde máme graf, ve kterém jsou barevně znázorněny kategorie příkazů. U těchto kategorií poté vidíme číslo, které uvádí počet použitých příkazů patřících do těchto kategorií a procentuální podíl kategorií v celkové sumě. Nejvýznamnějšími kategoriemi se jeví Vzhled, Ovládání, Události a Pohyb. Jako doplňkové můžeme označit Data, Pero, Zvuk, Vnímání. Nejméně využívané jsou příkazy z kategorie Operátorů. Jako samostatnou kategorii uveďme Vlastní bloky. Této kategorii se podrobněji věnuji v kapitole *10.2.3*.



Graf č. 1: Koláčový graf podílů kategorií



Obr. č. 19: Oficiální statistika [23]

Výrazněji se liší pouze kategorie Data a Ovládání. Rozdíl mezi kategoriemi Ovládání lze vysvětlit menším vzorkem projektů. Na druhou stranu, pokud porovnáme společně kategorie Ovládání a Události v této práci a v oficiální statistice, lze hovořit o téměř totožném trendu.

9.1.1 Nejvíce využívané příkazy

Jako zajímavý výsledek výzkumu je nutné zmínit nejvíce používané příkazy. V tomto výstupu jsou zahrnuta celková data ze všech analyzovaných projektů. Pro přehlednost je zde popis grafu ucelen do jedné formy: Kategorie (barva; počet příkazů z této kategorie), příkazy Název (číslo udávající jeho výskyt v počtu projektů).

Z grafu číslo 2 můžeme poznat, že mezi nejvíce využívanou kategorii patří Vzhled (fialové; 6) s příkazy Ukaž se/ Skryj se (133), Změň kostým na (116), Další kostým (90), Změň pozadí (89), Říkej (87) a Nastav efekt (60).

Příkazy z kategorie Ovládání na druhém místě vzhledem k využitým příkazům (oranžové, 4), do které patří příkazy: Čekej [] sekund (193), Opakuj dokola (170), Opakuj [] krát (98) a příkaz Když [] tak (94).

Dále můžeme pozorovat příkazy z kategorie Události (hnědé, 4), příkaz z této kategorie Po kliknutí na vlajku (255) je uveden na prvním místě. Další příkazy patřící do této kategorie se nazývají Po stisku [] klávesy (113) a shodně Rozešli všem [a čekej] (81) s Po obdržení zprávy [výběr] (81).

Zastoupení v tomto grafu má i kategorie Pohyb (modrá, 2) s příkazy Skoč na pozici x[] y[] (124) a Posuň se o [] (71).

Předposlední nejvyužívanější příkazy náleží do kategorie Zvuk (tmavě růžová, 2) s příkazy Přehraj zvuk (84) a Hraj zvuk dokonce (73).

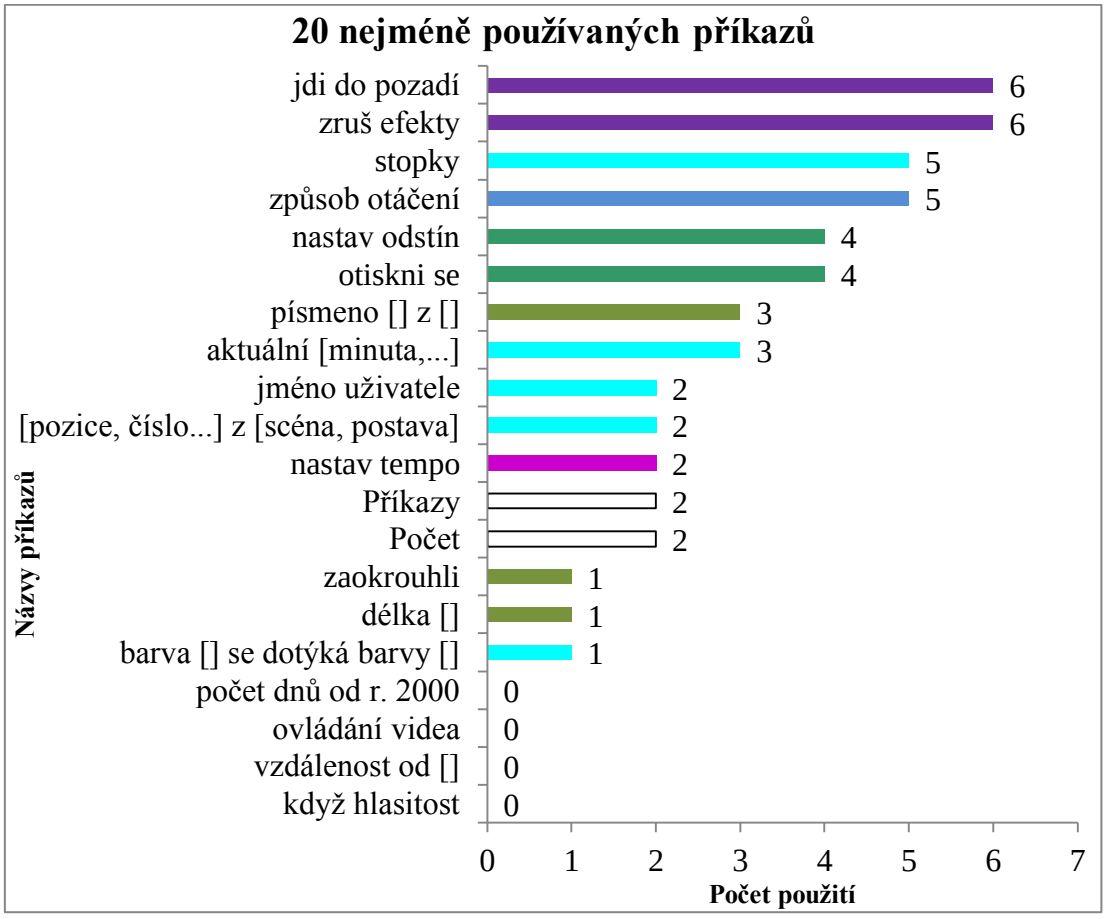
Jako poslední zmiňme kategorii Data (žlutá). Nejvýše umístěný příkaz z této kategorie nese název Počet (79) a udává počet vytvořených unikátních proměnných. Druhý příkaz s názvem Změň / Nastav (75) je souhrnným názvem pro příkaz určené k ovládání proměnných, tedy pro změnu a nastavení jejich hodnot.

Graf č. 2: 20 nejpoužívanějších příkazů

9.1.2 Nejméně využívané příkazy

Jak už z názvu kapitoly vyplývá, budeme se věnovat nejméně používaným příkazům. Data, která nám graf ukazuje, můžeme brát jako protiváhu ke grafu č. 2. Jako kategorii s největším výskytem příkazů zde najdeme kategorii Vnímání, konkrétně 8. V pěti projektech byl využit příkaz Stopky, ve třech příkaz *Aktuální [sekunda, minuta, hodina]*, každý ve dvou příkazy Jméno uživatele, *[pozice, číslo…] z [scéna, postava]*, jedenkrát *Barva [] se dotýká barvy []* a tři příkazy, které nebyly použity v celé množině zkoumaných projektů ani jednou: *Počet dnů od r. 2000*, *Ovládání videa*, *Vzdálenost od []*. Na vrcholu tohoto grafu můžeme pozorovat dva příkazy z kategorie Vzhled, každý byl nalezen v šesti projektech. Jejich názvy jsou: *Jdi do pozadí* a *Zruš efekty*. Další z nejméně používaných příkazů náleží do kategorie Pohyb, byl použit celkem v pěti projektech a nazývá se *Způsob otáčení*. Následují dva příkazy z kategorie Pero, konkrétně *Nastav odstín* a *Otiskni se*, oba byly shodně použity v projektech čtyřikrát. Dále v pořadí se nachází více početná skupina příkazů z kategorie Operátorů. Příkaz *Písmeno [] z []* byl nalezen třikrát, jedenkrát *Zaokrouhli* a stejně tak i *Délka []*. Zastoupení v tomto grafu má i kategorie Zvuk. K této kategorii náleží příkaz *Nastav tempo*, nalezen ve

dvou projektech, a *Když hlasitost*, nenalezen ani jednou. Bílou barvou a černým ohraničením jsou označeny příkazy týkající se implementace seznamů a příkazů pro jejich ovládání a kontrolu. Tento graf nám říká, že z 300 projektů se seznamy objevily jen ve dvou případech.



Graf č. 3: 20 nejméně používaných příkazů

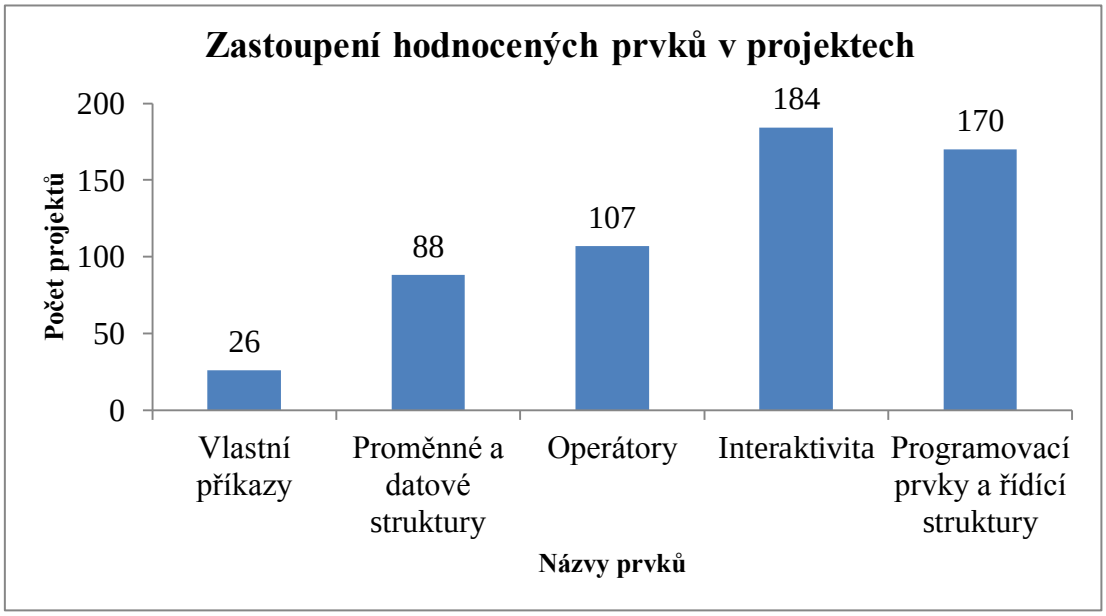
9.2 Výsledky hodnocení

V této kapitole se seznámíme s výsledky hodnocení projektů. Na projektech jsme zkoumali, jak velké nebo malé mají tyto prvky zastoupení a díky tomu si dokážeme vytvořit ucelený pohled na tuto problematiku. Nejdříve si ukážeme celkové hodnocení a srovnání prvků mezi sebou. Poté přijde na řadu pohled do těchto prvků a srovnání počtu výskytů příkazů patřících k jednotlivým částem hodnocení.

9.2.1 Zpracovaná data

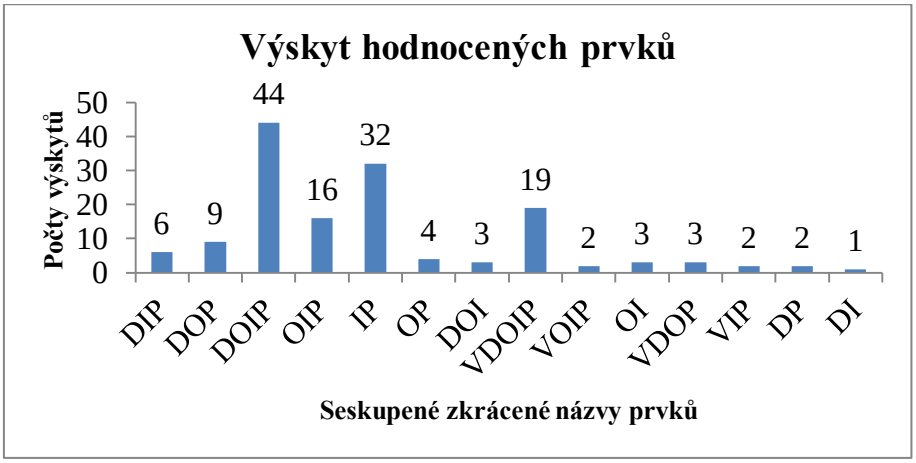
Na následujícím grafu můžeme pozorovat všechny hodnocené kategorie. Více než polovina projektů, celkem 184, obsahovala jeden nebo více interaktivních prvků. Další hodnocený prvek *Programovací prvky a řídící struktury* byl použit v celkem 170 projektech. I zde se stále držíme nad polovinou. Hodnocený prvek *Operátory* už se objevuje jen o málo více než v jed-

né třetině celkového počtu projektů (107). Podobné zastoupení můžeme pozorovat i u *Proměnných a datových struktur*. Ty se objevují v 88 projektech. A poslední prvek *Vlastní příkazy* byl zachycen jen v 26 projektech z 300 hodnocených.



Graf č. 4: Zastoupení hodnocených prvků

Za pozornost stojí i další grafické zpracování dat. V grafu č. 5 můžeme vidět 5 různých písmen. Tato písmena označují hodnocené prvky, tedy *Interaktivita* (I), *Operátory* (O), *Vlastní příkazy* (V), *Programovací prvky a řídící struktury* (P) a *Proměnné a datové struktury* (D). Jednotlivá písmena jsou dále seřazena do uskupení podle toho, jak se hodnocené prvky objevovaly v projektech. Pokud se například objevil prvek I a O ve stejném projektu, utvořili tak dvojici písmen IO.



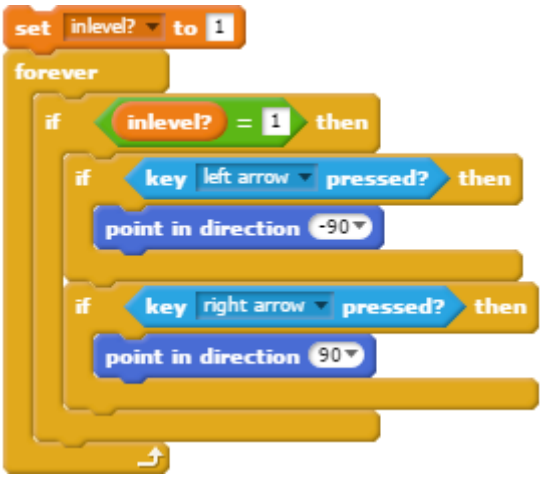
Graf č. 5: Seskupené prvky podle výskytu

Tento graf nezobrazuje prvky, které byly v projektech samotné. Jde nám totiž o zjištění, jaké prvky byly využívány společně a jaké spolu tvořily kombinace. Můžeme si všimnout, že v 11

případech ze 14 tvoří nějakou skupinu prvek P, včetně nejpočetnější DOIP, která byla zaznamenána celkem ve 44 projektech. Následuje ho prvek I. Ten se objevuje v 10 ze 14 uskupení a společně s prvkem P tvoří druhou nejhojnější skupinu IP zaznamenanou ve 32 projektech. Na 9 ze 14 nenulových uskupení participuje prvek O. S předchozími prvky I a P tvoří čtvrtou nejpočetnější skupinu OIP. Všechny prvky se v jednom uskupení sešly celkem devatenáctkrát.

9.2.2 Závěry analýzy

Z dostupných dat může vyvodit zajímavé závěry. Nejpočetněji se hodnocené prvky objevují pospolu ve skupině DOIP (tedy *Data, Operátory, Interaktivita a Programovací prvky*). Projekty, které v sobě ukrývaly tyto 4 prvky, se nejvíce vyskytovaly ve skupině Hry a patřily mezi nejpropracovanější. Příklad si uveďme na příkazu ze sofistikovaného projektu *Samus Vector Demo and Bosses (Rebound)* od autora David_Sh. [24] V tomto příkazu, který je vystřihnut ze složitějšího bloku, se promítají všechny prvky ze skupiny DOIP.



Obr. č. 20: Příklad 4 prvků pohromadě

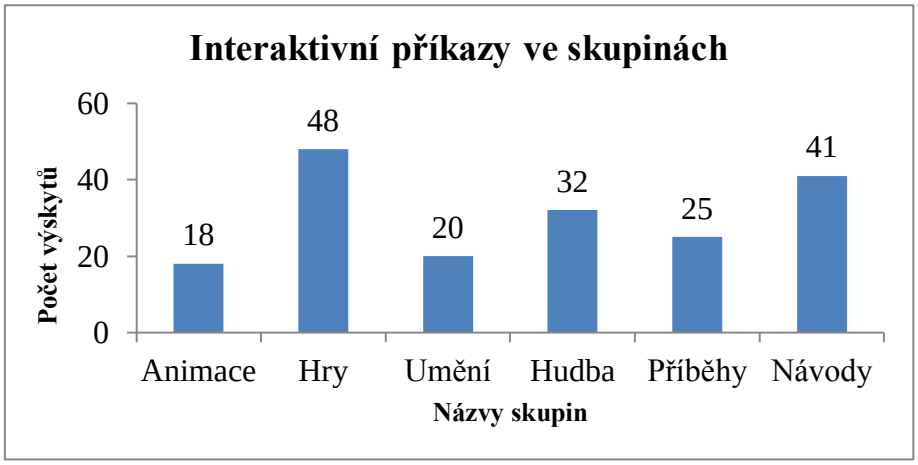
9.3 Interaktivita

Interaktivní prvky jsou dostupné v téměř každém projektu. Míra popularity projektu často souvisí s interaktivitou a zábavným zpracováním projektu. Autoři se tedy snaží těchto prvků komponovat co nejvíce. Ostatně paleta příkazů, filosofie Scratch a forma tvorby v editoru je k tomu sama vybízí.

9.3.1 Zpracovaná data

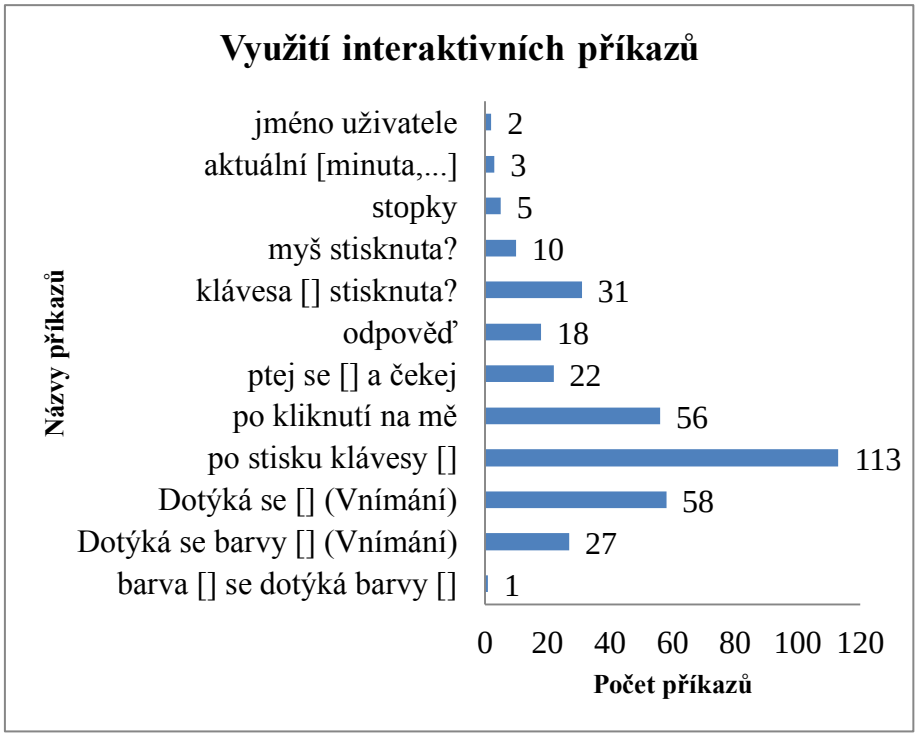
Analýza projektů nám objasnila, v jakém množství jsou používány Interaktivní příkazy a v jakých skupinách jsou nevíce zastoupeny. Graf č. 6 ukazuje, že v téměř většině (48 z 50) projektů ze skupiny *Hry*, se objevuje nějaký interaktivní příkaz. Další na tyto příkazy bohaté

skupiny nesou názvy *Návody* a *Hudba.* Jen polovina projektů ze skupiny *Příběhy* obsahovala některý z interaktivních příkazů. V ostatních skupinách *Animace* a *Umění* byly tyto příkazy přítomné už v méně než polovině projektů.



Graf č. 6: Rozložení interaktivních prvků ve skupinách

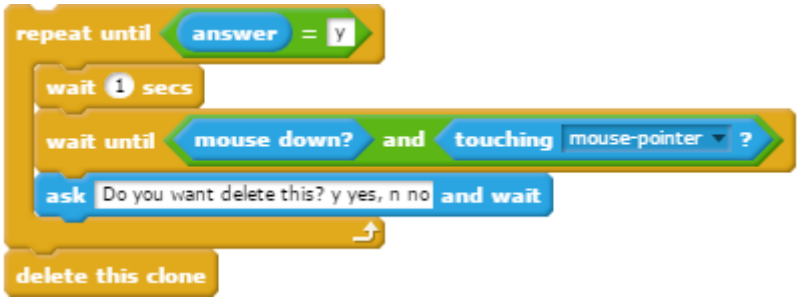
Jak na tom jsou tedy konkrétní příkazy? Z následujícího grafu můžeme poznat, že nejvíce využívaný příkaz je *Po stisku klávesy []* (113). Zhruba dvakrát méně byly využity příkazy *Dotýká se []* (58) a *Po kliknutí na mě* (56). *Příkaz Klávesa [] stisknuta?* byl použit celkem ve 31 projektech, dále *Dotýká se barvy []* (27) a *Ptej se [] a čekej* (22). Dále následují příkazy *Odpověď* (18), *Myš stisknuta?* (10), *Stopky* (5), *Aktuální [minuta, ...]* (3) a nejméně využívaný příkaz *Jméno uživatele* (2).



Graf č. 7: Interaktivní příkazy

9.3.2 Závěry analýzy

Podle dostupných dat a údajů jsou příkazy z kategorie *Interaktivita* nejvýraznější v počtu použití. Jsou velmi rozšířené v každé skupině projektů. Nejvýznamnější skupinou jsou Hry. Téměř v každém projektu z této skupiny se objevuje jeden a více interaktivních příkazů. Příkaz *Po stisku klávesy []* je s přehledem vůbec nejvyužívanějším příkazem ze všech hodnocených kategorií (celkově je sedmý nejpoužívanější). Nejvíce výskytů tohoto příkazu bylo zaznamenáno v projektu *Stories* od autora Eleanor_2003. [24] Největší suma použitých příkazů byla objevena v projektech *GAMES Version: 1.2* od autora RedditGames [26] a *Carbon Mechanic* od Friefree. [27] Interaktivita se dá považovat za nejvíce využívaný prvek při tvorbě projektů ve Scratch a je pro projekty nejzásadnější.



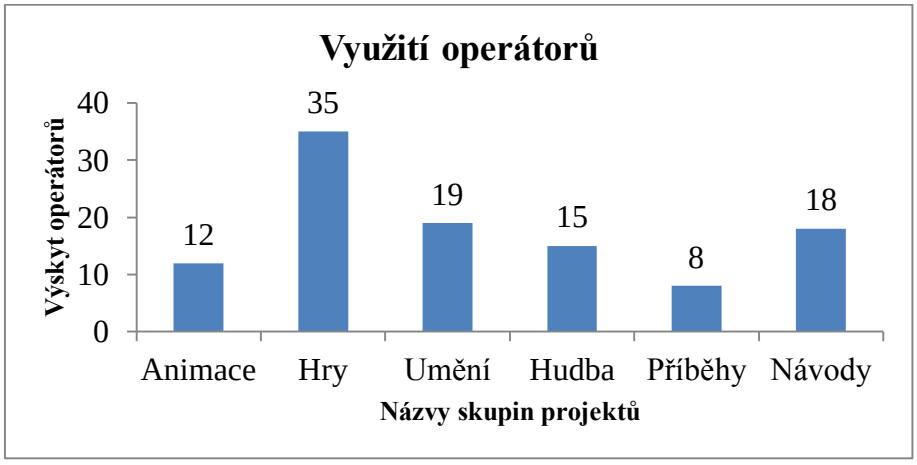
Obr. č. 21: Ukázka z Carbon Mechanics

9.4 Využití operátorů

Operátory jsou třetí nejvíce využívanou kategorií. Z této třetí pozice však mají blíže k posledním dvěma. Od prvních dvou je dělí téměř ¼ z celkového počtu. Nejbližší kategorií v počtu použitých příkazů jsou *Proměnné a datové struktury*. Celkem jsou příkazy z kategorie *Operátory* použity ve více než 1/3 projektů.

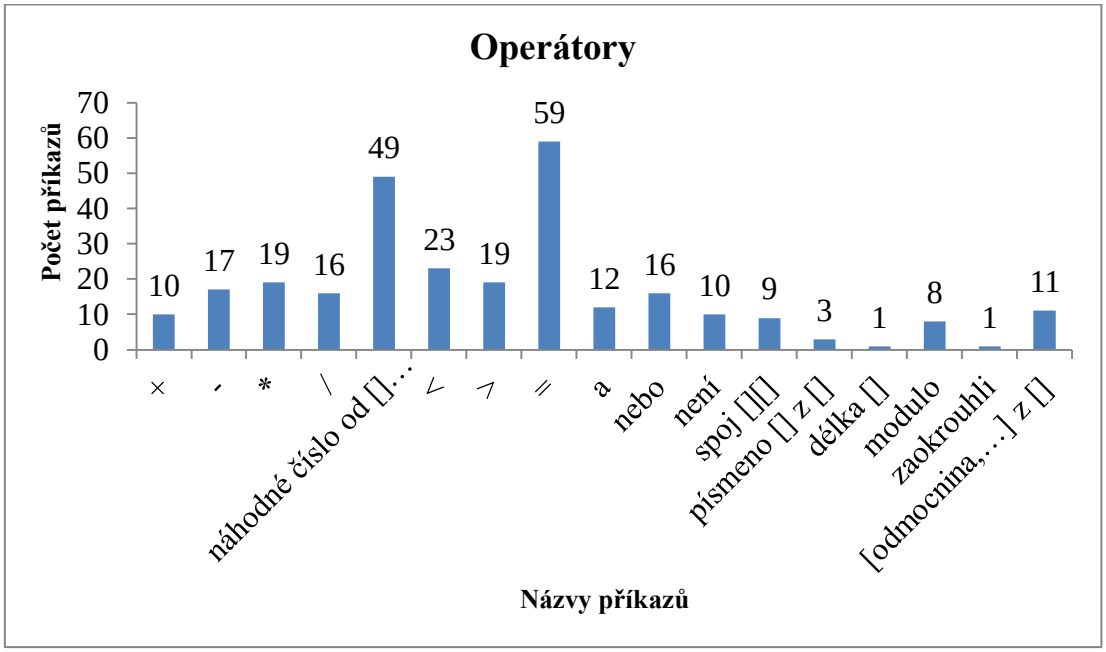
9.4.1 Zpracovaná data

Po shromáždění dat z oblasti *Operátorů* vychází najevo, že se příkazy z této kategorie nejvíce využívají v tematické skupině Hry. Fakticky se zde jeden nebo více příkazů vyskytoval ve 35 projektech z 50. V méně než dvaceti projektech se Operátory objevily ve skupinách Umění (19), Návody (18), Hudba (15) a Animace (12). Poslední místo v rámci tohoto hlediska obsadila skupina Příběhy, v té se operátorové příkazy objevily méně než v 1/5 z celkem 50 projektů.



Graf č. 8: Rozložení Operátorů ve skupinách

Přesuňme se ke konkrétním příkazům z této kategorie. Jako nejvyužívanější příkaz můžeme vidět znaménko *rovná se* (59). Druhý nejpoužívanější příkaz v této kategorii je *náhodné číslo od [] do []* (49). Poté následují další matematické značky *menší než* (23), *větší než* (19), *násobení* (19), *odčítání* (17) a *dělení* (16). Mezi nejpoužívanější logické operátory patří *disjunkce* (16), *konjunkce* (12), a *negace* (10). Pro práci s řetězci byly použité příkazy *spoj [] []* (9), *písmeno [] z []* (3) *a* jedenkrát *délka []*.



Graf č. 9: Využití operátorových příkazů

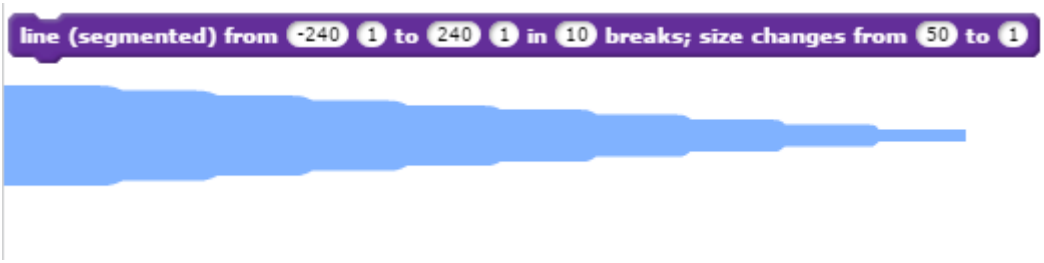
9.4.2 Závěry analýzy

Z provedené statistiky můžeme vyvodit několik závěrů. Jak již bylo řečeno, příkazy z této kategorie se objevily ve více než 1/3 projektů. Ve skupině Hry se příkazů nachází opět nejvíce. Rozmístění dalších skupin stejné ovšem nezůstalo. Skupina s nejmenším podílem jsou Příběhy. Nejpoužívanějším operátorem se stalo *rovná se*. Tento operátor se nejvíce vyskytoval v projektu *Music* od Brady_Lavoie [28] patřící do skupiny *Hudba*. V tomto projektu se objevil celkem 31 krát. V množství o dva menším se rovná se objevilo také v projektu *Binary Tutorial* od jg314159. [29] Značně hojný na různé druhy operátorů je také projekt *Games* od Dillchiplet. [30] Zde bylo použito celkem 9 různých operátorů.

Obr. č. 22: Ukázka z projektu Music

9.5 Vlastní příkazy

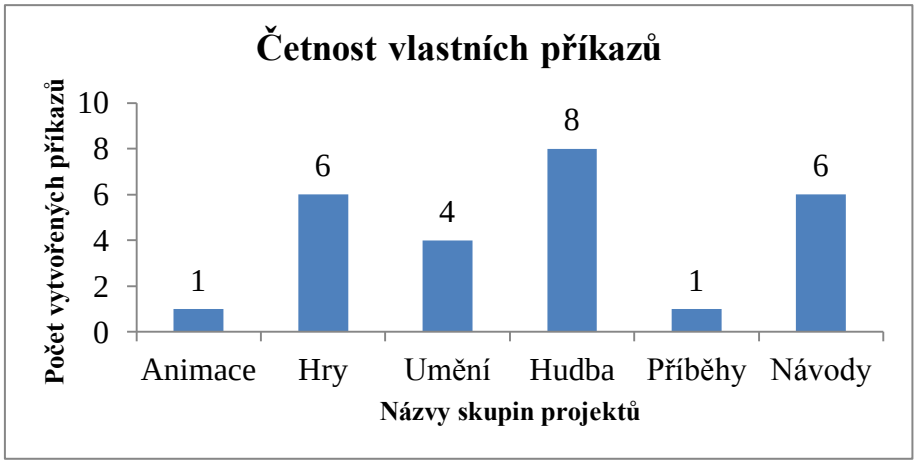
Za nejvzácnější hodnocený prvek můžeme považovat *Vlastní příkazy*. Tvorba vlastních příkazů je značně opomíjenou a nevyužívanou součástí programování ve Scratch. Ve Scratch databázi můžeme narazit na velké množství projektů, ve kterých se skrývají knihovny s již předpřipravenými bloky příkazů. Tyto projekty můžeme srovnat s knihovnami, které shromažďují metody a třídy v jiných jazycích. Uveďme například *Ultimate Custom Block Pack* od ProdigyZeta7. [31] Zde se nachází balíčky příkazů týkající se Pohybu, textových řetězců, Seznamů, Operátorů, Pera (kreslení), Vektorů, Vzhledu, Zvuku, Vnímání a tzv. hacknutých příkazů. Hacknuté příkazy jsou vlastně za normálních okolností nedostupné a nemůžeme si je vybrat z palety příkazů. Jsou samostatně naprogramované, napsané v JSON. Nalezneme zde příkazy pro tvorbu geometrie a mřížek, 2D a 3D efekty, algebraické pojmy, dynamické skoky a také zesílení a zeslabení zvuků.



Obr. č. 23: Ukázka vytvořeného příkazu z Ultimate Custom Block Pack

9.5.1 Zpracovaná data

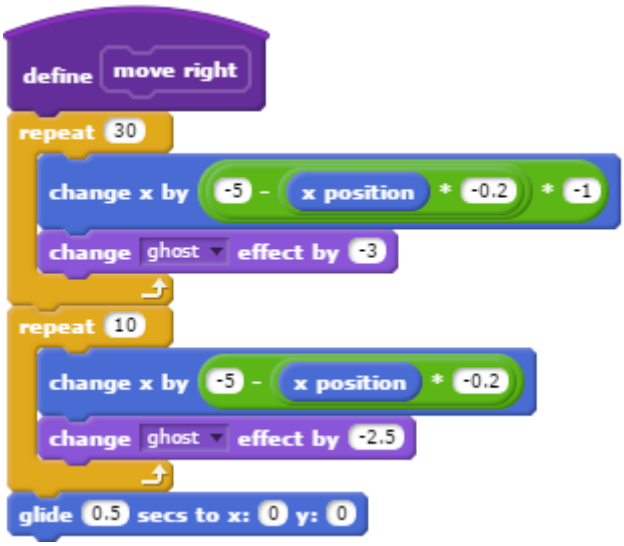
Ve zkoumaných projektech se vlastní příkazy nejvíce používaly ve skupině Hudba. Nejvíce zde znamená, že byly pozorovány celkem v osmi projektech. Ve skupinách Hry a Návody byly zachyceny šestkrát. V Umění byly k vidění ve 4 projektech a shodně po 1 v Animacích a Příbězích. Jedině v této kategorii se na prvním místě neumístila skupina Hry. Avšak pokud bereme v potaz jejich celkovou sumu, na prvním místě je se 14 příkazy skupina Umění, následovaná skupinou Návody se 13.



Graf č. 10: Rozložení Vlastních příkazů ve skupinách

9.5.2 Závěry analýzy

Četnost vlastních příkazů není nijak enormní. Celkem je můžeme pozorovat jen ve 26 projektech napříč všemi skupinami. Jejich celková suma se rovná počtu 46 originálních příkazů. Ne všechny započítané byly v době průzkumu plnohodnotné. Některé byly ve výstavbě a v projektu nevyužité. Bloky příkazů zkoumané ve skupině Umění byly velmi geometricky založené. Uveďme si alespoň jeden příklad příkazu z projektu s názvem *Sliding Tutorial* od aceh18. [32] V tomto projektu nás autor učí, jak vytvořit animaci klouzací postavy. Simulace je zobrazená pomocí textu postupně plynoucího do pravého kraje, ve kterém se poté ztrácí.



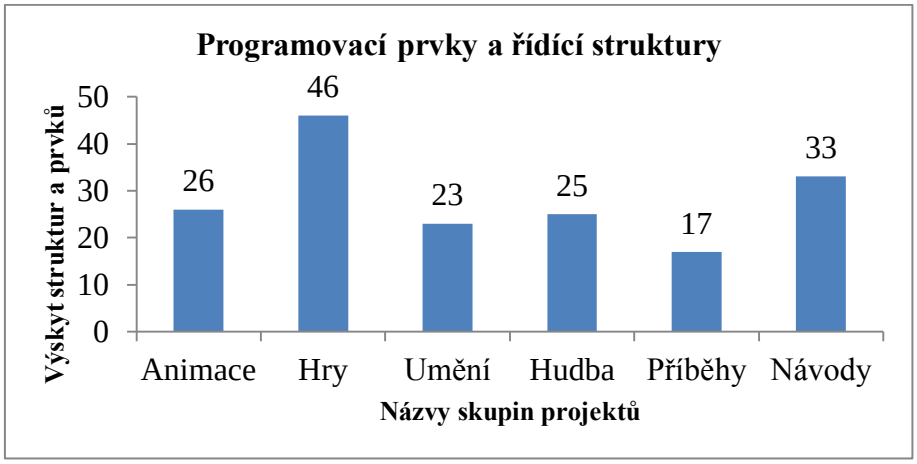
Obr. č. 24: Ukázka z projektu Sliding Tutorial

9.6 Programovací prvky a řídící struktury

Tento prvek sice nefiguruje na prvním místě v hojnosti obsazování některého z jeho příkazů v projektech, ale mnoho jeho příkazů je možné najít ve 20 nejvíce využívaných příkazech.

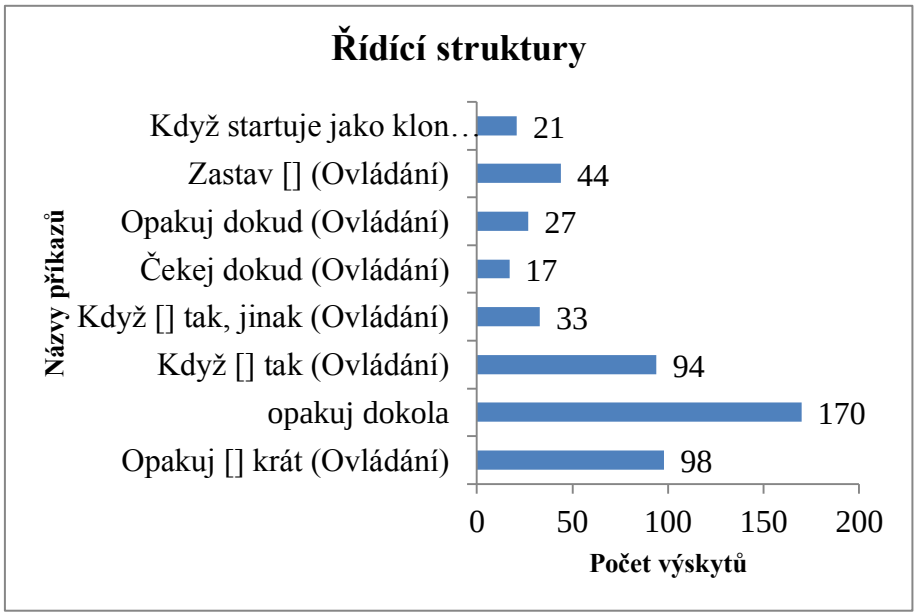
9.6.1 Zpracovaná data

Skupinou, ve kterých byl tento prvek nejvíce detekován, jsou opět Hry. Jen 4 projekty z této skupiny tento prvek neobsahovaly. Dále tento prvek obsahovalo 33 projektů ze skupiny Návody. Téměř ve shodných hodnotách (26, 25, 23) se pohybovaly skupiny Animace, Hudba a Umění. Nejméně *Programovacích prvků a řídících struktur* obsahovaly Příběhy (17).



Graf č. 11: Rozložení prvku ve skupinách

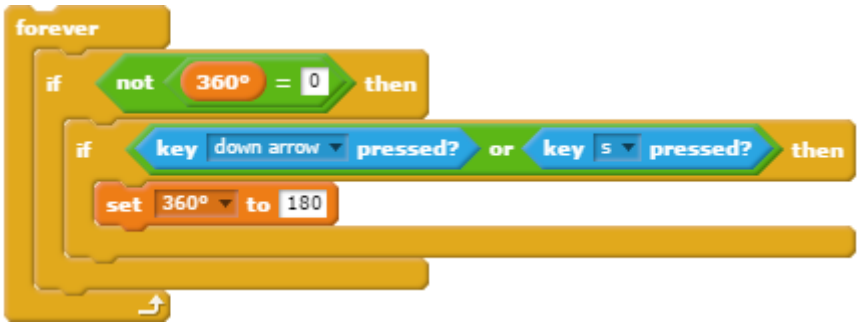
Jak můžeme vidět z grafu č. 10 nejvíce využívaným příkazem v této kategorii se s přehledem stal *Opakuj dokola* (170), následován příkazy *Opakuj [] krá*t (98) a *Když [] tak* (94). *Zastav []* je součástí jen 44 projektů, ukázka větvení v podobě příkazu *Když [] tak, jinak* se objevuje ve 33 z celkem 300. Příkaz *Opakuj dokud []*Následuje příkaz ovládající klonování postav - *Když startuje jako klon* (21). Poslední dva nejméně využívané příkazy jsou tedy *Čekej dokud* (17) a v žádném projektu použitý příkaz *Když hlasitost*.



Graf č. 12: Příkazy řídících struktur

9.6.2 Závěry analýzy

Je velmi potěšující, že mezi prvními můžeme vidět běžné programovací a řídící struktury. Znamená to totiž, že se s nimi děti seznamují již při svém kreativním tvoření a nebudou pro ně znamenat těžkou překážku ve školní výuce programování. Důležité je, když pochopí fungování těchto struktur a dokáží je implementovat. V projektech jsou však často kódy s těmito prvky ve velkém množství kopírovány a činí celkový kód nepřehledným. Jak již bylo řečeno, příkazy z této kategorie patří mezi nejčastěji využívané, neboť skutečně vytvářejí kostru – páteř projektů. Můžeme pozorovat trend, že projekty s nejvíce využitými příkazy z této kategorie patří mezi ty nejsložitější, tedy využívají velké množství ovládacích příkazů. Uveďme si příklad z projektu *Snake 3310* od peroutkavojta. [33]



Obr. č. 25: Ukázka aplikace řídících struktur

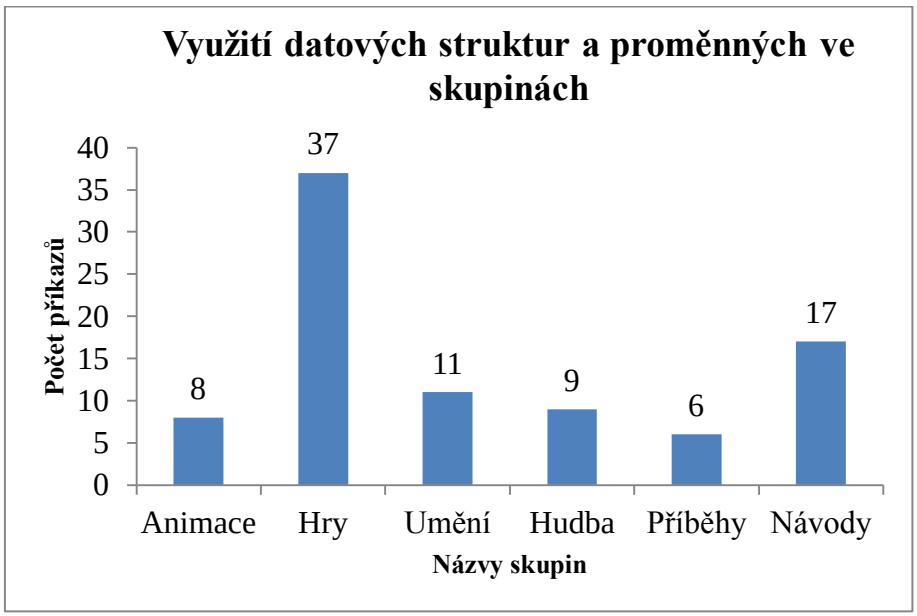
Kód na obrázku je kvůli velikosti zkrácen. Ale je součástí mnohem většího bloku příkazů. K nalezení chyby a jejímu opravení nebo jen k upravení části kódu v této podobě, je z vlastní zkušenosti potřeba silné vůle a znamená ztrátu času, výhodnější je segmentace kódu pomocí vlastních příkazů.

9.7 Proměnné a datové struktury

Tento prvek je v této práci sice popisován na posledním místě, ovšem to neznamená, že je málo důležitý. Naopak, implementace datových struktur a proměnných byla po vlastních příkazech nejzajímavější částí průzkumu. V prvé řadě šlo hlavně o to zjistit, jak běžné je jejich použití a částečně zmapovat, k čemu a v jaké podobě jsou využívány.

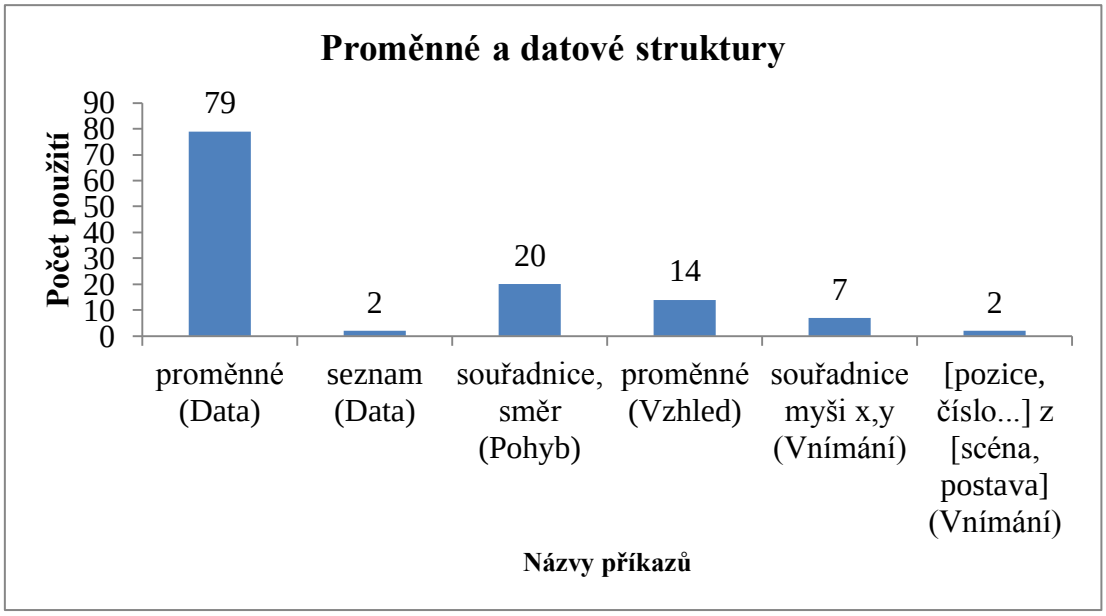
9.7.1 Zpracovaná data

Jako téměř každý prvek je i tento nejvíce obsažen ve skupině Hry. Můžeme ho naleznout ve 37 projektech. U ostatní skupin již není tak běžně využívaný. V Návodech jsem na něj narazil v 17 projektech, což je jen o trochu více než 1/3. V téměř stejném množství je možné najít jeden nebo více příkazů týkajících se této kategorie ve skupinách Umění (11), Hudba (9) a Animace (8). Nejmenší počet je nakonec ve skupině Příběhy (6).



Graf č. 13: Rozložení datových struktur ve skupinách

Nejvíce využívaný způsob je vytvoření vlastní proměnné (79). Následovaly proměnné ze skupiny Pohyb, tedy *souřadnice x, y* nebo *směr* natočení postavy. Dále se využívaly proměnné z kategorie příkazů Vzhled, *číslo kostýmu* a *pozadí*, nebo *velikost postavy*. Můžeme na ně narazit celkem ve 14 zkoumaných projektech. Dalšími proměnnými jsou *souřadnice myši x, y* patřící ke skupině Vnímání. Ty byly použity v 7 projektech. Nejmenší podíl v tomto prvku drží proměnná *[pozice, číslo…] z [scéna, postava]* (2) a implementace seznamů (2).



Graf č. 14: Proměnné a datové struktury

9.7.2 Závěry analýzy

Malá četnost těchto prvků není překvapující. Slabé využití uvádějí oficiální statistiky i tato práce. Překvapením je, že téměř v každém projektu, ve kterém byl detekován tento prvek, se nacházela vlastní proměnná. Malá je též četnost seznamů.

Obr. č. 26: Ukázka použití proměnných

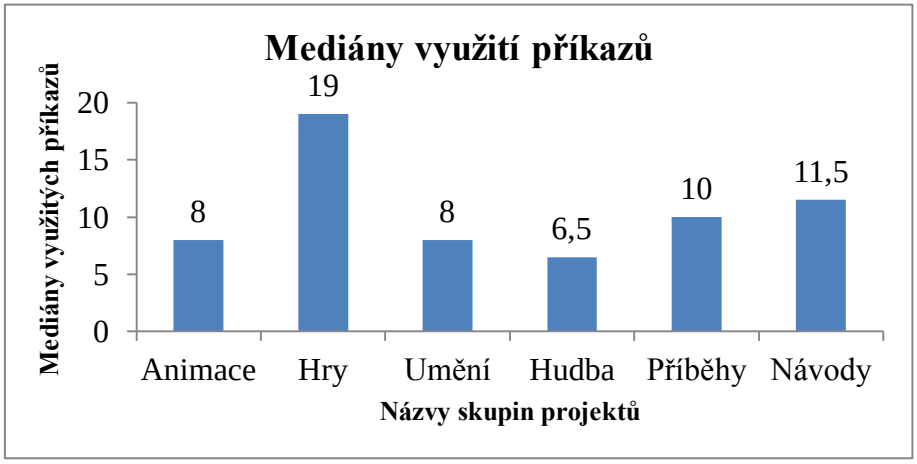
Uveďme jeden příklad, ve kterém se místo elegantnějšího použití seznamu, použilo velké množství proměnných. Projekt se nazývá *fishy stories!* od GW404. [34] Jde v něm o to, že postava vypráví příběh a doplňuje ho o vaše odpovědi, které ukládá do celkem 9 proměnných.

9.8 Dílčí výsledky

Díky velkému množství akumulovaných dat bylo možné vytvořit více pohledů na danou problematiku. V kapitole *Dílčí výsledky* se můžeme například dozvědět, jaká je střední hodnota využitých příkazů ve skupině projektů. Poslední grafické zpracování dat se týká mediánů celkových sum příkazů ve skupinách.

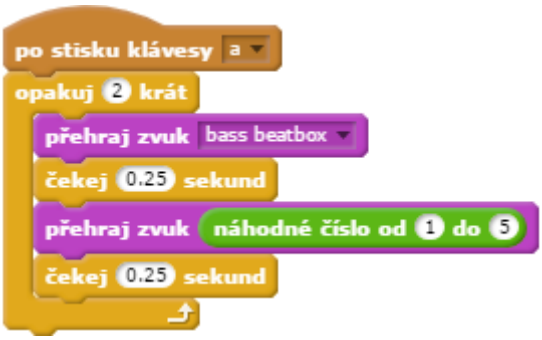
9.8.1 Nevyužité příkazy

Nejdříve je nutné vysvětlit, co vlastně znamená tento graf. Každému projektu bylo spočítáno, kolik se v něm vyskytuje příkazů. Poté se ze všech projektů patřících do skupiny spočítala střední hodnota těchto výskytů. Můžeme vidět, že nejvíce druhů příkazů na jeden projekt bylo zjištěno ve skupině Hry (19). Ostatní skupiny měly víceméně podobné hodnoty, Návody (11,5), Příběhy (10), Animace shodně s Uměním (8) a nejméně na příkazy různorodá skupina Hudba.



Graf č. 15: Mediány využití příkazů ve skupinách

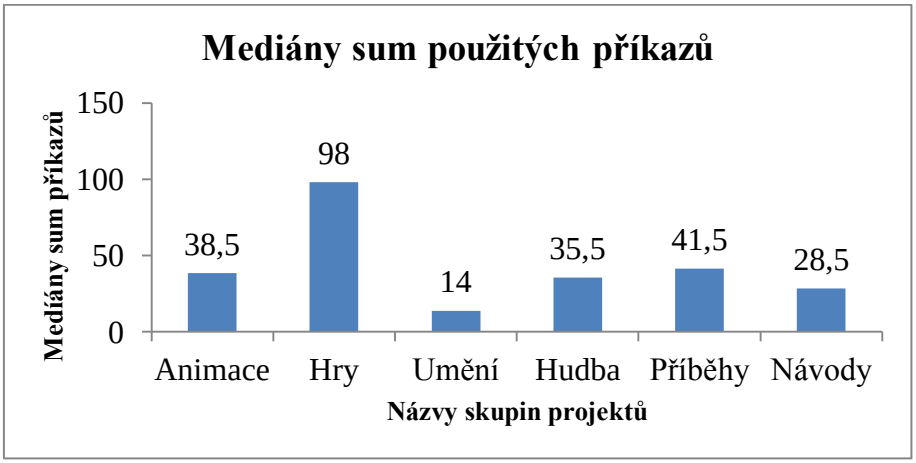
Data nám ukazují, že šesti nejvíce využívanými příkazy jsou (číslo v závorce znamená, v kolika projektech byl příkaz nalezen): Přehraj zvuk (35), Po kliknutí na vlajku (35), Čekej [] sekund (29), Opakuj dokola (21), Opakuj [] krát (19), Po stisku klávesy [] (18). Velmi běžným příkladem v hudebních projektech je například tento blok příkazů z projektu *Music* od young1534321. [35]



Obr. č. 27: Ukázka z projektu Music

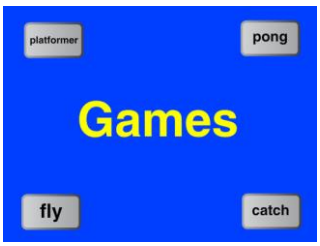
9.8.2 Mediány sum použitých příkazů v rámci skupin

Výraznou diferenciaci skupin můžeme vidět v tomto grafu. Každému projektu byla spočítána suma použitých příkazů. Z těchto sum byl poté vypočítán medián. A tyto mediány nám jasně ukazují, jak obsáhlé projekty ve skupinách jsou. Nejvíce příkazů bylo dokumentováno v projektech ze skupiny Hry. Střední hodnota se rovná 98. Ostatní skupiny jsou již na méně než dvojnásobně menších hodnotách. Nejhůře dopadly projekty ze skupiny Umění. Zde se medián pohybuje okolo 14 příkazů na projekt.



Graf č. 6: Mediány sum použitých příkazů

Jeden z nejobsáhlejších projektů byl nalezen ve skupině Hry, obsahoval dohromady 1559 příkazů [36] a druhý nejpočetnější ze stejné skupiny jich obsahoval 1004. [37] Tyto projekty byly složeny z více her, které jste si mohli vybrat v nabídce a hrát.



Obr. č. 28: Snímek scény z projektu Games [38]

10 Závěr

Cílem práce bylo odpovědět na otázku *Co vlastně děti programují ve Scratch?* Nejjednodušší odpovědí je, že zde děti vytvářejí různě obtížné i jednoduché druhy projektů od galerií, her a animací po simulace hudebních přehrávačů a návodů na různé aktivity. A spíše než cílené programování děti k tvorbě táhne zábavnost a přitažlivé produkční prostředí s pestrou paletou nástrojů a příkazů.

Víme, že se všechny informace o projektu zapisují do formátu JSON. Známe všechny klíče a hodnoty, kterých mohou nabývat. Poznali jsme i strukturu zápisu těchto informací v javascriptovém objektu.

Zjistili jsme, jaké nástroje se používají k vyhledávání a analýze projektů. Program Hairball je užitečný pro analýzu projektů, dokáže nám mimo jiné říct, jaké příkazy se v projektu nacházejí.Vyhledávat můžeme pomocí Scratch Explorer nebo Scratch API, každý způsob má svá specifika.

Celkem jsme analyzovali 21840 příkazů. Nejpoužívanější příkazy při programování ve Scratch jsou z kategorií Ovládání, Události a Vzhled. Nejméně používané příkazy se nachází v kategoriích Operátory a Vnímání.

Dále známe nejrozšířenější tematické celky, do kterých se dají projekty zařadit. Ty nejobsáhlejší projekty mají většinou téma Hry, dále se setkáme s Animacemi, Návody, Hudebními projekty a Příběhy.

Je vytvořeno hodnocení zkoumaných projektů podle prvků, které se v nich objevují. Hledané prvky jsou Interaktivita, Proměnné a datové struktury, Programové prvky a řídící struktury, Vlastní příkazy a Operátory. Nejčastěji jsme se setkali s Interaktivními prvky a prvky Řídících struktur. Dále můžeme tvrdit, že víme, jaké prvky se k sobě vážou. Pokud se v projektu nacházelo více prvků, většinou to byly Datové struktury, Operátory, Interaktivita a Programové prvky a řídící struktury.

Po vypracování této práce zastávám názor, že Scratch není jen o výuce programování. Scratch podporuje dětskou kreativitu a samostatnost při řešení problémů. Lze ho využít pro simulaci jevů z různých vědních oborů i pro prezentaci výuky dalších předmětů. Další výzkum by se mohl týkat přímo školních studií a v nich uložených projektů, jejich témat a výukových metod. Posledním námětem pro další práce je analýza a srovnání rozšiřitelných doplňků Scratch.

Vzhledem k možnostem rozsahu bakalářské práce jsme se nevěnovali dalším vlastnostem projektů jako popularita, oblíbenost, remixování a jiné. Také nedošlo k průzkumu sociální interakce mezi tvůrci projektů ať už v komentářích pod projekty nebo v diskuzních fórech a ke sledování grafické a zvukové tvorby tak, abychom mohli výsledky dále interpretovat. Tato témata mohou být námětem pro další podobné práce.

11 Seznam použité literatury

[1] Scratch Program. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Scratch_Program

[2] LAMB, Annette. "Scratch: Computer Programming for 21st Century Learners". *Teacher Librarian*. 2011, **38**(4), 64-68.

[3] Project. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Project

[4] Scratch User Interface. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Scratch_User_Interface

[5] Blocks. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Blocks

[6] Scratch. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Scratch

[7] Studio. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Studio

[8] MORENO-LEÓN, Jesús, Gregorio ROBLES a Marcos ROMÁN-GONZÁLEZ. *Dr. Scratch: Automatic Analysis of Scratch Projects to Assess and Foster Computational Thinking* [online]. Programamos.es. Spain., 2015 [cit. 2017-07-05]. Dostupné z: https://www.researchgate.net/publication/281714025_Dr_Scratch_Automatic_Analysis_of_Scratch_Projects_to_Assess_and_Foster_Computational_Thinking

[9] JSON. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/JSON

[10] Scratch File Format (2.0). *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Scratch_File_Format_(2.0)

[11] Custom Blocks. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Custom_Blocks

[12] BOE, Bryce, Charlotte HILL, Michelle LEN, Greg DRESCHLER, Phillip CONRAD a Diana FRANKLIN. Hairball: Lint-inspired Static Analysis of Scratch Projects. In: *Proceedings of the 44th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2013)*. Denver, CO, 2013.

[13] Hairball. *GitHub* [online]. [cit. 2017-07-04]. Dostupné z: https://github.com/ucsb-cs-education/hairball

[14] Scratch API (2.0). *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/API

[15] CARINI, Gaia. *Finding a Needle in a Haystack: New Ways to Search and Browse on Scratch*. Department of Electrical Engineering and Computer Science, 2012. Mitchel Resnick, LEGO Papert Professor of Learning Research. MIT.

[16] Explore. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Explore

[17] Animation Projects. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Animation_Projects

[18] Games. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Game_Projects

[19] Art Projects. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Art_Projects

[20] Music Projects. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Music_Projects

[21] Story Projects. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Story_Projects

[22] Tutorial Projects. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/Tutorial_Projects

[23] Statistics: Scratch Block Usage (random sample). *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/statistics/

[24] DAVID_SH. Samus Vector Demo and Bosses (ZeroFusion). In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/154413874/

[25] ELEANOR_2003. Stories. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/148484482/

[26] REDDITGAMES. GAMES Version: 1.2. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/155121784/

[27] FRIEFREE. Carbon Mechanics. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/150143922/

[28] BRADY_LAVOIE. Music. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/154947928/

[29] JG314159. Binary Tutorial. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/154947928/

[30] DILLCHIPLET. Games. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/154047407/

[31] PRODIGYZETA7. Ultimate Custom Block Pack. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/12676518/

[32] ACEH18. Sliding Tutorial. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/154356216/

[33] PEROUTKAVOJTA. Snake 3310. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/151296328/

[34] GW404. Fishy stories!. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/155432719/

[35] YOUNG1534321. Music. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/155198314/

[36] DAMEKTM. Games. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/153051344/

[37] POKOLOP. Žebrák 4. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/150672990/

[38] THUNDERMASTER1. Games. In: *Scratch* [online]. [cit. 2017-07-04]. Dostupné z: https://scratch.mit.edu/projects/153626817/

[39] Scratch 3.0. *Scratch Wiki* [online]. [cit. 2017-07-04]. Dostupné z: https://wiki.scratch.mit.edu/wiki/3.0

Přílohy

Příloha č. 1: CD – ROM

- Bakalářská práce ve formátu PDF