Univerzita Hradec Králové

Pedagogická fakulta

Katedra kybernetiky

POROVNÁNÍ VYBRANÝCH VÝVOJOVÝCH PROSTŘEDÍ PRO OPERAČNÍ SYSTÉM ANDROID SE ZAMĚŘENÍM NA VÝUKU NA STŘEDNÍCH ŠKOLÁCH

Diplomová práce

Autor:	Bc. Martin Jodas
Studijní program:	N7504 / Učitelství pro střední školy
Studijní obor:	7504T261 / Učitelství pro střední školy - informatika
	7504T / Učitelství pro střední školy - společný základ
	7503T163 / Učitelství pro 2. stupeň ZŠ - německý jazyk a literatura
Vedoucí práce:	Ing. Petr Voborník, Ph.D..
Oponent:	Mgr. et Bc. Radek Němec, Ph.D.

Hradec Králové

2018

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval pod vedením vedoucího diplomové práce samostatně a uvedl jsem všechny použité prameny a literaturu.

V Hradci Králové, dne

Anotace

JODAS, Martin. *Porovnání vybraných vývojových prostředí pro operační systém Android se zaměřením na výuku na středních školách.* Hradec Králové: Pedagogická fakulta Univerzity Hradec Králové, 2018. 96 s. Diplomová práce.

Diplomová práce se v první části věnuje analýze současného stavu výuky programování v českém školství. V části druhé jsou dle předem stanovených kritérií zkoumána a porovnávána tři vývojová prostředí (MIT App Inventor 2, Android Studio a Visual Studio s nástrojem Xamarin) pro operační systém Android. V části třetí jsou prezentovány výsledky vlastního dotazníkového šetření, které zkoumalo postoj českých pedagogů k výuce programování a potažmo programování, jehož výstupem je mobilní aplikace. V části čtvrté jsou popsány zkušenosti nabyté při psaní testovací aplikace ve všech třech vývojových prostředích a je také přiložen pracovní list pro MIT App Inventor 2, který může být využit při výuce. Zdrojové kódy pro jednotlivá vývojová prostředí jsou v přílohách.

Klíčová slova: výuka programování, MIT App Inventor 2, Android Studio, Visual Studio, Xamarin

Annotation

JODAS, Martin. *A comparison of selected integrated development environments for the Android operating system with a focus on secondary education.* Faculty of Education, University of Hradec Králové, 2018. 96 pp. Diploma Thesis.

In the first part, the diploma thesis analyzes contemporary situation of teaching programming in Czech education system. In the second part, three integrated development environments are explored and compared according to given criteria (MIT App Inventor 2, Android Studio and Visual Studio with Xamarin tool) for Android operating system. In the third part, there are presented results of own questionnaire research which analysed the attitude of Czech teachers to programming, resp. programming whose result is a mobile application. In the fourth part, the experiences acquired by programming a testing application in all three integrated development environments are described and moreover, a working list for Mit App Inventor 2 is enclosed, which can be used in teaching. Source codes for individual integrated development environments can be found in attachments.

Key words: teaching of programming, MIT App Inventor 2, Android Studio, Visual Studio, Xamarin

Prohlášení

Prohlašuji, že diplomová práce je uložena v souladu s rektorským výnosem č. 1/2013 (Řád pro nakládání se školními a některými jinými autorskými díky na UHK).

Datum:……. Podpis studenta:……

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu diplomové práce Ing. Petru Voborníkovi, Ph.D. za jeho cenné rady, bezměrnou trpělivost a ochotu.

Díky za podporu patří i mé skvělé snoubence a rodině.

Obsah

Přehled použitých zkratek

HTML - Hypertext Markup Language

HW - Hardware

ICT - Information and Communication Technologies

IDE - Integrated Development Environment

IS – Informační systém

IT – Informační technologie

MIT - Massachusetts Institute of Technology

OS – Operační systém

PC – Personal computer

RVP – Rámcový vzdělávací program

SŠ – Střední škola

ŠVP – Školský vzdělávací program

UHK – Univerzita Hradec Králové

UI – User Interface

ZŠ – základní škola

Úvod

V současné době je diskutována možná změna RVP[1] v oblasti ICT, která by měla přinést zakotvení výuky algoritmizace, potažmo programování do ŠVP[2] již od prvního stupně základní školy. Vzhledem k tomu, že v páté třídě vlastní mobilní telefon téměř každé dítě [1], bylo by skvělé využít tuto skutečnost právě pro již zmíněnou výuku programování či algoritmizace.

Tato diplomová práce analyzuje současný stav výuky programování na základě RVP a na základě vlastního dotazníkového šetření.

V této diplomové práci je nabízeno srovnání tří vývojových prostředí pro operační systém Android, který je dle společnosti Rankings.cz[3] v České republice s 82,75 % tržním podílem nejrozšířenějším operačním systémem pro mobilní telefony [5]. Srovnávání jednotlivých IDE je realizováno v různých kategoriích a je tak možné přizpůsobit výběr IDE v závislosti na podmínkách, ve kterých se toto IDE bude používat. Ke každému IDE je přiložena ukázková aplikace včetně zdrojových kódů, aby rozhodnutí, které vývojové prostředí využít, bylo jednodušší.

[1] RVP *„tvoří obecně závazný rámec pro tvorbu školních vzdělávacích programů škol všech oborů vzdělání v předškolním, základním, základním uměleckém, jazykovém a středním vzdělávání“* [1].
[2] ŠVP je tvořen učiteli každé školy, závazný dokument pro jeho tvorbu je výše uvedené RVP [2].
[3] Společnost Ranking.cz *„[…] odhad využívanosti operačních systémů počítá z 1,7 miliardy zobrazených stránek na největších webech Česka.“* https://www.cnews.cz/je-tady-windows-10-je-nejpouzivanejsim-systemem-v-cesku/

1 Výuka programování

První část této diplomové práce je věnována analýze současného stavu školství v oblasti výuky programování. Následně bude ukázáno, proč je důležité věnovat pozornost výuce programování na základních i středních školách a proč je dobré se zaměřit i na výuku programování pro mobilní operační systémy (OS).

1.1 Současný stav výuky programování

Současný stav výuky bude posouzen ze dvou hledisek – z dostupných dokumentů RVP, které následně určují ŠVP a z osobní zkušenosti.

RVP jsou psány pro předškolní vzdělávání, základní vzdělávání, gymnázia – (gymnázia, gymnázia se sportovním zaměřením, gymnázia v angličtině, dvojjazyčná gymnázia) a pro střední odborné vzdělávání (obory J, E, H, L0 a M, konzervatoře a nástavbové studium), speciální vzdělávání a základní umělecké vzdělávání [6]. V těchto dokumentech je hlavní pozornost věnována části *„Informační a komunikační technologie“*, která obsahuje cíle, které by měl žák po dokončení příslušné školy ovládat. Tyto cíle jsou ještě následně rozpracovávány do několika vzdělávacích oblastí, které konkrétněji definují požadované výstupy.

1.1.1 Situace na základních školách

Mezi cíli chybí jakákoliv zmínka o programování, nejblíže se programování blíží požadavek na užívání algoritmického myšlení při interakci s počítačem (PC) [7, s.38]

1.1.2 Shrnutí situace na gymnáziích

V cílech pro gymnázia je uvedeno „*Uplatňování algoritmického způsobu myšlení při řešení problémových úloh“* [8, s. 63]. Nicméně i tato formulace programování přímo nezmiňuje. Zajímavěji zní definice v jedné části RVP pro gymnázia *„Zpracování a prezentace informací“*, ve které je jedním z požadovaných cílů i *„algoritmizace úloh – algoritmus, zápis algoritmu, úvod do programování“* (8, s. 65). Nutno podotknout, že stejné cíle i rozpracování platí nejen pro běžná gymnázia ale i pro další tři typy gymnázií, které jsou uvedeny v prvním odstavci kapitoly 1.1.

1.1.3 Shrnutí situace ve středním odborném vzdělávání

Ve většině RVP pro obory jmenované v prvním odstavci kapitoly 1.1. je zakotvena tato věta: *„ovládá principy algoritmizace úloh a sestavuje algoritmy řešení*

konkrétních úloh (dekompozice úlohy na jednotlivé elementárnější činnosti za použití přiměřené míry abstrakce)" [9, s.37] – citace pochází z RVP pro obor autolakýrník, jen pro představu, stejnou citaci ale nalezneme i v RVP pro obor hudebně dramatické umění [10, s. 29]. Jiná je ale situace u oborů, které jsou orientovány na informační technologie, jako je například obor *„Informační technologie"*, v jehož RVP je výuka programování přímo zanesena [11, s. 52–53]. Oproti tomu ale zmínka i jen o algoritmizaci chybí například v oborech jako *„Ladění klavíru a kulturní činnost"* [12] a *„Pečovatelské služby"* [13] a dalších.

1.1.4 Osobní zkušenost ze vzdělávacího systému

Ve vzdělávacím systému jsem působil nejen jako studující, ale také jako praktikant na jedné základní škole v Hradci Králové. Na této škole jsem také jeden roku vyučoval informatiku. S programováním jsem se setkal na gymnáziu v rámci speciálního semináře, ve kterém byl vyučován programovací jazyk Pascal, v rámci běžné výuky jsme se učili tvořit webové stránky v HTML, nicméně HTML je značkovací jazyk [13] a proto není k programovacím jazykům řazen.[4] Během studia na UHK jsem absolvoval 5 předmětů, ve kterých bylo probíráno programování v jazyce C#, příkazy v asembleru a dětské programovací jazyky, například Karel [16], Logo [17] či SCRATCH [18]. Jako učitel jsem se snažil programování do hodin informatiky vnést, během mého působení jsme s žáky pracovali ve SCRATCHi, Karlovi či například na webových stránkách CodeCombat.com [19], na kterých si žáci zábavnou formou mohou osvojit základy jazyka PYTHON, JavaScript, Lua či CoffeeScript. Pokud se hodina dobře didakticky připraví, žáky programování baví.

1.2 Inovativní přínos vlády České republiky

Digitální technologie stále více a více prostupují a budou prostupovat naše životy. Aby byly příští generace na tuto skutečnost adaptovány, přijala vláda České republiky 12.11.2014 dokument s názvem „Strategie digitálního vzdělávání". Mezi prioritní cíle tohoto programu patří:

[4] *„Programovací jazyky jsou jazyky sloužící k tvorbě počítačových programů (programování). Programování je proces algoritmizace dané úlohy, tj. vytváření postupu, jenž vede k řešení dané úlohy."* [14], zatímco jazyky značkovací jsou *„prostředek k obohacení textu o dodatečné informace – nejčastěji o významu, struktuře a způsobu zobrazování jednotlivých částí textu"* [15].

- *„otevřít vzdělávání novým metodám a způsobům učení prostřednictvím digitálních technologií,*
- *zlepšit kompetence žáků v oblasti práce s informacemi a digitálními technologiemi,*
- *rozvíjet informatické myšlení žáků“* [20, s. 16]

Tyto cíle mají být naplněny těmito sedmi intervencemi:

1) *„Zajistit nediskriminační přístup k digitálním vzdělávacím zdrojům.*
2) *Zajistit podmínky pro rozvoj digitální gramotnosti a informatického myšlení žáků.*
3) *Zajistit podmínky pro rozvoj digitální gramotnosti a informatického myšlení učitelů.*
4) *Zajistit budování a obnovu vzdělávací infrastruktury.*
5) *Podpořit inovační postupy, sledování, hodnocení a šíření jejich výsledků.*
6) *Zajistit systém podporující rozvoj škol v oblasti integrace digitálních technologií do výuky a do života školy.*
7) *Zvýšit porozumění veřejnosti cílům a procesům integrace technologií do vzdělávání.“* [20, s. 18]

Jak se daří tyto intervence plnit, lze zkontrolovat ve „Zprávě o implementaci Strategie digitálního vzdělávání“, která uvádí nové návrhy a úpravy RVP [21, s. 12]. Dne 8.2.2018 proběhla konference na téma „Informatika, roboti a informatické myšlení ve škole? Představení vize nového obsahu výuky informatiky a informatického myšlení od MŠ po SŠ“, na kterém účastníci připomínkovali návrh aktualizace RVP v oblasti ICT [22]. Dle tohoto návrhu se má již u dětí na prvním stupni základních škol rozvíjet algoritmické myšlení a žák na druhém stupni základní školy umí sestavit program v blokově orientovaném programovacím jazyce, studenti na středních školách v závislosti na jejich oboru a typu školy tuto dovednost ještě dále rozvádějí [23].

1.2 Hledisko poptávky trhu

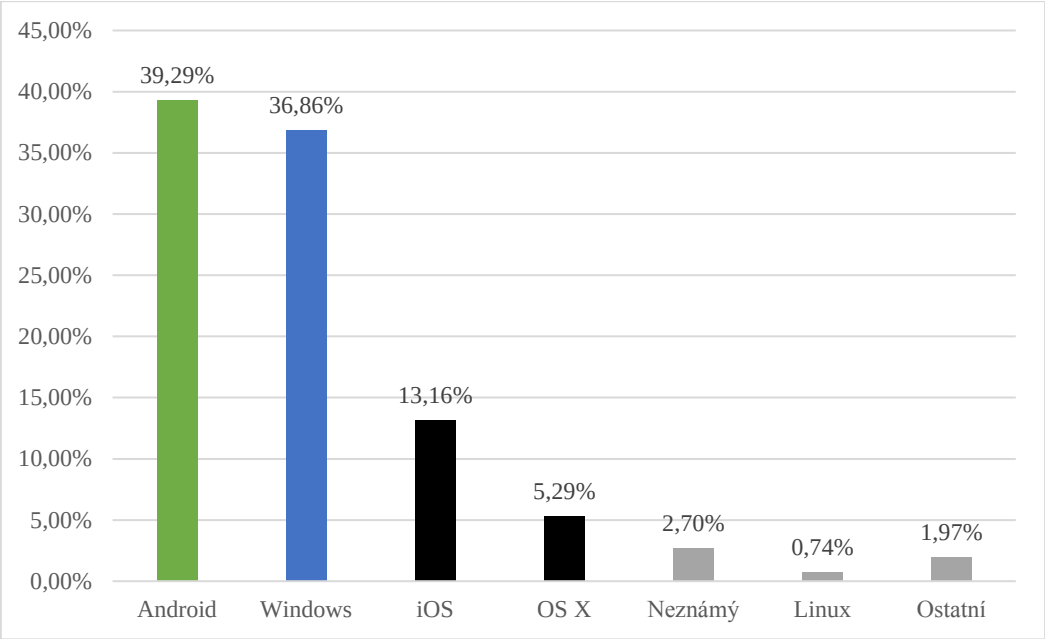
Z tohoto hlediska z určité části vychází i stanovisko vlády České republiky, která podporuje zařazení výuky programování a algoritmizace do osnov pro školy (viz kapitola 1.1), nicméně v této části bude toto stanovisko ještě více rozpracováno.

Předpoklad, že počet programátorů na trhu je nedostatkový, byl potvrzen po zadání dotazu na serveru Jobs.cz[5][24] na poptávku práce *„Programátor + IS/IT Vývoj aplikací a systémů“*, když se objevilo zhruba 2400 nabídek, z nichž zhruba 1000 bylo přidáno v posledním týdnu. Tento druh práce je také velmi dobře placený, neboť celá polovina inzerátů (1233) nabízela plat začínající od 50 000 Kč, 219 plat od 100 000 Kč.

1.3. Význam výuky programování pro mobilní OS

Význam chytrých mobilních telefonů neustále roste. Dokladem tohoto tvrzení je například statistika společnosti StatCounter, která uvádí, že stránky, na kterých je instalován jejich kód (asi 2 miliony stránek s asi 10 miliardami shlédnutími za měsíc) [25], prohlíží 51,78 % uživatelů přes mobilní telefony, 43,65 % z počítače a 4,57 % přes tablet [26]. Mezi nejrozšířenější operační systémy patří OS Android s 39,29 % podílem, dále Windows s 36,86 %, iOS s 13,16 % a OS X s 5,29 % (viz Graf 1).

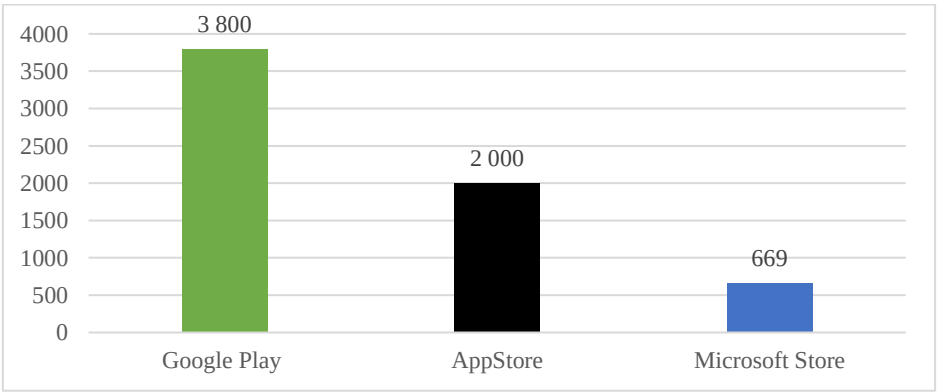
[5] Server Jobs.cz se věnuje zprostředkování práce.



Graf 1 - Procentuální rozšíření operačních systémů za období únor 2017 - únor 2018 [27]

Za rok 2017 se celosvětově dle analytické společnosti Gartner prodalo 262,5 milionu počítačů [28], zatímco za stejné období se prodalo 1,5 miliardy chytrých telefonů [29].

Dle údajů stránky Statista.com bylo za první kvartál roku 2018 k dispozici pro obchod Google Play 3 800 000 aplikací, pro konkurenční Apple AppStore 2 000 000 a pro Microsoft Store 669 000 aplikací [30] (viz Graf 2).



Graf 2 - Počty aplikací uváděné v tisících v obchodech Google Play, Apple App Store a Microsoft Store

2 Srovnání tří vývojových prostředí pro platformu Android

V této části práce jsou srovnávána tři vybraná vývojová prostředí, konkrétně App Inventor 2, Android Studio a Visual Studio s nástrojem Xamarin. Pro srovnání byla vybrána tato IDE kvůli jejich rozdílným přístupům k programování aplikací.

MIT App Inventor je prostředí založené na vizuálním programování, Android Studio je oficiálním IDE pro OS Android a Visual Studio s doplňkem Xamarin umožňuje psát aplikace v jednom programovacím jazyce a publikovat je na různých platformách. Aby bylo porovnání autentičtější, byla ve všech třech prostředích naprogramována vzorová aplikace. Zdrojové kódy jsou k dispozici v přílohách. IDE byla testována na notebooku Lenovo E520 s operačním systémem Windows 7, dvoujádrovým procesorem Intel Pentium B960 a s 4 GB paměti RAM.

Ke srovnání IDE bude docházet obzvláště dle předem stanovených kritérií, kterými jsou: Čas strávený instalací, nastavováním a následnými aktualizacemi, srozumitelnost vývojového prostředí pro žáky, programovací jazyk, v kterém lze v IDE programovat a podpora k IDE a programovacímu jazyku. Důvody, proč byla zvolena tato kritéria, jsou uvedena v následujícím výčtu.

Čas strávený instalací, nastavováním a následnými aktualizacemi – tento faktor je důležitý pro učitele, který se rozhoduje, jaké IDE bude ve výuce používat. Po instalaci IDE je nutné věnovat pozornost i občasné aktualizaci tohoto prostředí, případně instalaci různých doplňků, balíčků a knihoven. A jestliže učitel nemá oprávnění instalace na školních PC, může vzniknout velká prodleva mezitím, než pověřený pracovník s administrátorským heslem vše nastaví.

Srozumitelnost vývojového prostředí pro žáky – v tomto kritériu je zahrnuto například „počeštění“ IDE, zda jsou přeloženy nabídky, menu, případně chybová hlášení. Čeština může vést k větší srozumitelnosti, nicméně není pro toto kritérium tolik zásadní, neboť žáci se v programování většinou setkají s nutností používat anglický jazyk. Dále je hodnocena celková přehlednost prostředí, ve zkratce, zda se v tomto prostředí dobře orientuje.

Programovací jazyk, který se používá pro kódování v IDE – Toto kritérium popisuje užitý programovací jazyk a jeho perspektivitu pro výuku, potažmo pro budoucnost absolventa. Pro pokročilejší výuku programování je vhodné zvolit nějaký více

frekventovaný jazyk, který může poté absolvent využít v praxi. Naopak pro základní výuku programování může být vhodné vyučovat v jazyku, který není tak rozšířený, ale naopak je uzpůsoben pro výuku.

Podpora k IDE a programovacímu jazyku – toto kritérium je důležité nejen pro žáka, ale i pro vyučujícího. Díky širší podpoře pro určité vývojové prostředí je možné se lépe samo vzdělávat a zvyšovat pravděpodobnost vyřešení nějakého problému, který při vývoji aplikací často nastává. Pod pojmem „podpora“ jsou myšleny dokumentace, podpora na komunitních webech, případně učebnice. Pokud jsou tyto zdroje v češtině, jedná se jistě o pozitivum.

2.1 MIT App Inventor 2

Jak uvádí oficiální stránky MIT App Inventoru, začala se psát historie tohoto projektu v roce 2010 [31]. Přičemž do 15.7.2015 bylo možné vyvíjet v Mit App Inventor Classic, poté bylo toto vývojové prostředí vypnuto a nahrazeno prostředím App Inventor 2, které je zkoumáno v této diplomové práci. Dle slov vedoucího projektu App Inventoru Marka Friedmana bylo hlavní inspirací pro vznik této aplikace přání využít motivační síly mobilních telefonů pro vzdělávání, pro přiblížení počítačové vědy běžným uživatelům a dále také přeměnit pasivní uživatele telefonů a aplikací na tvůrčí osobnosti, kteří rozumí tomu, jak tato technika funguje a kteří dokáží realizovat svoje nápady na aplikace i bez nutnosti najmutí programátora [32, s.15]. Díky jednoduchému vývojovému prostředí lze naprogramovat první menší aplikaci do několika desítek minut, neboť toto vývojové prostředí je řešeno vizuálně. Uživatel si totiž nejprve v App Inventoru navrhne uživatelské prostředí aplikace a poté pomocí bloků definuje, jak se bude aplikace chovat. Níže je uveden popis App Inventoru dle kritérií stanovených v kapitole 2.

2.1.1 Čas strávený instalací, nastavováním a následnými aktualizacemi s prostředím MIT App Inventor

Vyvíjení aplikací pro Android probíhá v internetovém prohlížeči, jejich podpora je zobrazena v následující tabulce:

Tabulka 1 - Požadavky App Inventoru na prohlížeč [33]

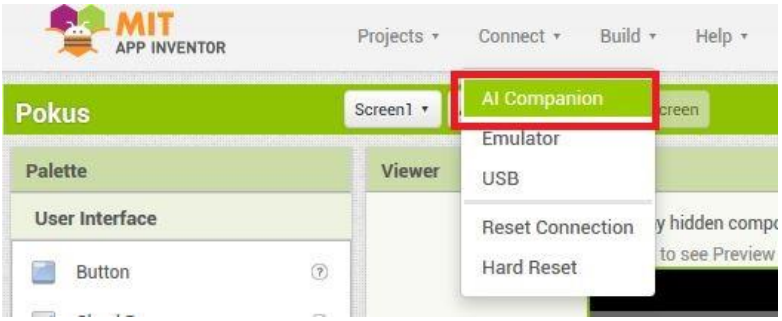
Prohlížeč	Verze
Firefox	3.6 a vyšší, je nutné mít vypnuté rozšíření NoScript
Chrome	4 a vyšší
Apple Safari	5 a vyšší
Microsoft Edge	Podporován
Internet Explorer	není podporován

Na oficiálních stránkách Microsoft Edge není uveden mezi podporovanými prohlížeči, nicméně experimentálně bylo ověřeno, že Mit App Inventor v tomto prohlížeči funguje.

Požadavky na operační systém jsou velmi mírné, postačí počítač s Windows od verze XP nebo Linux s Ubuntu od verze 8 či s verzí Debian 5 a vyšší. Macintosh je též podporován (verze s procesory Intel) a to od verze Mac OS X 10.5 [33].

Aplikace lze při tvoření testovat třemi způsoby, po otestování lze aplikaci samozřejmě stáhnout a sdílet. První možností je spouštět je v emulátoru. Pokud má vývojář k dispozici telefon s Androidem, nabízejí se dvě možnosti testování. Pro všechny tři způsoby testování je přiložen návod:

1) V první možnosti je předpokládáno, že jsou obě zařízení připojena v jedné síti, ve které je povolena komunikace mezi zařízeními.
    a) Stáhnout aplikaci „MIT AI2 Companion“ lze přes Google Play, případně z webových stránek vývojářů [33].
    b) Po instalaci aplikace je možné přistoupit k samotnému testování. Ve vývojovém okně v prohlížeči je nutné klinout na „Build“ a následně na „AI Companion“ tak, jak je ukázáno na Obrázku 1:

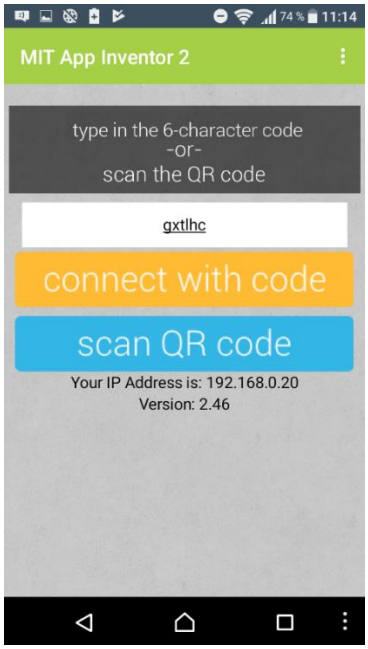


Obrázek 1 - Live testing MIT App Inventor

c) Po rozkliknutí této možnosti se objeví QR kód a 6 - místný textový kód.

Obrázek 2 - QR kód a textový kód pro Live testing

d) QR kód či textový řetězec zadejte v aplikaci MIT AI2 Companion.
Poznámka: V případě, že se při skenování QR kódu objevovala hláška „Camera Framework Bug“, je možné využít textový kód.



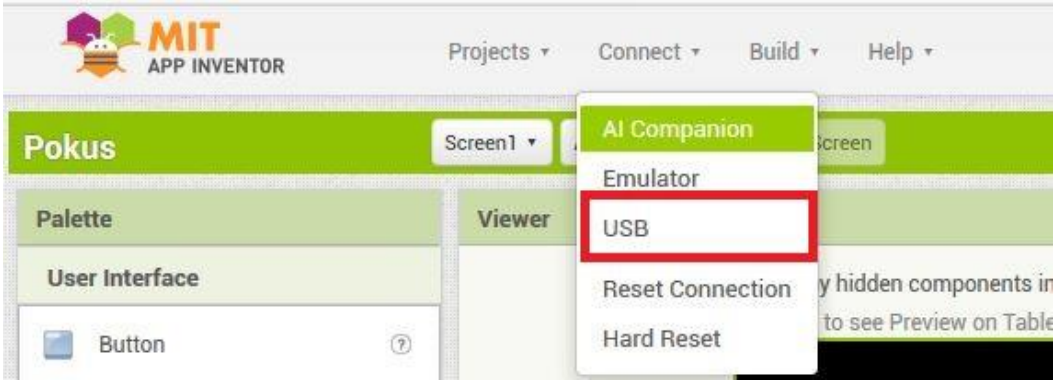
Obrázek 3 - Spárování počítače a telefonu v aplikaci MIT AI2 Companion

e) Testovaná aplikace se nyní načte do telefonu, tento přenos trval kolem 4 vteřin.

V dokumentaci App Inventoru se uvádí, že toto přenášení vyvíjené aplikace do telefonu probíhá tak, že servery App Inventoru zprostředkovávají přímou komunikaci mezi počítačem a telefonem v síti, ke které jsou obě zařízení připojeny, přes servery App Inventoru se tedy aplikace nepřenáší [34]. To, že jsou zařízení připojena do jedné sítě je nutná podmínka. Někdy může nastat problém v nastavení firewallu, který může komunikaci mezi zařízeními blokovat. Tento a podobné problémy jsou podrobněji popsány v dokumentaci App Inventoru, jako vhodné řešení se ale nabízí vytvořit přístupový bod pro žáky například z notebooku, vlastního routeru či staršího telefonu [35]. Možnost live testingu je doporučena vývojáři MIT App Inventoru, nicméně klade větší nároky na nastavení sítě.

2) Druhou možností, jak využívat live testing, je stáhnout do počítače program, který bude komunikovat s aplikací v telefonu přes USB. Pro toto je nutné udělat následující:
   a) Nainstalovat na PC „App Inventor Setup Software“ [36].

b) Nainstalovat na telefon aplikaci „Mit App Companion“. Návod je k dispozici na stránce 18 v bodě 1) a).
c) V telefonu povolit takzvaný „USB debugging“, tedy česky „Ladění USB“. Podrobný návod je sepsán na stránkách vývojářů [36].
d) Spustit na PC nainstalovaný software pod názvem „aiStarter“.
e) Připojit telefon USB kabelem. *Poznámka: Pořadí bodu d) a bodu e) je možné zaměnit*.
f) Nyní již postačí v prohlížeči v otevřeném App Inventoru kliknout na „Connect“ → „USB“ tak, jak je uvedeno na obrázku číslo 4. Následně se vyvíjená aplikace přenese do telefonu a spustí se, přičemž tato operace trvá kolem 20 sekund. Změny, které budou provedeny v App Inventoru se na telefonu projeví téměř okamžitě.



Obrázek 4 - Spuštění live testingu přes USB

Poznámka k aplikaci aiStarter: Při ukončování aplikace klasickým křížkem a následném znovuspuštění aplikace nepřenášela data do telefonu. Proto ji doporučuji ukončovat zadáním příkazu CTRL + C přímo v rozhraní aplikace.

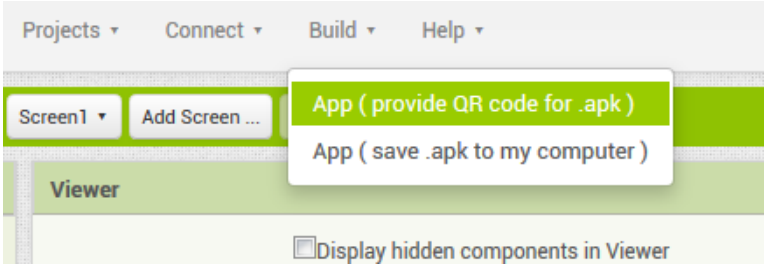
3) Pokud není k dispozici zařízení s OS Android, lze využít emulátor. Pro tuto možnost je potřebné zajistit následující:
   a) Nainstalovat na PC „App Inventor Setup Software“ [36].
      Poznámka: Po prvním spuštění emulátoru anebo když je k dispozici nová verze emulátoru, bude nutné tento emulátor aktualizovat.
   b) Spustit na PC nainstalovaný software pod názvem „aiStarter“.
   c) Ve vývojovém okně kliknout na „Connect“ → „Emulator“

Obrázek 5 - Emulátor App Inventoru

Poznámka k emulátoru: Ve srovnání s předchozími možnostmi live testingu vychází tato možnost jako nejpomalejší, neboť celá operace od nahrání aplikace až po její spuštění v emulátoru trvá kolem 90 vteřin. Reakce na změny v aplikaci jsou lehce pomalejší než u testování přes USB či Wifi. V prostředí emulátoru je použita starší verze Androidu (2.2) a také nenabízí takové možnosti, jako skutečné testování na telefonu. Například aplikace využívající různé senzory v emulátoru ozkoušet nelze. Je možné využít jiné emulátory, uvedené např. v kapitole 2.5, nicméně poté je nutné pokaždé aplikaci stáhnout a znovu instalovat. Tímto způsobem se pak ale ztrácí čas.

Pro vyvíjení v App Inventoru je tedy potřeba webový prohlížeč + v závislostí na zvolené metodě live testingu příslušný software. Osobně bych doporučil testování přes USB, neboť tato možnost neklade v porovnání s Wifi testováním takové nároky na nastavení sítě a v porovnání s emulátorem vychází USB jako rychlejší a také je možné využívat všech senzorů, které jsou na testovacím telefonu k dispozici.

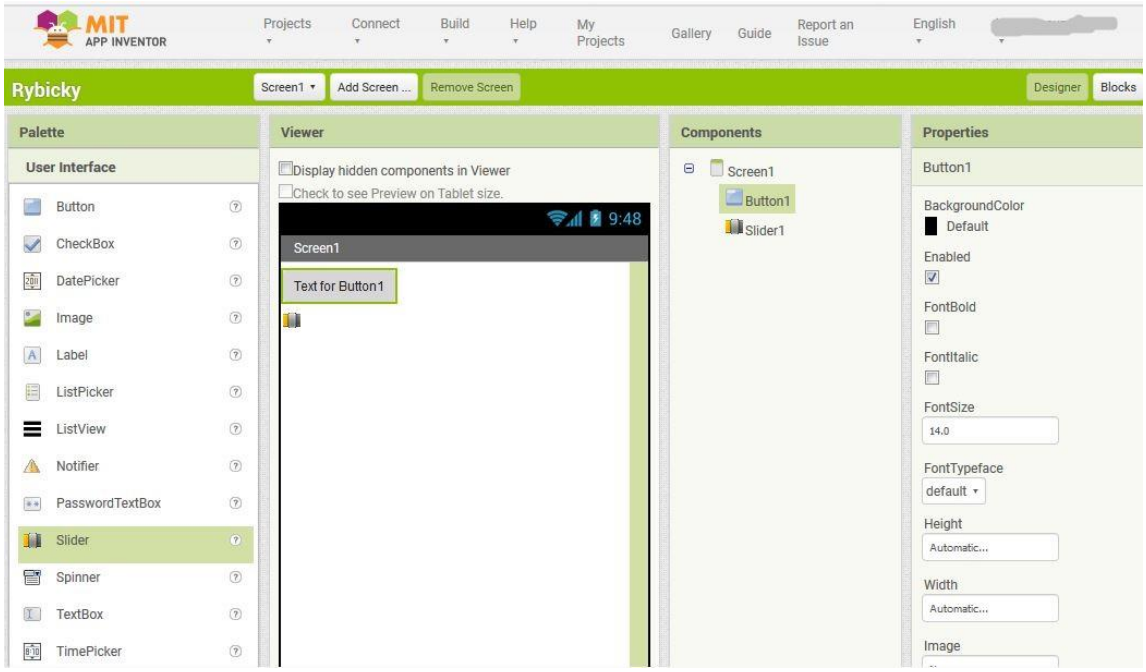
Otestovanou aplikaci lze stáhnout do telefonu či počítače v závislosti na zvolené volbě po rozkliknutí možnosti „Build“ → „App (Provide QR code for .APK)“ anebo „App (save .APK to my computer).



Obrázek 6 - „Build aplikace“

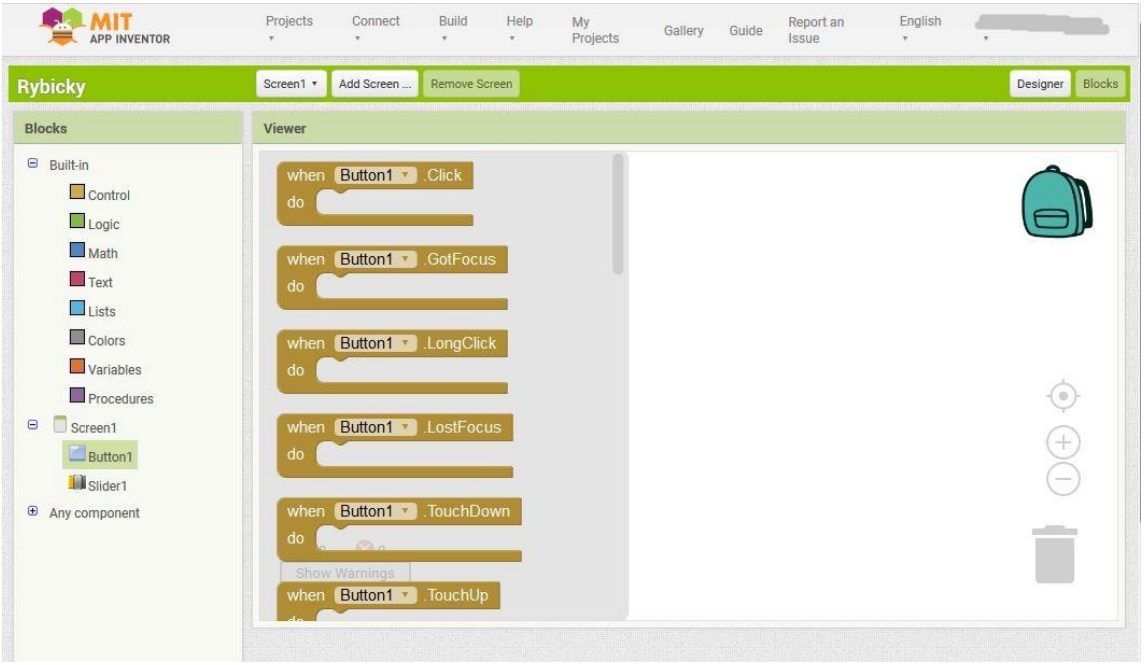
2.1.2 Srozumitelnost vývojového prostředí MIT App Inventor pro žáky

Vývojové prostředí je velmi přehledné, je členěno do dvou hlavních částí, a to na část „Designer“ a na část „Blocks“. V části Designer si programátor nastaví vizuální stránku jednotlivých oken aplikace, může přidat obsluhující prvky jako jsou tlačítka, menu, slidery a další. Těmto komponentám může přiřazovat různé vlastnosti, jako jsou barva, velikost, popisky a podobně.



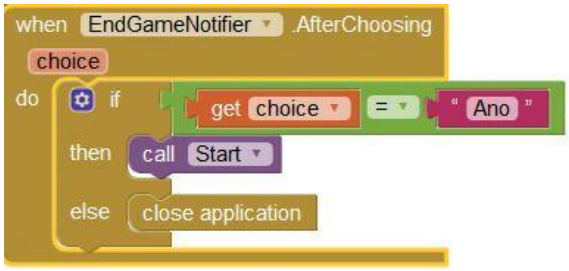
Obrázek 7 - Vývojové prostředí MIT App Inventor, část Designer

Po přepnutí do části „Blocks“ se již tvoří program, a to pomocí přetahování bloků s kódem. V této části jsou k dispozici obvyklé prostředky pro programování jako jsou proměnné, procedury, smyčky a další.



Obrázek 8 - Vývojové prostředí MIT App Inventor, část Blocks

Práce v App Inventoru je velmi příjemná a intuitivní, neboť kód, který člověk tvoří, je k okamžitě vidět na monitoru a je možné si průběh programu lépe představit. Každý blok kódu připomíná jednu kostičku puzzle a může tak zapadnout jen do některých jiných bloků, do některých nezapadne. Tímto programátoři MIT App ošetřili případné chyby programátorů, aby například do podmínky „if“ nebylo možné vložit blok „close application“. Jak je vidět na obrázku číslo 9, nelze poslední jmenovaný blok vložit do podmínky, ale až do části „then“ či „else“. Prostředí je velmi podobné projektu Scratch [18].



Obrázek 9 - Bloky kódu v IDE MIT App Inventor

Primárním jazykem vývojového prostředí je angličtina, mezi dalšími nabízenými jazyky čeština chybí.

Pokud bude v App Inventoru vyvíjena větší aplikace, může se prostředí stát nepřehledným, neboť organizace všech bloků je možná pouze přetahováním myši.

2.1.3 Programovací jazyk, který se používá pro kódování v IDE MIT App Inventor

U App Inventoru nelze hovořit o použití nějakého „oficiálního“ programovacího jazyku, spíše hovoříme o vizuálním programovacím jazyku. Tento styl programování je vhodný zejména pro začátečníky a lze jej tedy i dobře použít při výuce na školách. Zajímavé rozšíření, které App Inventor nabízí, je kompatibilita s LEGO MINDSTORMS [37]. V App Inventoru je tedy možné napsat aplikaci, která spolupracuje s roboty z Lega. Jedinou podmínkou k této funkcionalitě je mít k dispozici na telefonu Bluetooth. K možnostem App Inventoru patří i možnost práce s různými databázemi, jako jsou TinyDB, TinyWebDB, CloudDB od Redis a FirebaseDB. Na tyto příklady je možné aplikovat začátečnickou výuku databází. Zajímavou možnost nabízí server appinventortojava.com [38], který nabízí takzvaný „Java bridge“, díky kterému je možné aplikace napsané v App Inventoru přeložit do Javy a tímto způsobem tak žáky připravit na programování v Javě a potažmo v oficiálním vývojovému prostředí pro Android, kterým je Android Studio.

2.1.4 Podpora k IDE App Inventor a k programovacímu jazyku

Podpora k tomuto IDE je široká, nabízí hodně materiálů, a to jak internetové zdroje, tak zdroje knižní. Jak bylo uvedeno v předchozí kapitole, App Inventor používá vizuální programovací jazyk vázaný na vlastní prostředí, proto podpora pro IDE i programovací jazyk je totožná.

Knižní zdroje, které pojednávají o App Inventoru je možné dohledat na oficiálních stránkách [39], níže jsou uvedeny alespoň některé.

Kniha *App Inventor* od Davida Wolbera [32] byla citována v této diplomové práci a je k dispozici v češtině. Tato kniha byla v originále vydána v roce 2011, českého překladu se dočkala až v roce 2014. V současné době je tato kniha již poněkud zastaralejší, neboť odkazuje na již neexistující adresy a na obrázcích je ještě vidět

starší prostředí App Inventoru. V roce 2014 ale vyšla v originále první verze knihy App Inventor 2 a 13.4.2018 druhá verze App Inventoru 2. K pozitivum této knihy patří to, že jsou programátorské dovednosti ukazovány na konkrétních příkladech, které si žáci mohou sami vyzkoušet a dobře pochopit. Kniha je k dispozici zdarma v aktuální verzi on-line [40].

Kniha *Building android apps in easy steps: covers app inventor 2. edition* funguje jako průvodce od nastavení App Inventoru až po vývoj složitějších aplikací [41].

Kniha od Sarag Guthalsové *Building a mobile app: design and program your own app!* je určena pro naprosté začátečníky a pro menší děti [42].

Karl-Hermann Rollke vydal v lednu roku 2018 knihu *Android Apps with App Inventor 2: Easy App Development for Everyone*. Jedná o knihu, ve které jsou k nalezení návody krok za krokem. Tato příručka se nezaměřuje na komplexní popis programovacího jazyku, ale na demonstraci možností tohoto vývojového prostředí [43].

Elektronické zdroje:

Na oficiálních stránkách App Inventoru je k dispozici mnoho materiálů pro učitele, které jsou k nalezení v sekci „Resources“. V této sekci je vhodné se zmínit o podsekci „Teach“, ve které je možné vyfiltrovat požadované materiály dle jazyku a typu. Mezi nabízenými materiály jsou výuková videa nebo ukázkové aplikace [44]. V podsekci „Tutorials“ je možné si vyhledat požadovaný tutoriál například dle senzorů, které aplikace využívá [45]. K těmto návodům je přidán okomentovaný zdrojový kód, který lze stáhnout a následně do App Inventoru naimportovat.

V českém prostředí je výborně metodicky i odborně zpracován minikurz programování pro App Inventor od Západočeské univerzity v Plzni na dvanáct dvouhodinových lekcí v rámci projektu Popularizace vědy. V každém pracovním listě je odhad na dobu trvání příslušné aktivity, podrobný popis aktivity a samozřejmě cíl hodiny. Pokud žáci budou pracovat rychleji, jsou k dispozici i návrhy na modifikaci aplikace. Materiály jsou včetně zdrojových kódů. Tento minikurz je možné využívat zdarma po přihlášení v roli hosta do e-learningového prostředí Západočeské univerzity [46].

Velkou nabídku materiálů nabízí server appinventor.org [47]. Na této stránce je možné nalézt sadu tutoriálů, výukových videí, didakticky zpracované materiály pro výuku, které jsou kombinovány s učebnicí App Inventor uvedenou na stránce 26, a také glosář dotazů, na které jsou přehledně zpracované odpovědi. Tento server nabízí i již zmíněný Java bridge (kapitola 2.1.3).

Po zadání fráze MIT App Inventor 2 do serveru Youtube.com je také možné nalézt nepřebernou sbírku tutoriálů.

Problémy s tvorbou aplikací či se samotným App Inventorem lze řešit na oficiálním diskusním fóru [48].

2.2 Android Studio

Android Studio bylo publikováno poprvé v květnu roku 2013, oficiálním vývojovým prostředím se stalo až v prosinci 2014, kdy vyšla verze 1.0 [49], v době psaní této diplomové práce byla aktuální verze 3.1.2, která vyšla v dubnu 2018 [50]. Android Studio je IDE založené na IntelliJ IDEA [51], které je speciálně uzpůsobené pro vývoj aplikací pro Android.

2.2.1 Čas strávený instalací, nastavováním a následnými aktualizacemi s prostředím Android Studio

Android Studio je dostupné pro všechny tři dominující operační systémy na počítačích, kterými jsou Windows, Mac a Linux. Minimální požadavky na systém jsou uvedeny v Tabulce 2:

Tabulka 2 - Požadavky na instalaci Android Studia [50]

OS	**Windows 7,8,10 (32- nebo 64-bit), Mac OS X 10.10 – 10.13, Linux GNOME nebo KDE PC, testováno na Ubuntu 14.04**
RAM	minimálně 3 GB, doporučeno 8 GB, plus 1 GB pro Android Emulátor
Disk	2 GB minimum, 4 GB doporučeno (500 MB pro IDE a 1,5 GB pro Android SDK a systémový obraz emulátoru
Rozlišení obrazovky	1280 x 800

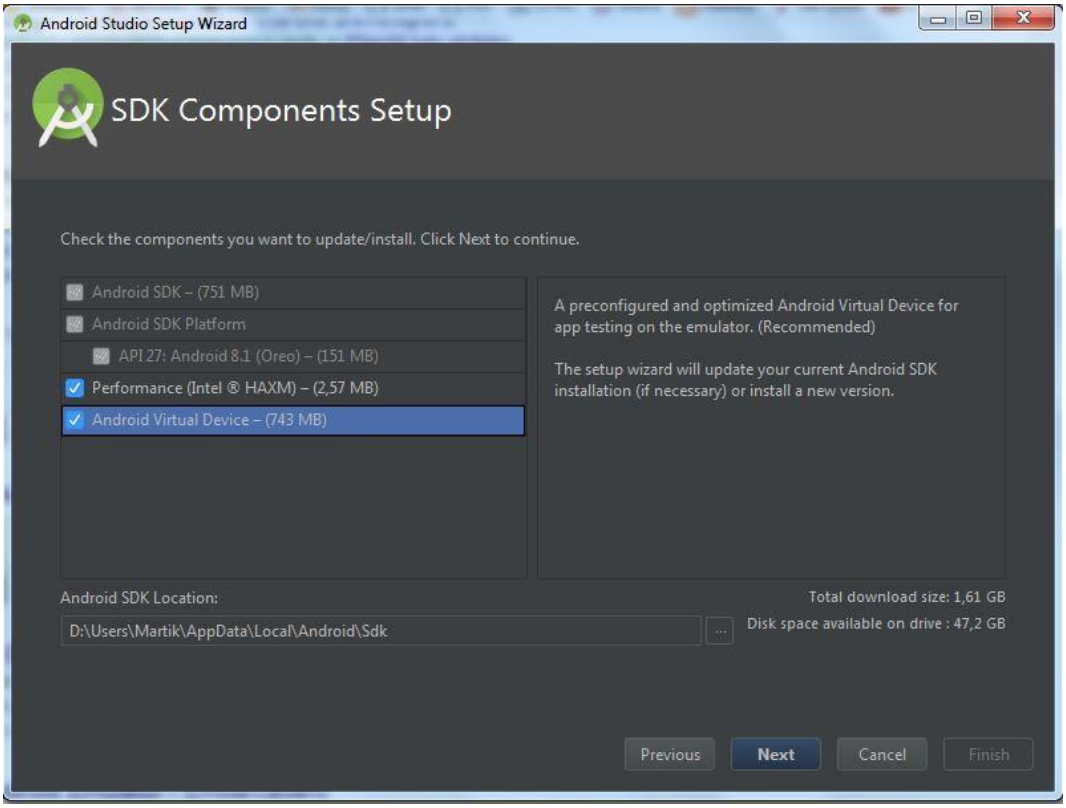
Z tabulky je vidět, že je možné snížit nároky na počítač tím, že nebude instalován a využíván emulátor. V tabulce je též uvedeno, že Android Studio potřebuje pro běh programu 3 GB paměti RAM, nicméně reálně využívalo během mého testování asi 1 GB paměti RAM.

Samotný instalační balíček má v závislosti od požadovaného OS velikost od 750 MB do 850 MB. Při instalaci bude ještě třeba stáhnout další data, objem těchto dat závisí na tom, zda si vývojář přeje instalovat i emulátor. I s emulátorem bylo na OS Windows potřebné přenést kolem 1,6 GB, bez emulátoru 857 MB. Před instalací Android Studia je dobré zjistit, zda na vývojářském PC půjde emulátor vůbec spustit. Kromě požadavků uvedených v tabulce číslo dvě musí hardware splňovat podmínky uvedené na stránkách Android Studia. [52]. V případě, že není možné spustit na vývojářském PC emulátor, je možné aplikace testovat na připojeném mobilním telefonu s OS Android, případně je možné využít konkurenční emulátor s nižšími požadavky na hardware, seznam takových emulátorů je uveden v kapitole 2.5.

Doba celkové instalace po stažení instalačního balíčku i s emulátorem trvala na referenčním notebooku (viz. kapitola 2) zhruba hodinu. Pro instalaci a základní nastavení Android Studia bude potřebné vykonat následující kroky:

1) Stáhnout instalační balíček z oficiálních stránek Android Studia [50].
2) Zahájit instalaci. Instalaci je možné si upravit dle vlastních preferencí volbou „Custom“. Je například zbytečné instalovat Android Virtual Device v případě, že není možné na cílovém PC emulátor spustit. Stejné to je i

v případě, že není možné využít hardwarovou akceleraci. V průběhu instalace se ještě dostahují vybrané balíčky.



Obrázek 10 - Ukázka možností volby komponent Android Studia

3) Instalace je dokončena.

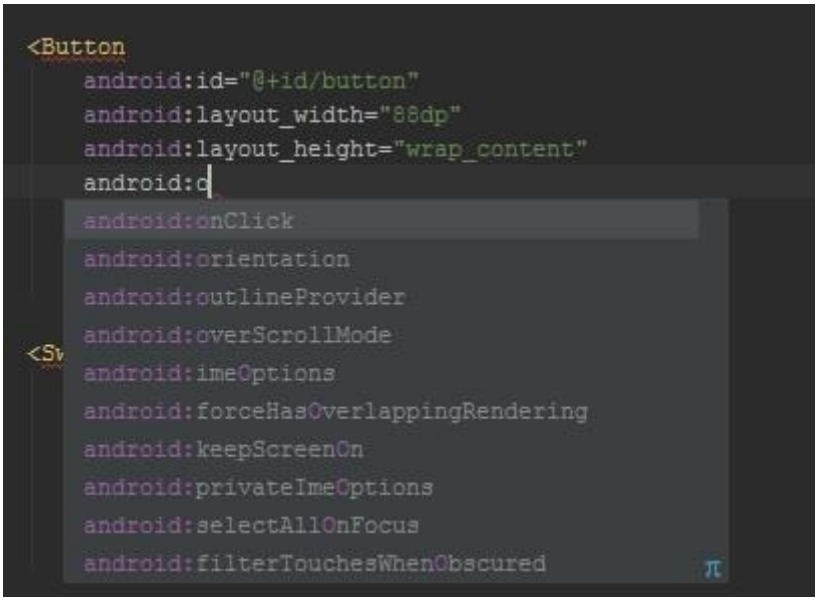
Aktualizace Android Studia a příslušných komponent je velice intuitivní, neboť Android Studio při každém startu zkontroluje, zda jsou nějaké aktualizace k dispozici a nabídne jejich instalaci. Pro správu a doinstalování dalších funkcionalit slouží v Android Studiu v části „Tools“ SDK Manager, ve kterém je možné na dvě kliknutí doinstalovat potřebné ovladače či další služby.

2.2.2 Srozumitelnost vývojového prostředí Android Studio pro žáky

Android Studio je oficiální IDE pro OS Android. Proto je tato aplikace na jedné straně velmi robustní, na straně druhé ovšem nabízí i velké množství funkcionalit.

Android Studio nenabízí český překlad, celé je k dispozici pouze v angličtině.

Vzhled aplikace lze upravovat buď v části „Design“, ve které stačí přetahovat potřebné komponenty na zobrazený displej telefonu a následně jim upravovat vlastnosti, anebo v části „Text“, ve které lze komponenty přidávat a nastavovat jim vlastnosti kódem v jazyce XML. Obsluhu komponent, tedy vlastní program se píše po přepnutí do části, která je označena „JménoAktivity.java“, ovšem pouze v případě, že je projekt psán v jazyce Java (aplikace je možné psát i v jazyce Kotlin). Programování usnadňuje „našeptávání“ kódu. Android Studio kód neustále kontroluje a v případě, že došlo k nějaké nesrovnalosti, upozorní na ni.



Obrázek 11 - Našeptávání kódu v Android Studiu

Android Emulátor, který je součástí Android Studia, umožňuje aplikace spouštět a testovat. Je možné simulovat příchozí hovory, naklápění telefonu, nastavovat GPS polohu a další.

Jako zajímavá možnost se jeví propojení vývojového prostředí MIT App Inventor s Android Studiem, kdy lze vypracovaný projekt z MIT App Inventoru díky serveru appinventortojava.com [38] online převést do Java projektu a tento projekt následně naimportovat do Android Studia a učit tak žáky na již známé aplikaci. Tento postup je podrobně popsán v dokumentaci projektu [53], osobně jsem postup testoval a funguje výborně.

2.2.3 Programovací jazyk, který se používá pro kódování v IDE Android Studio

V Android Studiu lze programovat primárně v Javě, Kotlinu, přidávat skripty lze i v jazycích C a C++ [50]. Dle serveru Tiobe.com, který tvoří tabulku nejpopulárnějších Turing-kompletních[6] programovacích jazyků na základě dotazů zadaných do určitých vyhledávačů, je Java nejpopulárnějším jazykem [54]. Server Helloworld.cz vydal článek [55], ve kterém analyzoval počet nabídek práce na serveru Indeed.com při dotazech právě na populární programovací jazyky. Na prvním místě se objevilo SQL, na místě druhém Java. Z výše uvedeného je vidět, že vyučovat programovací jazyk Java má pro žáky jistě potenciál.

2.2.4 Podpora k IDE Android Studio a k programovacímu jazyku

Vzhledem k tomu, že je Android Studio oficiálním vývojovým prostředím pro OS Android, je podpůrných materiálů k dispozici dost. Uvádím alespoň některé vybrané:

Knižních zdrojů, které by byly k dispozici v češtině a věnovaly se vývoji v Android Studiu je málo. Za tento fakt může možná i to, že oficiálním IDE pro Android se Android Studio stalo až v roce 2016 [56]. V tomto roce byla též ukončena podpora doplňku Android Developer Tools pro IDE Eclipse.

Mistrovství Android: Kompletní průvodce vývojáře od autora Ľuboslava Lacka z roku 2017 pokrývá velmi dobře celou tematiku programování pro OS Android. Kniha provádí čtenáře od instalace a nastavení Android Studia přes napsaní vlastní aplikace až po její nasazení v Google Play. V knize jsou k dispozici i zdrojové kódy k ukázkám a vše je dobře vysvětleno [57].

Programujeme hry pro Android 4 od J.F. DiMarzio je starší publikace z roku 2012 a funguje celá jako průvodce pro vytvoření vesmírné střílečky [58].

Na oficiálních stránkách Android Developers [59] je k dispozici jak robustní dokumentace k Android Studiu, tak ukázkové kódy a dokonce je možné i zdarma využít interaktivní e-learning v službě Udacity.com

[6] Turing kompletní programovací jazyk umí cykly, podmíněné operace, sekvenční operace a práci s pamětí. Mezi Turing kompletní jazyky se tedy nepočítá například HTML. http://moodle.ics.muni.cz/moodle-site/mod/page/view.php?id=7279

Online kniha či sbírka tutoriálů je dostupná na vogella.com [60]. Vše je psané sice v anglickém jazyce, nicméně sbírka příkladů je skutečně velká.

Server itnetwork.cz nabízí vícero seriálů obsahujících přehledné tutoriály většinou pro jazyk Java [61], výrazně méně jich je k dispozici pro jazyk Kotlin [62].

2.3 Visual Studio

Visual Studio je IDE vydávané společností Microsoft. V době psaní této diplomové práce bylo ve verzi 15.7.5 a k dispozici byla preview verze 15.8. Visual bylo zařazeno do srovnání IDE hlavně z důvodu avizované možnosti Cross-Platform developmentu, tedy možnosti, jak vyvíjet aplikace v jednom programovacím jazyce, konkrétně v jazyce C# (případně i F#) a publikovat je na platformách Android, iOS a UWP[7] [64]. Tento vývoj je možný díky nástroji Xamarin. Společnost Microsoft koupila Xamarin v roce 2016 [65, s.12] a od tohoto data ho začala poskytovat zdarma jako volitelnou součást Visual Studia. Ve Visual Studiu je možné také psát aplikace pro Windows, Microsoft Office, psát weby, hry a další [66].

2.3.1 Čas strávený instalací, nastavováním a následnými aktualizacemi s prostředím Visual Studio

Visual Studio ve verzi Community[8] je zdarma šířené IDE pro OS Windows a Mac, v následujících tabulkách jsou uvedeny minimální požadavky na PC:

Tabulka 3 - Požadavky na PC pro instalaci Visual Studia na Windows [67]

Systém:	**Windows od verze 7 se SP1**
Procesor:	alespoň 1,8 GHz, doporučuje se dvoujádrový procesor
RAM:	2 GB, doporučeno 4 GB
Disk:	typická instalace 20 – 50 GB

[7] UWP je „Universal Windows Platform“, která umožňuje spustitelnost na desktopech, tabletech a telefonech s Windows 10 a také na Xboxu, HoloLens a Internet of Things [63].
[8] Verze Community je určena pro studenty, vývojáře open-source a individuální vývojáře, lze ji tedy používat i na školách.

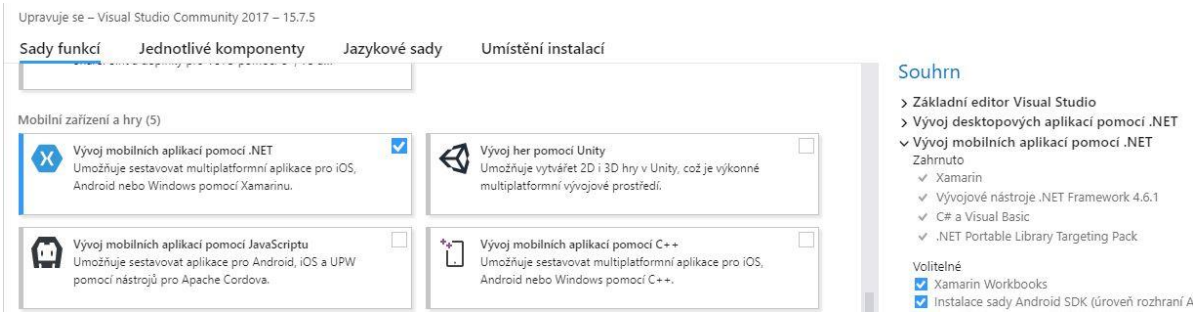
Tabulka 4 - Požadavky na PC pro instalaci Visual Studia na macOS [68]

Systém:	**macOS Sierra 10.12, Mac OS X El Capitan 10.11**
Procesor:	alespoň 1,8 GHz, doporučuje se dvoujádrový procesor
RAM:	4 GB, doporučeno 8 GB
Disk:	1 GB

Požadavky na Xamarin jsou stejné, jen se lehce liší v požadavku na macOS, kde je požadován macOS Sierra 10.12 při instalovaném prostředí Xcode 8.3.[9] Je vhodné doplnit informaci, že pro vývoj iOS aplikací ve Visual Studiu je nutné vlastnit Mac počítač, který se s Visual Studiem spáruje [65, s.14]. A dále, pro vývoj UWP aplikací je nutné instalovat Visual Studio na počítače s OS Windows 10 [70]. Visual Studio má připravené dvě možnosti pro instalaci – pro PC se slabým internetovým připojením je připravena speciální „offline“ instalace [71]. Pro instalaci Visual Studia 17 Community a nástrojem Xamarin za pomoci doporučeného instalačního programu je potřeba udělat následovné:

1) Stáhnout instalační program Visual Studio Community 17 [72]. V doporučeních pro instalaci je [73]. doporučen i restart počítače před samotnou instalací.
2) V programu vyberte komponenty, které chcete instalovat, pro vývoj Cross-Platform aplikací je nutné stáhnout položku Xamarin.
   Poznámka: Visual Studio hlásilo při sestavování aplikací pro Android chybu, která se opravila stáhnutím .NET Frameworku ve verzi 4.7.1, proto doporučuji vybrat i tuto komponentu v záložce „Jednotlivé komponenty“ [74].

[9] Xcode je IDE od společnosti Apple [69].



Obrázek 12 - Volba komponent v instalačním souboru Visual Studio

3) Instalace je dokončena.

Celková doba instalace a stahování závisí samozřejmě na volbě vybraných komponent a na rychlosti internetového připojení. Na referenčním notebooku, jehož vlastnosti jsou uvedeny v kapitole 2, trvala instalace při zvolených komponentách „Vývoj desktopových aplikací pro .NET“ a „Xamarin“ kolem jedné hodiny.

Testovat aplikace je možné více způsoby a pro některé z nich bude třeba ještě instalovat další aplikace či komponenty. Je tedy možné:

1) Testování na připojeném fyzickém zařízení, kdy se aplikace sestaví a nahraje do telefonu. Tato volba se spustí kliknutím na název telefonu, jak je ukázáno na Obrázku 13., řádek první. (Při testování aplikace pro Android je telefon připojený k vývojářskému PC, při testování iOS aplikace je telefon s iOS připojen k počítači Mac, viz. kapitola 2.3.1). Kompilace aplikace a její první přenos do telefonu tímto způsobem trval kolem mezi 2 a 3 minutami, další kompilace trvaly do 20 vteřin.
2) Živé testování, ve kterém jsou za pomocí Xamarin Live Player (XLP) vidět změny v kódu aplikace přímo v telefonu [75]. Pro tuto možnost je nutné:
   a) Mít Visual Studio 2017 ve verzi alespoň 15.4 a vzhledem k tomu, že telefon je připojen k Visual Studiu přes WiFi, je nutné, aby telefon a PC byly ve stejné síti [75].
   b) Stáhnout aplikaci do testovacích telefonů. Stejnojmenná aplikace je dostupná pro telefony s Android na Google Play [76], a pro zařízení s iOS je k dispozici preview verze, pro jejíž získání je třeba se zaregistrovat na stránkách Microsoftu [77].

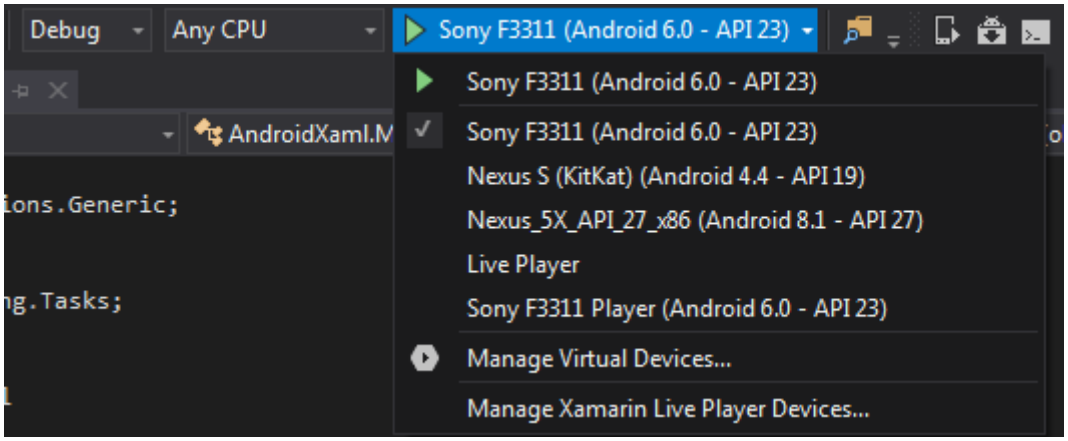
c) Spustit aplikaci na telefonu a při prvním spuštění je nutné spárovat zařízení s Visual Studiem. Ve Visual Studiu je třeba kliknout na „Tools“ → „Xamarin Live Player“ → „Manage Devices“ (Pokud není volba XLP k dispozici, přestože je Xamarin nainstalovaný, je nutné jej povolit v „Tools“ → „Options“ → „Xamarin“ → „Other“ → „Enable XLP“). Poté se naskenuje zobrazený QR kód aplikací v telefonu anebo se do ní zadá číselný kód.
d) Kliknout ve Visual Studiu na spárovaný telefon s „přívlastkem“ Live, následně dojde k přenosu aplikace do telefonu. Tato možnost je na obrázku číslo 13 uvedena jako šestá v pořadí.

Doba kompilace a přenosu se pohybovala v řádu vteřin. Živý náhled je možné spustit v „Tools“ → „Xamarin Live Player“ → „Live Run Current View“. Náhled se aktualizuje po každém uložení změn v kódu. V případě, že se nedaří přenést aplikaci do telefonu, doporučuji telefon restartovat a aplikaci Xamarin Live a Visual Studio znovu spárovat.

3) Využít Microsoft Visual Studio Emulátor pro Android. Microsoft v požadavcích na systém uvádí celkem náročné požadavky na hardware a systém (minimálně 6 GB RAM, podporu pro Hyper-V[10], Windows ve verzi Pro a systému 8 a vyšší) [79] tyto požadavky jsou při instalaci kontrolovány a pokud nejsou splněny, k instalaci ani nedojde. Mezi výhody tohoto emulátoru patří i možnost vytvoření testovacího telefonu se zadáním vlastních parametrů.
4) Testovat aplikace v Xamarin Android Playeru [80] (XAP). XAP byl na referenčním PC nainstalovaný ještě před instalací Visual Studia a po instalaci se objevil ve Visual Studiu obraz OS Android, který byl do XAP stažen (Volba 3 na obrázku číslo 13). Více informací o emulátoru XAP je uvedeno v kapitole 2.5. První spuštění emulátoru trvá asi dvě minuty, první kompilace aplikace trvá asi pět minut a při dalších spouštěních kolem 30 vteřin. Spouštění XAP z Visual Studia nedoporučuji, neboť k nahrání testované aplikace nedošlo ani po 13 minutách a IDE stále

[10] Hyper-V je nativní hypervizor který umožní vytvořit virtuální stroj na x86-64 systému běžícím na Windows [78].

čekalo na spuštění emulátoru, který přitom již spuštěn byl. Proto se jako lepší možnost jeví spouštět XAP jako samostatný program.



Obrázek 13 - Možnosti testování aplikací ve Visual Studiu

Aktualizace se ve Visual Studiu kontrolují po každém startu IDE, správa rozšíření pro IDE je možná v nabídce „Tools“ → „Extensions and Updates“.

2.3.2 Srozumitelnost vývojového prostředí Visual Studio pro žáky

Visual Studio je komplexní nástroj pro vývoj desktopových, cloudových, mobilních a dalších aplikací. V této diplomové práci bude ale hlavní zřetel brán na framework Xamarin, který umožňuje vývoj aplikací pro OS Android.

Velkým plusem pro srozumitelnost je dostupný český překlad. Čeština je nabízena jako jeden ze 13 dostupných jazyků [81]. Přesto bych doporučoval angličtinu také stáhnout, a to z důvodu větší pravděpodobnosti nalezení odpovědi či podpory k případné chybě na anglicky psaných webech.

Vzhled aplikací lze určovat pouze „textově“, kdy se komponenty definují v buď kódem v C# anebo za pomocí jazyka XAML.[11] Celkovou práci v IDE zpříjemňuje IntelliSense[12].

[11] XAML z anglické zkratky Extensible Application Markup Language je značkovací jazyk založený na XML a využívaný k definici grafického prostředí v aplikacích společnosti Microsoft [82].
[12] IntelliSense je nástroj od společnosti Microsoft, který zjednodušuje psaní kódu našeptáváním či radami [83].

2.3.3 Programovací jazyk, který se používá pro kódování v IDE Visual Studio

Visual Studio nabízí podporu pro mnoho programovacích jazyků, uvádím alespoň ty nejzásadnější: C#, F#, B, C++, JavaScript či Python [84]. Výše zmíněné jazyky v žebříčku serveru Tiobe obsadili druhé až šesté a osmé místo [54]. Dle serveru helloworld.cz [55]. jsou mezi devíti nejžádanějšími jazyky, ve kterých lze programovat ve Visual Studiu, zastoupeny Python, JavaScript, C# a C++. S Visual Studiem je tedy možné žáky učit v nejžádanějších programovacích jazycích.

2.2.4 Podpora k IDE Visual Studio s doplňkem Xamarin a k programovacímu jazyku

Pro vývoj mobilních aplikací v Xamarinu jsou dostupné jak papírové, tak elektronické knihy a k dispozici jsou také zdroje elektronické. Nepodařilo se mi nalézt žádnou knihu věnující se Xamarinu v českém jazyce.

V dubnu roku 2018 byla na blogu Xamarinu [85] zveřejněna stránka se seznamem knih, které se věnují právě vývoji aplikací ve frameworku Xamarin. Ke každé knize je uveden krátký popis, čemu se kniha věnuje a také rok, ve kterém byla vydána.

Online a zdarma dostupná kniha *Xamarin.Forms Succintly* od Alessandra de Sole [65] vysvětluje vše potřebné pro vývoj Cross-Platform aplikace společně s krátkými ukázkami. Kniha též popisuje, jak nastavit a instalovat IDE. Autor této knihy natočil i krátký seriál „Xamarin for beginners“, o pěti videiích [86], ve kterých autor uvádí do programování s Xamarinem.

Microsoft v dokumentaci pro Xamarin [87] nabízí zdarma knihu *Creating Mobile Apps with Xamarin.Forms* od Charlese Petzolda. Na více než tisíci stránkách je popsáno fungování Xamarin.Forms spolu s krátkými ukázkami.

Na webu Xamarin [88] jsou dostupné návody pro začínající vývojáře, velké množství ukázkových kódů a také tzv. „Xamarin University“. Pod položkou Xamarin University se skrývá kvalitní e-learningový kurs, který obsahují vysvětlení, otázky, kvízy, ukázkové kódy a videa.

K dispozici je pro komunitu Xamarin vývojářů také fórum, na kterém lze nalézt různé užitečné rady, či položit vlastní otázky [89].

Obzvláště u Xamarinu doporučuji sledovat aktuální dění, například v knize Xamarin.Forms Succintly uvedené v roce 2017 se píše o očekávaných změnách,

které již opravdu nastaly. Jedná se například o možnosti tvorby Cross-Platform projektů za pomoci „PCL Libraries“, která již není aktuální, tyto projekty se tvoří jako „.NET Standard“, který v nové verzi převzal výhody PCL Libraries [90], anebo jako „Shared Project“ (srovnání těchto tří přístupů k vývoji Cross-Platform aplikací nabízí například kniha Xamarin.Forms. Succintly [65, s.30 -37]. Nicméně i oficiální dokumentace na stránkách Microsoftu stále hovoří [91] o třech možnostech tvorby, stejně tak i Xamarin University [92]. Přičemž jiná stránka z dokumentace Microsoftu mluví již správně o dvou možnostech [93]. Tuto neaktuálnost obsahu považuji za velké minus. Doporučuji proto zabývat se a učit se jen z těch nejaktuálnějších článků.

K vývojovým jazykům C# a F# je k dispozici dokumentace na stránkách Microsoftu [94], případně na českém webu itnetwork.cz lze nalézt tutoriály k jazyku C# [95].

2.4 Shrnutí výsledků tří vývojových prostředí

V této podkapitole je popsáno závěrečné shrnutí a výsledky tří prostředí, které umožňují vývoj aplikací pro OS Android. Prostředí byla porovnávána v každé kategorií podle toho, jak odpovídají zadaným kritériím uvedených v kapitole 2.

Nejrychlejší pro nasazení a nejméně náročné prostředí na hardware je MIT App Inventor. Výhodou pro začátečníky může být i jednoduché vizuální prostředí. Nevýhodou oproti druhým IDE je, že nelze hovořit u tohoto IDE o nějakém programovacím jazyku. Visual Studio nabízí širokou paletu programovacích jazyků a také možnosti vývoje pro různé platformy, potažmo vyvíjet pro více platforem v jednom programovacím jazyku. Android Studio nabízí sice programování v Javě, ale nijakou další přidanou hodnotu oproti Visual Studiu nenabízí.

Podrobnější popis ke každé z kategorií je uveden v následujících podkapitolách.

2.4.1 Čas strávený instalací, nastavováním a následnými aktualizacemi s prostředím

Tabulka 5 - Srovnání jednotlivých IDE podle času stráveného instalací, nastavováním a následnými aktualizacemi

IDE:	MIT App Inventor	Android Studio	Visual Studio
Výsledek:	+ rychlá instalace, aktualizace + malé nároky na hardware - nutnost mít Google účet	- asi hodinová instalace - větší nároky na hardware	- asi hodinová instalace - větší nároky na hardware - nutnost se registrovat

Na první místo jsem umístil MIT App Inventor, neboť příprava tohoto prostředí zabere nejméně času a také klade nejmenší požadavky na hardware. Pro běh tohoto IDE stačí nainstalovaný webový prohlížeč, pro testování aplikací je třeba stáhnout ještě malý program aiStarter. Android Studio i Visual Studio jsou komplexní IDE a toto se odráží i v nárocích na vývojářský počítač. Instalace obou posledně jmenovaných IDE trvala asi hodinu. Pro výuku je jistě efektivní, když si žáci mohou naprogramované aplikace i vyzkoušet. Za nejvíce motivační pro žáky považuji možnost testovat aplikace na vlastních telefonech, nicméně je nutné počítat s tím, že toto testování nebude možné zrealizovat na všech žákovských telefonech, neboť ne všichni žáci vlastní telefon s operačním systémem Android. Proto je dobré mít na žákovských počítačích připravený i funkční emulátor. Pokud není možné spustit, nebo z jakéhokoliv důvodu nevyhovují emulátory dodávané k Android Studiu či Visual Studiu, je nutné počítat ještě s další časovou investicí pro instalaci vhodného emulátoru. Několik samostatných emulátorů je uvedeno v kapitole 2.5.

2.4.2 Srozumitelnost vývojového prostředí pro žáky

Tabulka 6 - Srovnání jednotlivých IDE podle srozumitelnosti prostředí pro žáky

IDE:	MIT App Inventor	Android Studio	Visual Studio
Výsledek:	+ Výhody vizuálního programování - méně schopný emulátor - chybějící lokalizace do češtiny	+ kvalitní emulátor + přítomný WYSIWYG [13]editor	+ lokalizace do češtiny + kvalitní emulátor - chybějící WYSIWYG editor (Pro Xamarin projekty)

MIT App Inventor nabízí nejpřehlednější prostředí očištěné od pokročilých funkcí a možností, které by mohly žáky mást. V tomto prostředí tvoří programátor graficky jak UI aplikace, tak samotný kód. K návrhu funkční aplikace postačí znát základy programování. V Android Studiu lze UI tvořit jak graficky, tak psaním XML kódu, ve Visual Studiu je nutné UI zapisovat buď kódem v C# anebo v XAMLu. Kód obsluhující události se zapisuje již čistě textově v příslušných programovacích jazycích, tedy v Android Studiu v Javě či Kotlinu, ve Visual Studiu v C# či v F#.

Každé z IDE nabízí možnost testovat aplikace na fyzickém zařízení připojeném přes USB, speciální možnost umožňující nahrávání testovací aplikace nabízí Mit App Inventor a Visual Studio. Všechna tři IDE mají také vlastní emulátor, který umožňuje testování aplikací. Emulátor s nejmenšími nároky nabízí MIT App Inventor, nicméně není tak pokročilý, jako konkurenční emulátory, mimo jiné v tomto emulátoru běží Android ve verzi 2.2, navíc není možné testovat aplikace využívající senzory jako akcelerometr, kompas a další.

Do češtiny je lokalizováno pouze Visual Studio, nicméně pro zapisování kódu bude nutné využít angličtinu.

[13] *„WYSIWYG je akronym anglické věty „What you see is what you get“, česky „co vidíš, to dostaneš“. Tato zkratka označuje způsob editace dokumentů v počítači, při kterém je verze zobrazená na obrazovce vzhledově totožná s výslednou verzí dokumentu.“* [96]

2.4.3 Programovací jazyk, který se používá pro kódování v IDE

Tabulka 7 - Srovnání jednotlivých IDE podle užitnosti programovacího jazyku, který si žáci v jednotlivých IDE osvojí

IDE:	MIT App Inventor	Android Studio	Visual Studio
Výsledek:	- jazyk vázaný na IDE (i když je následně možné portovat projekty do Javy)	+ Java, Kotlin	+ C#, F#, B, C++, JavaScript či Python, možnost psát i konzolové či desktopové aplikace, psát weby

Pro žáky, kteří se nikdy s programováním nesetkali, doporučuji vyučovat v MIT App Inventoru. Toto prostředí je totiž vizualizované a lze žáky s malým úsilím v krátkém čase namotivovat vytvořením jednoduchých aplikací. Nicméně dovednosti, které se žáci naučí během práce s MIT App Inventorem bude nutné postupem času zúročit do nějakého na trhu poptávaného jazyka. Nabízí se možnost přenést kód z MIT App Inventoru do jazyku Java (kapitola 2.2.2) a s tímto kódem dále pracovat v Android Studiu.

S žáky, kteří jsou na pokročilejší úrovni, je vhodné pracovat v Android Studiu anebo Visual Studiu. Pro rozhodnutí, jaké z těchto dvou IDE použít může pomoci oblíbenost programovacích jazyků na trhu. Nejpopulárnějším jazykem je Java [54], jejím nástupcem by měl být Kotlin. V těchto programovacích jazycích lze psát v Android Studiu. Ve Visual Studiu je nicméně možné programovat v šesti programovacích jazycích, které jsou v první desítce nejpopulárnějších jazyků [54]. Visual Studio také díky frameworku Xamarin umožňuje psaní Cross-Platform aplikací.

2.4.4 Podpora k srovnávaným IDE a k programovacímu jazyku

Tabulka 8 - Srovnání podpory k jednotlivým srovnávaným IDE

IDE:	MIT App Inventor	Android Studio	Visual Studio
Výsledek:	+ didakticky zpracovaný jeden e-learning v češtině + kniha věnující se IDE v češtině + fórum pro řešení problémů + k dispozici vypracované materiály	+ didakticky propracovaný, robustní e-learning (pouze v AJ) + kniha věnující se IDE v češtině + fórum pro řešení problémů + k dispozici vypracované materiály	+ didakticky propracovaný, robustní e-learning (pouze v AJ) + fórum pro řešení problémů + k dispozici vypracované materiály - chybí kniha věnující se Xamarinu v češtině, pro C# knihy existují

Ke všem třem IDE jsou k dispozici komunitní fóra, na kterých lze nalézt odpovědi, případně nové otázky položit.

Didakticky zpracované materiály, které je možné využít učitelem ve výuce, mají všechny tři IDE – viz. příslušné kapitoly. Nejpropracovanější e-learningy, které zahrnují vysvětlení, videa či průběžné otázky jsou k dispozici pro Android Studio a Visual Studio s Xamarinem. Je nutné podotknout, že tyto e-learningy jsou v angličtině. Nicméně s pomocí učitele, který by případná nejasná slovíčka objasnil, je možné tyto e-learningy dobře využít při výuce. Pro MIT App Inventor je zpracován jeden e-learningový kurz v češtině.

Ke všem třem IDE je možné nastudovat a využívat knihy jak v papírové, tak v elektronické verzi, případně je možné využívat různé weby, které jsou uvedeny v odpovídajících kapitolách této diplomové práce.

Pokud pro výuku bude nutné využít materiály v češtině, vyhrává MIT App Inventor, neboť je k dispozici jak již zmíněný e-learning, tak kniha, která je počeštěna. Na druhém místě je Android Studio, pro které je možné si pořídit knihu

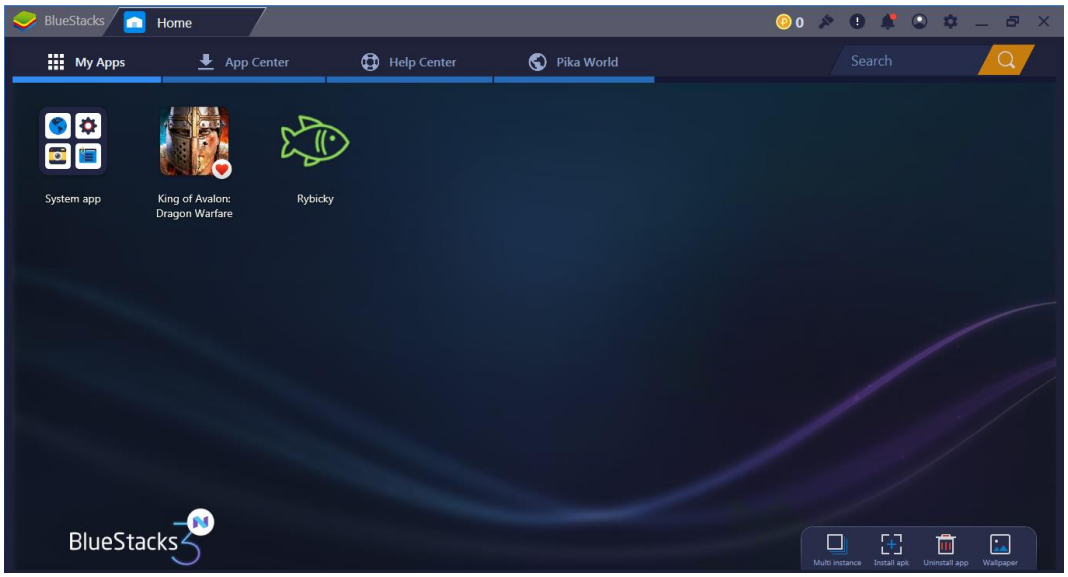
v češtině a též je možné se dovzdělat na různých webových stránkách. Pro Visual Studio a framework Xamarin existuje v češtině jen pár webových stránek.

2.5 Emulátory

V případě, že nebylo možné zprovoznit emulátor přikládaný k MIT App Inventoru, Android Studiu či Visual Studiu, je možné využít konkurenčních emulátorů.

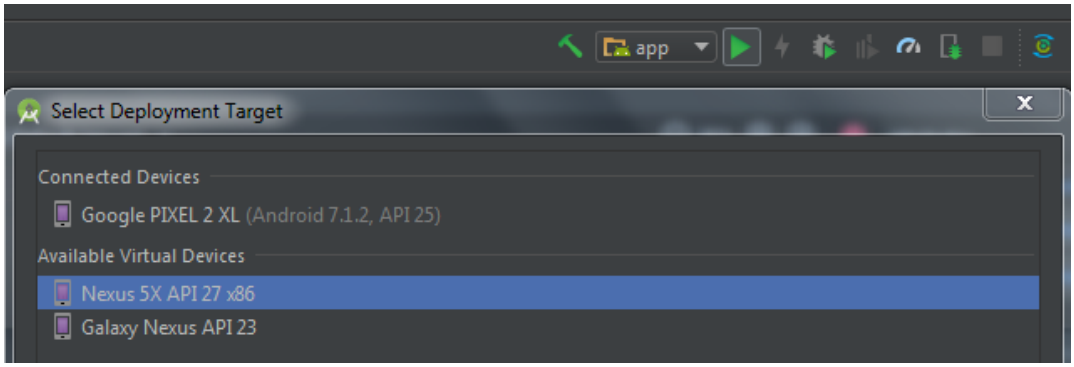
BlueStacks (ve verzi 4, tedy nejaktuálnější):

Tato zdarma šířená aplikace se na PC chová jako tablet s OS Android. Instalační balíček má asi 425 MB, na disku požaduje minimálně 4 GB, k tomu 2 GB RAM. [97]. Pokud není na PC k dispozici hardwarová akcelerace, je instalace a úvodní spuštění poněkud pomalejší. Při běžném používání a klikání je odezva lehce pomalejší, při hraní/testování náročnějších her je obraz zpomalený a dochází k zasekávání obrazu. Do BlueStacks je integrována podpora pro myš, klávesnici, mikrofon a další vstupy z PC. Díky tomu je možné nasimulovat skutečné ovládání telefonu. Co se týče reálného využití RAM, tak BlueStacks si vystačil i s 1 GB paměti RAM. Co se procesoru týče, tak aplikace využívala kolem 50 % z 2,2 GHz při nastavení na nejnižší rozlišení 1280*720 a DPI 160.



Obrázek 14 - Emulátor Blue Stacks

Velkou výhodou BlueStacks je spolupráce s Android Studiem. Po nainstalování a spuštění emulátoru se objeví v Android Studio v sekci Android Virtual Device nové připojené zařízení. Tímto zařízením je právě emulátor BlueStacks, v mém případě „Google Pixel 2 XL s Androidem 7.1.2 a API 25).



Obrázek 15 - Integrace emulátoru v Android Studiu

Genymotion:

Tento emulátor je možné spustit jak v cloudu, tak na desktopu [98]. K dispozici je více než 3000 konfigurací. Využívání služeb Genymotion je ale zpoplatněno.

Xamarin Android Player (XAP):

Tento emulátor od společnosti Microsoft z roku 2015 [99] již není nadále vyvíjen, má svého nástupce v podobě Android emulátoru [100]. Uvádím zde přesto XAP, neboť má menší nároky na hardware počítače, nevyžaduje totiž ve srovnání s

Android emulátorem Hyper-V a Windows od verze 8 a na základní testování aplikací postačí. Instalační soubor XAP je možné stáhnout z oficiálních stránek. Během instalace XAP se doinstaluje i Oracle VM Virtual Box, pokud není na cílovém PC nainstalován. Restart počítače vyřešil hlášení XAP, že není Virtual Box nainstalován. Po prvním spuštění je možné si vybrat z 12 nabízených virtuálních zařízení, které se liší rozlišením displeje a instalovanou verzí Androidu. Image vybraného zařízení je ještě potřeba stáhnout a nainstalovat, vše se ale děje velice jednoduše, na jedno klepnutí. Některá zařízení mi nešla spustit, fungoval ale například Nexus S s Androidem 4.4.2, API 19 a rozlišením 480 x 800. V liště vedle „displeje“ virtuálního mobilního telefonu je možné využívat „hardwarová“ tlačítka k ovládání telefonu.

Obrázek 16 - Prostředí emulátoru XAP

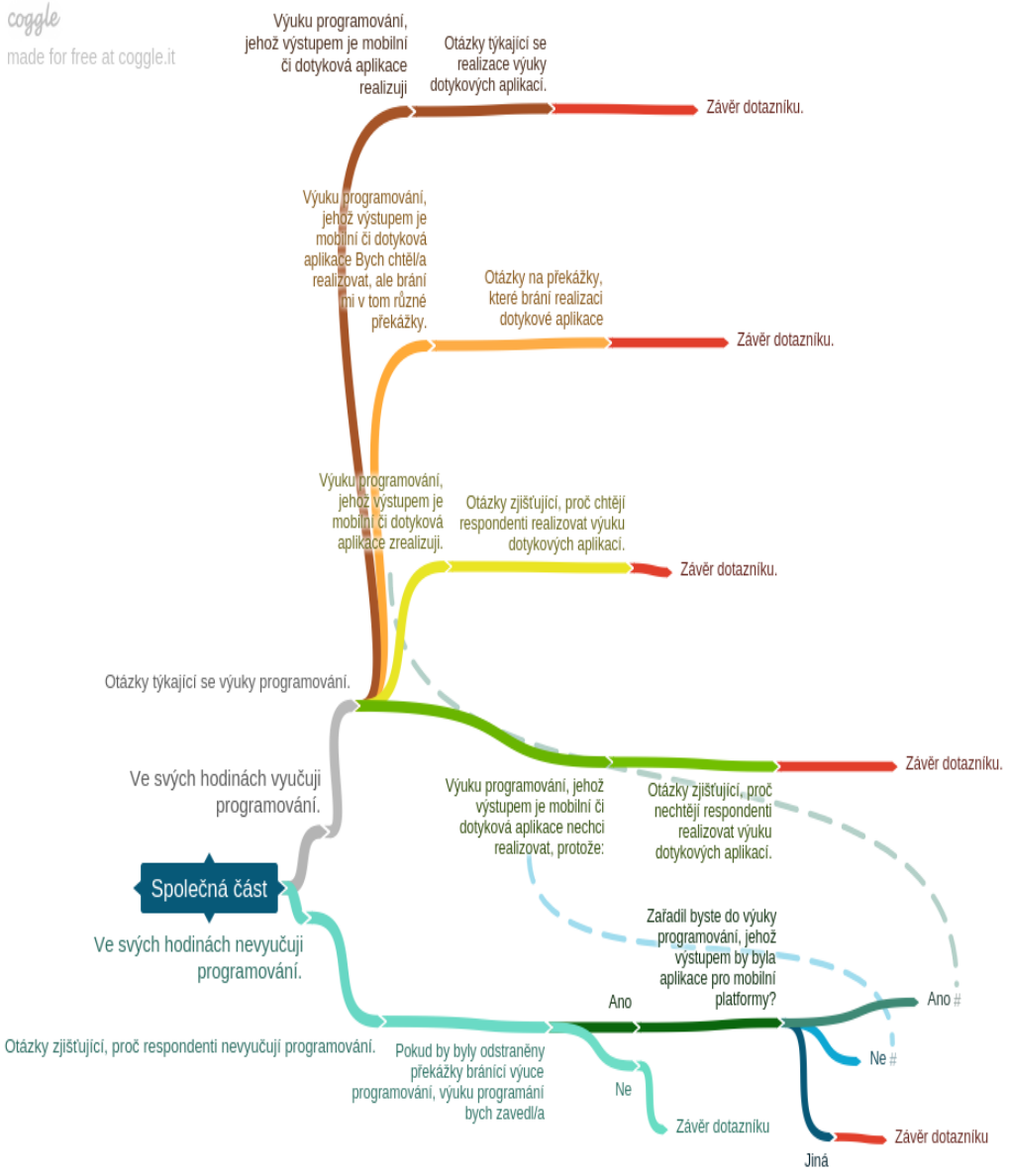
Aplikace se do spuštěného emulátoru instalují stylem „Drag and drop“, stačí tedy chytit .APK soubor a pustit jej do okna telefonu. Instalace probíhá velmi rychle. Při testování jsem narazil na problém nemožnosti instalovat aplikace, v jejichž názvu

je diakritika, proto doporučuji v případě nějaké chybové hlášky změnit název aplikace.

3 Vlastní výzkum

Cílem dotazníkového šetření bylo zjistit, jaký je skutečný stav výuky programování v českém školství. Prosba o předání on-line dotazníku učitelům informatiky, programování či vedoucím různých informatických kroužků byla adresována ředitelům základních a středních škol. Při zjišťování e-mailových adres pro základní školy bylo využito webu Seznamskol.cz [101] a pro střední školy webu Stredniskoly.cz [102]. Na těchto webech jsou školy děleny do kategorií dle krajů, přičemž byly kontaktovány školy ve všech krajích České republiky. Dotazník byl realizován službou Google Formuláře. Celkem bylo zasláno 883 e-mailů a zhruba 50 adres bylo nefunkčních. Dotazník vyplnilo 226 respondentů, jeho výsledky je možné nahlédnout v přílohách v souboru s názvem „Dotaznik.xlsx“.

Dotazník se dělí na část společnou a části, které se zobrazovaly respondentům v závislosti na tom, jak odpovídali, závěr dotazníku byl společný opět pro všechny respondenty. Celková logika dotazníku je objasněna na níže přiložené myšlenkové mapě vytvořené službou Coggle.it:



Obrázek 17 - Struktura dotazníku

3.1 Vyhodnocení dotazníku – společná úvodní část

1) Jste muž nebo žena?

Tabulka 9 - Počet vyučujících v závislosti na pohlaví respondentů

Pohlaví	Počet odpovědí	Procentuální zastoupení	Počet respondentů vyučujících programování	Procentuální zastoupení vyučujících programování
Žena	77	34,1	25	32,5
Muž	149	65,9	88	59,1

Dotazník vyplnilo 149 mužů a 77 žen, přičemž je vidět výrazný rozdíl mezi ženskými a mužskými učiteli, výuce programování se ve svých hodinách věnuje téměř 60 % mužů a 32 % žen.

2) Jaký je Váš věk?

Tabulka 10 - Počet respondentů vyučujících programování v závislosti na věkovém rozložení

Věk	Počet odpovědí	Procentuální zastoupení	Počet respondentů vyučujících programování v dané věkové skupině	Procentuální zastoupení vyučujících programování v dané věkové skupině
18 - 23	7	3,1	4	57,1
24 - 29	19	8,4	10	52,6
30 - 35	31	13,7	19	61,3
36 - 41	46	20,4	25	54,3
42 - 47	28	12,4	13	46,4
48 - 53	30	13,3	9	30,0
54 - 59	36	15,9	19	52,8
60 - 65	25	11,1	12	48,0
66 +	4	1,8	2	50,0

Z tabulky je vidět nejsilnější zastoupení vyučujících ve skupinách 36 – 41 let a poté 54 – 59 let. V tabulce je také porovnáván počet a procenta

respondentů vyučujících programování v dané věkové skupině. Výsledky jsou poměrně vyrovnané, nejvíce programování vyučuje skupina učitelů ve věku 30 – 35 let, na druhém místě se umístili mladí učitelé ve věku 18 – 23, tedy učitelé, kteří ještě studují vysokou školu, popřípadě svoje povolání vykonávají krátce. Nejméně se programování vyučuje ve věkové skupině 48 – 53 let, zde programování vyučuje každý třetí učitel.

3) Na jaké škole učíte?

Tabulka 11 - Počet respondentů vyučujících programování v závislosti na typu školy

Typ školy	Počet respondentů působících na tomto typu školy	Počet respondentů vyučujících programování na tomto typu školy	Počet respondentů vyučujících programování na tomto typu školy vyjádřený v procentech
ZŠ	61	20	32,8
Gymnáziu	38	31	81,6
Střední odborné škole technického zaměření	76	54	71,1
Střední škole netechnického zaměření	60	17	28,3
Vyšší odborná škola technického zaměření	6	4	66,7
Vyšší odborná škola netechnického zaměření	3	1	33,3
VŠ	5	5	100,0
Součet:	249	132	

Nesoulad mezi počtem respondentů působících na škole a celkovým počtem zúčastněných respondentů je způsoben tím, že někteří učitelé působí na více školách než na jedné. Pokud takový učitel odpověděl, že vyučuje programování, započítala se tato odpověď jako kladná ku všem typům škol, na kterých vyučuje.

Procentuálně nejvíce programování vyučují učitelé vysokých škol, na druhém místě se umístila gymnázia a na třetím střední školy s technickým zaměřením, na místě posledním se umístily základní školy. K problematice výuky programování se několik respondentů vyjádřilo v komentářích na konci dotazníku, v kapitole 3.4 na straně 72.
Ve třech z pěti citací je voláno po větší časové dotaci pro výuku ICT. Pedagogům také chybí nedostatečné vybavení učeben. Případně mají obavy, že by výuka pro děti byla náročná.

4) Škola, na které vyučujete se nachází v kraji:

Tabulka 12 - Počet respondentů vyučujících programování v závislosti na daném kraji

Kraj:	Počet odpovědí:	Počet respondentů vyučujících programování v daném kraji:	Počet respondentů vyučujících programování v daném kraji vyjádřený v procentech
Hlavním městě Praha	20	11	55,0
Jihočeském	11	7	63,6
Jihomoravském	17	7	41,2
Karlovarském	14	6	42,9
Královéhradeckém	36	18	50,0
Libereckém	26	12	46,2
Moravskoslezském	17	10	58,8
Olomouckém	28	16	57,1
Pardubickém	15	7	46,7
Plzeňském	4	3	75,0
Středočeském	7	3	42,9
Ústeckém	11	4	36,4
Vysočina	7	2	28,6
Zlínském	13	7	53,8

Největší zastoupení v tomto dotazníku mají učitelé z Královéhradeckého kraje, druhé největší učitelé z Olomouckého kraje. Nejmenší zastoupení mají učitelé z kraje Vysočina. Programování se procentuálně nejvíce vyučuje v kraji Plzeňském, Jihočeském a Moravskoslezském, nejméně pak v kraji Vysočina a v kraji Ústeckém.

5) V jak velké obci co do počtu obyvatel se nachází škola, ve které vyučujete?

Tabulka 13 - Počet respondentů vyučujících programování v závislosti na velikosti obce, ve které se nachází škola

Velikost obce vyjádřená v počtu obyvatel:	Počet odpovědí:	Počet respondentů vyučujících programování v dané velikosti obce:	Počet respondentů vyučujících programování v závislosti na velikosti obce vyjádřený v procentech
Do 4999	38	8	21,1
5000 - 49 999	96	50	52,1
50 000 +	85	51	60,0

V této tabulce je vidět tendence vyučovat programování spíš ve větších obcích. Tento fakt ale může být ovlivněn tím, že v menších obcích nejsou gymnázia a střední školy, na kterých se programování většinou vyučuje (viz Tabulka 11).

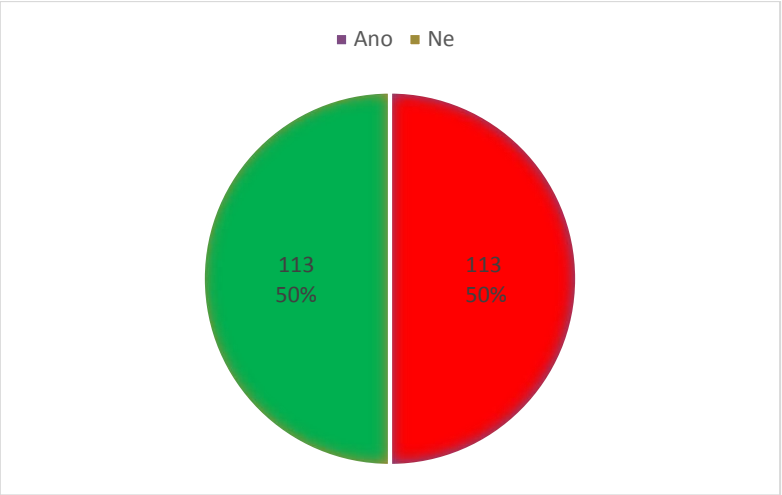
6) Během svého studia jsem absolvoval základy programování na VŠ.

Tabulka 14 - Počet respondentů vyučujících programování v závislosti na jejich předchozí přípravě během studia

	Počet odpovědí:	**Ve svých hodinách vyučují programování:**	**Počet respondentů vyučujících programování v závislosti na jejich předchozí přípravě vyjádřený v procentech:**
Respondenti, kteří absolvovali přípravu	167	103	61,7
Respondenti, kteří neabsolvovali přípravu	59	10	16,9

Z této tabulky lze vysledovat velký vliv přípravy učitelů během jejich studií na to, zda respondenti programování ve svých hodinách programování vyučují či nikoliv. Ze 167 pedagogů, kteří absolvovali přípravu během svého studiu, celých 103 vyučuje programování ve svých hodinách. Zatímco učitelé, kteří nebyli připravováni na programování během svých studií v drtivé většině programování neučí ani ve svých hodinách.

7) Ve svých hodinách vyučuji programování:



Graf 3 - Počet respondentů vyučujících programování ve svých hodinách

Polovina respondentů ve svých hodinách programování vyučuje a druhá polovina nevyučuje. Dotazník se následně věnoval každé z těchto skupin zvlášť, dle klíče uvedeného v kapitole 3 na Obrázku 17.

8) Kolik pedagogů vyučuje programování na Vaší škole?

Tabulka 15 - Počet pedagogů vyučujících programování na jedné škole

Počet pedagogů vyučujících programování na respondentově škole:	**Počet:**
Žádný	79
Jenom já	43
2	32
3	29
4 a více	27

Pedagogové měli možnost odpovědět i do kolonky „Jiná možnost“, z těchto odpovědí jsem počty přičetl k těm, kteří využili již předpřipravených možností.

3.2 Vyhodnocení části dotazníku: „Ve svých hodinách vyučuji programování“

Ve svých hodinách vyučuje programování 113 respondentů, tedy přesně 50 % z celkového počtu vyplněných dotazníku. Tato kapitola se dále děli do podkapitol, ve kterých je uváděno, zda pedagogové ve svých hodinách vyučují programování, jehož výstupem je mobilní aplikace a jsou zde popsány důvody, proč tento typ programování realizují, případně nikoliv. Celková logika dotazníku je uvedena na Obrázku 17 v kapitole 3.

1) Programování učím v rámci:

Tabulka 16 - Zjištění, v rámci jakých hodin je výuka programování realizována

Typ hodiny:	Počet odpovědí:
Hodin informatiky	76
Hodin programování	40
Zájmového kroužku	29
Jiné předměty	2

Nejvíce respondentů realizuje výuku v standardních hodinách informatiky. Na druhém místě se umístila výuka v rámci programování, 29 pedagogů vyučuje programování v rámci zájmového kroužku. V rámci jiných předmětů jako je robotika či elektronické počítače.

2) V kolika třídách učíte programování?

Tabulka 17 - Zjištění, v kolika třídách pedagogové vyučují programování

Počet tříd, ve kterých pedagog vyučuje programování:	Počet odpovědí:
1 – 3	80
4 – 7	31
7+	2

Z tabulky je vidět, že většina respondentů vyučuje v 1 – 3 třídách, zhruba třetina z nich ale vyučuje ve 4 – 7 třídách.

3) Jaká je týdenní průměrná hodinová dotace pro výuku programování na třídu?

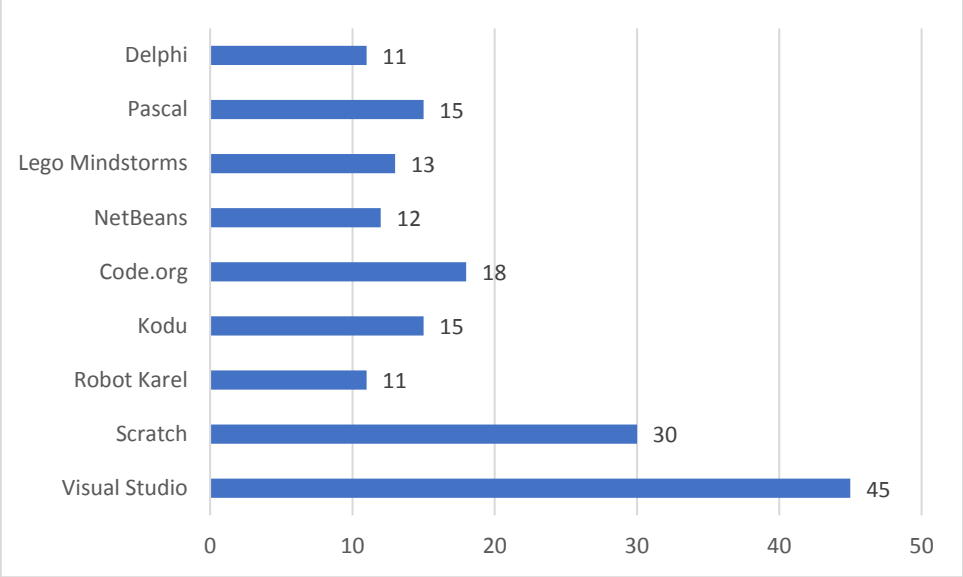
Tabulka 18 - Zjištění časové dotace pro programování na třídu

Průměr hodinové dotace na třídu:	Počet odpovědí:
Méně než jedna hodina	31
1 - 2 hodiny	61
3 - 6 hodin	15
6 +	4

Zhruba jedna třetina respondentů odpovídá, že se programování ve svých hodinách věnují méně než jednu hodinu. Z osobní zkušenosti vím, že tato časová dotace je pro kvalitní výuku programování hraniční, optimální by byly alespoň dvě až čtyři hodiny programování týdně. Na hranici dvou hodin se dostává 54 % pedagogů a možnost vyučovat v jedné třídě mezi 3 a 6 hodinami má pouze 14 % vyučujících. Pokud projde návrh na změnu RVP uváděný v kapitole 1.2 a mezi získané schopnosti žáků druhých stupňů základních škol bude patřit dovednost napsat program v blokově

orientovaném jazyce, bude dle mé zkušenosti potřeba navýšit časovou dotaci pro ICT minimálně o dvě hodiny.

4) V jakých vývojových prostředích (IDE) vyučujete?



Graf 4 - Nejpoužívanější IDE

V tomto grafu jsou uvedeny pouze IDE, která byla zmíněna více než desetkrát. Nejpoužívanějším IDE u respondentů je Visual Studio, kterému je věnována v této diplomové práci kapitola 2.3 – 2.3.4. Na místě druhém místě je IDE s možností vizuálního programování Scratch [18] a na třetím Code.org, jazyk určený pro výuku [103]. Následuje Pascal a Kodu, ve kterém lze opět využít vizuální programování [104]. Dále respondenti používají Lego Mindstorms – prostředí pro programování Lego robotů. Pedagogové používají pro výuku i NetBeans, což je prostředí pro vývoj primárně v Javě, ale také v PHP, HTML5/CSS, JavaScriptu, C/C++ a další [105]. Na místě posledním se umístila IDE Delphi a Robot Karel, což je vizuální programovací IDE dostupné v prohlížeči [16]. Z uvedeného výčtu lze k vizuálním programovacím jazykům můžeme řadit Scratch, Kodu, pro výuku jsou také určeny Code.org, Robot Karel, Lego Mindstorms [106] či Pascal [107]. Tedy celých sedm z výše uvedených devíti IDE či jazyků je určeno pro vzdělávání.

5) Jak důležitá jsou pro Vás následující kritéria při výběru IDE? 1 značí nejméně, 5 nejvíce

Tabulka 19 - Hodnocení důležitosti kritéria při výběru IDE

Hodnocené kritérium:	Počet odpovědí „1“:	Počet odpovědí „2“:	Počet odpovědí „3“:	Počet odpovědí „4“:	Počet odpovědí „5“:	Koeficient*
Čas, který strávím instalací, nastavováním a následnými aktualizacemi	32	20	28	18	11	283
Srozumitelnost prostředí pro žáky	6	1	10	18	74	480
Nařízení vedení školy	61	13	23	2	10	214
Programovací jazyk, který mohu v IDE učit	8	8	29	22	42	409
Podpora k IDE (knihy, komunitní weby)	7	13	40	27	22	371
Jsem s tímto IDE už obeznámen.	12	3	41	26	27	380

* Koeficient byl počítán z důvodu jednoznačného zjištění, které kritérium je pro pedagogy nejdůležitější. Postup výpočtu: Počet odpovědí * známka = koeficient.

Za nejdůležitější považují respondenti kritérium „Srozumitelnost prostředí pro žáky“. Za druhý nejdůležitější považují „Programovací jazyk, který mohu v IDE učit“. Velmi podobně umístili kritéria „Jsem s tímto IDE už obeznámen“ a „Podpora k IDE“. Následuje „Čas, který strávím instalací, nastavováním a následnými aktualizacemi“ a za nejméně důležité považují pedagogové kritérium „Nařízení školy“.

Z prvních dvou kritérií lze vyčíst orientaci na žáka, což považuji za velice pozitivní zjištění.

6) Jaké reakce převládaly u žáků po prvním půl roce výuky programování?

Tabulka 20 - Reakce žáků po půl roce výuky programování

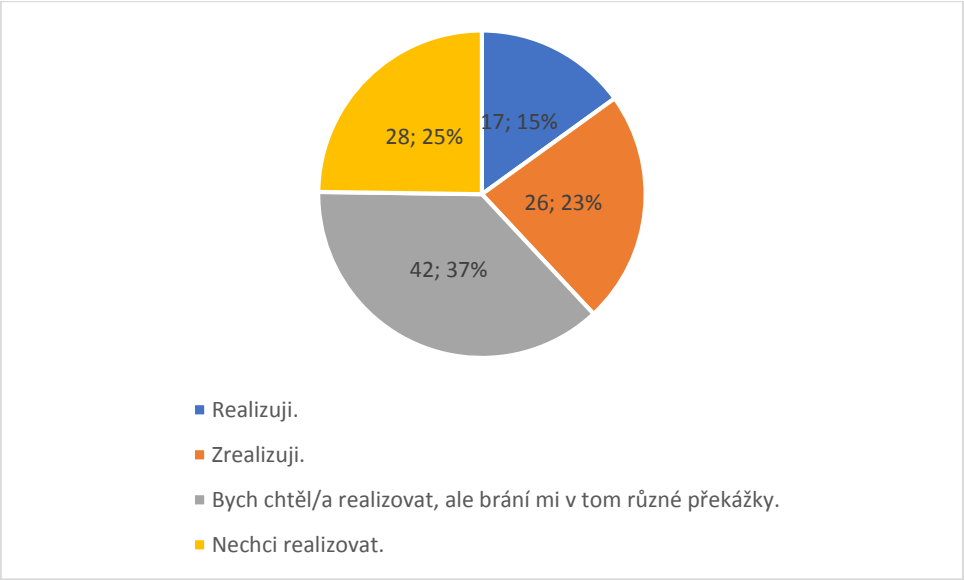
Typ reakce:	Počet odpovědí:
Zájem	89
Odpor	13
Nezájem	31

Po půl roce výuky uvádějí respondenti pozitivní reakce u 67 % žáků, vyložený odpor k programování projevilo pouhých asi 10 % z nich. V komentářích k této otázce se objevovaly také reakce jako pohoda a také to, že nelze odpověď na tuto otázku generalizovat pro celou třídu, neboť každý žák má svůj individuální projev.

7) Výuku programování, jehož výstupem je mobilní či dotyková aplikace:
Respondenti v této otázce mohli zaškrtnout jednu ze čtyř následujících odpovědí:

a) Realizuji.
b) Zrealizuji.
c) Bych chtěl/a realizovat, ale brání mi v tom různé překážky. (Pokud jste zaškrtli tuto možnost, prosím, zdůvodněte ji v následující otázce, která se Vám objeví po kliknutí na "Další")
d) Nechci realizovat, protože: (Pokud jste zaškrtli tuto možnost, prosím, zdůvodněte ji v následující otázce, která se Vám objeví po kliknutí na "Další")

Výsledky odpovědí na tuto otázku jsou zobrazeny v následujícím grafu:



Graf 5 - Zjištění postoje respondentů k výuce programování, jehož výstupem je mobilní či dotyková aplikace

Nejvíce respondentů uvádí, že by rádi tento typ programování realizovali, ale že jim v tom brání různé překážky. Čtvrtina respondentů tento typ programování učit vůbec nechce, pětina se v budoucnu chystá zrealizovat výuku, jejímž výstupem bude mobilní doteková aplikace a 15 % již tento typ programování vyučuje. Pro doplnění je přiložena následující tabulka číslo X, který zachycuje výše uvedené odpovědi ve vztahu ke školám, na kterých respondenti působí. Nesoulad mezi součty v jednotlivých kategoriích je dán tím, že někteří respondenti působí na více školách.

Tabulka 21 - Postoj respondentů k výuce programování, jehož výstupem by byla mobilní aplikace v závislosti na typu školy

Typ školy:	Tento typ programování realizuji:	Tento typ programování zrealizuji:	Tento typ programování bych chtěl zrealizovat:	Tento typ programování nechci realizovat:
ZŠ	1	6	8	5
Gymnáziu	3	10	13	5
Střední odborné škole technického zaměření	13	10	19	12
Střední škole netechnického zaměření	1	5	5	6
Vyšší odborná škola technického zaměření	0	0	3	1
Vyšší odborná škola netechnického zaměření	0	0	0	1
VŠ	0	2	2	1

Z tabulky jasně vyplývá dominance výuky tohoto typu programování na středních odborných školách technického zaměření. V plánu zrealizovat tento typ programování mají i další učitelé na středních odporných školách a také pedagogové na gymnáziích a základních školách.

Po této otázce se dotazník dělil opět do čtyř částí podle zaškrtnuté odpovědi, proto jsou motivace, proč respondenti odpovídali právě takto, popisovány v následujících kapitolách.

3.2.1 Vyhodnocení části dotazníku „Výuku programování, jehož výstupem je mobilní či dotyková aplikace realizuji“

Tento typ programování realizuje 17 respondentů, viz Graf 5.

1) Programování, jehož výstupem je aplikace pro mobilní OS realizuji protože:

Tabulka 22 - Důvody výuky programování, jehož výstupem je mobilní aplikace

Důvod:	Počet odpovědí:
Je pro děti zábavnější než programování, jehož výstupem je počítačová či jiná aplikace	11
Je do budoucnosti více perspektivnější než počítačové, či jiné aplikace	7
Jej máme ve školních osnovách	2
Mě baví	6

65 % respondentů, kteří ve svých hodinách vyučují programování, jehož výstupem je doteková aplikace, si myslí, že tento typ výuky programování je zábavnější než programování s rozdílnou výstupní aplikací a 41 % uvádí, že je do budoucnosti více perspektivnější.

2) Vyberte prosím třídu, ve které jste učili tento typ programování nejdéle. Tuto třídu jste vedl/a:

Jelikož tato otázka úzce souvisí s následující, jsou tyto otázky vyhodnoceny společně pod otázkou číslo 3)

3) Kolik času jste celkem věnovali tomuto typu programování v rámci této třídy?

Tabulka 23 - Délka výuky programování, jehož výstupem je mobilní aplikace

Délka vedení třídy	Délka výuky tohoto typu programování	Počet odpovědí
Rok	Asi měsíc	2
Rok	2 – 5 měsíců	2
2 roky	2 – 5 měsíců	2
2 roky	asi měsíc	2
2 roky	rok a více	1
3 roky	2 – 5 měsíců	1
3 roky	6 – 11 měsíců	2
3 roky	rok a více	2
4 roky	rok a více	2

Více než 5 měsíců se tomuto typu programování věnovalo přes 40 % pedagogů a více než rok se svými třídami programovalo mobilní aplikace zhruba 30 % z nich.

3.2.2 Vyhodnocení části dotazníku „Výuku programování, jehož výstupem je mobilní či dotyková aplikace zrealizuji“

Na tuto otázku odpovědělo 26 respondentů (Viz. kapitola 3.2, otázka číslo 7, Graf 5) a to respondenti vyučující programování, ale také respondenti, kteří programování do výuky teprve zařadit plánují. (Více o struktuře dotazníku v kapitole č. 3 na obrázku číslo 17).

1) Programování, jehož výstupem je aplikace pro mobilní OS zrealizuji protože:

Graf 6 - Důvody, proč respondenti chtějí realizovat výuku programování, jehož výstupem je mobilní aplikace.

45 % pedagogů uvádí jako důvod perspektivu tohoto programování, což naprosto odpovídá realitě – OS Android je nejpoužívanějším OS na světě s 39 % tržním podílem, na třetím místě je s 13 % iOS [26]. 41 % respondentů považuje tento typ programování za pro děti zábavnější, naopak malým důvodem pro učitele je to, že je tento typ výuky baví a za nejslabší důvod pro výuku programování pro mobilní telefony považují její zařazení do školních osnov.

3.2.3 Vyhodnocení části dotazníku „Výuku programování, jehož výstupem je mobilní či dotyková aplikace, bych chtěl/a realizovat, ale brání mi v tom různé překážky.“

Na tuto otázku odpovídalo 42 respondentů, viz. Graf 5 v kapitole 3.2, otázce sedmé.

1) Výuku bych chtěl/a realizovat, ale brání mi v tom různé překážky, jaké?

Tabulka 24 - Jaké překážky brání pedagogům vyučovat programování, jehož výstupem by byla mobilní aplikace

Překážka:	Počet odpovědí:	Vyjádření četnosti odpovědi v procentech:
Nevím, jak na to.	8	19,0
Mám málo času na přeučení se.	13	31,0
Ve škole nemám k dispozici potřebný hardware pro výuku.	20	47,6
Nemám k dispozici potřebnou technickou podporu na škole	0	0,0
Podpora programování není v RVP tolik zakotvena.	14	33,3
Bylo by potřeba změnit ŠVP.	9	21,4
Hrozí možné problémy s vedením školy.	1	2,4
Hrozí možné problémy s rodiči.	1	2,4
Hrozí možné problémy s žáky.	4	9,5
Na školních PC mám problémy s různými omezeními, jako nemožnost instalace, kolize s ostatním softwarem jako jsou antiviry, firewall a další.	0	0

Největší překážkou, která pedagogům brání realizovat výuku programování, jehož výstupem by byla mobilní aplikace, je nedostatečné hardwarové vybavení, kterým disponují ve svých třídách. Každý třetí respondent uvádí jako problém to, že není podpora pro tento typ programování v RVP dostatečně zakotvena a pro téměř každého třetího respondenta je problémem nedostatek času na přeučení se. Mezi dalšími uváděnými problémy se vyskytuje nutnost změnit ŠVP (21, 4%) a také to, že pedagog neví, jak takovou výuku zrealizovat.

V textových komentářích se objevila také obava ze zklamání žáků, kdyby naprogramovaná aplikace na telefonech žákům nefungovala:
„Velké množství různých systémů a případná nekompatibilita s vytvořenou aplikací = zklamání žáků z výsledku“
Mezi dalším uváděným problémem byl zákaz používání mobilních telefonů žáky ve škole.

3.2.4 Vyhodnocení části dotazníku „Výuku programování, jehož výstupem je mobilní či dotyková aplikace nechci realizovat“

Tuto otázku zodpovědělo 22 pedagogů, kteří ve svých hodinách programování vyučují a 27 učitelů, kteří programování zatím nevyučují, ale pokud by byly odstraněny překážky bránící výuce programování, zařadili by jej do výuky, ovšem bez programování, jehož výstupem by byla mobilní aplikace.

1) Tuto výuku nechci realizovat, protože:

Tabulka 25 - Důvody, proč respondenti nechtějí vyučovat programování, jehož výstupem by byla mobilní aplikace

Uváděný důvod:	Počet odpovědí:	Vyjádření četnosti odpovědi v procentech:
Nevím, jak na to.	9	18,4
Mám málo času na přeučení se.	10	20,4
Ve škole nemám k dispozici potřebný hardware pro výuku.	17	34,7
Nemám k dispozici potřebnou technickou podporu na škole	11	22,4
Podpora programování není v RVP tolik zakotvena.	9	18,4
Bylo by potřeba změnit ŠVP.	0	0,0
Hrozí možné problémy s vedením školy.	0	0,0
Hrozí možné problémy s rodiči.	0	0,0
Hrozí možné problémy s žáky.	2	4,1
Na školních PC mám problémy s různými omezeními, jako nemožnost instalace, kolize s ostatním softwarem jako jsou antiviry, firewall a další.	7	14,3
Není pro děti zábavnější než programování, jehož výstupem je počítačová či jiná aplikace.	3	6,1
Není do budoucnosti více perspektivnější než počítačové, či jiné aplikace.	2	4,1
Jej nemáme ve školních osnovách.	11	22,4
Mě nebaví.	2	4,1
Mi to přijde zbytečné.	18	36,7

Největšímu procentu respondentů přijde tento typ výuky programování jako zbytečný, jako druhý nejzásadnější důvod je uváděno nedostatečné hardwarové vybavení učeben. V textových komentářích se vícekrát

objevil důvod „Nedostatečná časová dotace“, případně nedůvěra ve schopnosti žáků si toto programování osvojit.

3.3 Vyhodnocení části dotazníku: „Ve svých hodinách nevyučuji programování“

Ve svých hodinách nevyučuje programování 113 pedagogů, tedy přesně 50 % všech zúčastněných respondentů.

1) Jaké jdou důvody toho, že nevyučujete programování:

Tabulka 26 - Důvody, proč respondenti ve svých hodinách nevyučují programování

Uváděný důvod:	Počet odpovědí:	Vyjádření četnosti odpovědi v procentech:
Nevím jak na to.	11	9,7
Mám málo času na přeučení se.	9	8,0
Ve škole nemám k dispozici potřebný hardware pro výuku.	6	5,3
Nemám k dispozici potřebnou technickou podporu na škole	8	7,1
Podpora programování není v RVP tolik zakotvena.	30	26,5
Hrozí možné problémy s vedením školy.	1	0,9
Hrozí možné problémy s rodiči.	1	0,9
Hrozí možné problémy s žáky.	9	8,0
Na školních PC mám problémy s různými omezeními, jako nemožnost instalace, kolize s ostatním softwarem jako jsou antiviry, firewall a další.	9	8,0
Myslím si, že to není potřeba.	27	23,9
Výuka programování není zakotvena v ŠVP.	65	57,5

Téměř 60 % respondentů nevyučuje programování kvůli jeho nezařazení do ŠVP, druhý nejčastěji uváděný důvod úzce souvisí s prvním, téměř 30 % pedagogů uvádí, že neučí programování kvůli dostatečné opoře v RVP. Třetím nejzásadnější důvodem, proč se programování nevyučuje, je to, že si pedagogové myslí, že to není potřeba.

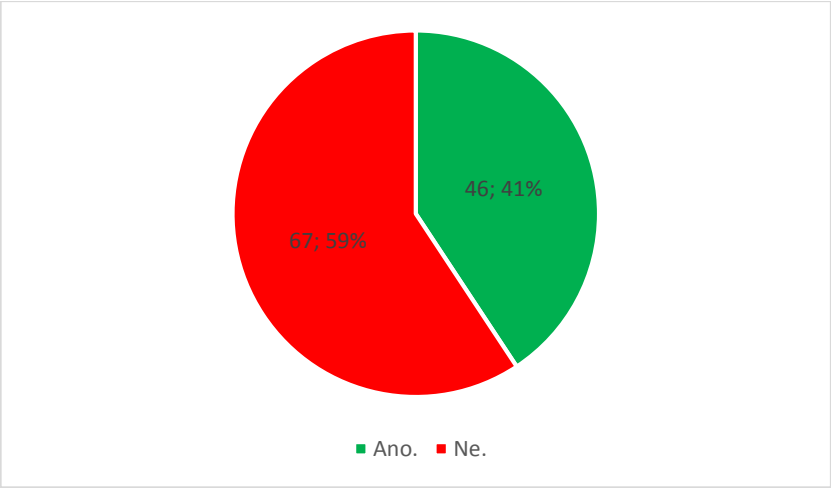
Mezi komentáři k této otázce se vyskytly i důvody takové, že programování na škole učí někdo jiný, případně že není programování na dané škole s daným zaměřením potřeba. Vícekrát se objevovaly komentáře vysvětlující nemožnost učit programování při současné hodinové dotaci pro informatiku, jak dokládají například tyto komentáře:

„Jak učit programování při naplnění RVP a dotaci 1h/týdně 5.třídy a 1h/týdně 6.třídy???“

„Se současnou dotací hodin IKT je často problém úplně zvládnout OFFICE + HW + grafiku“

„Nemám vhodný SW ani časovou dotaci“

2) Pokud by byly odstraněny překážky, které uvádíte výše, zavedl/a bych výuku programování.



Graf 7 - Zjištění ochoty respondentů zavést výuku programování po případném odstranění překážek bránících jeho výuce

Výsledek je poměrně překvapivý, téměř 70 % respondentů by programování ve i po odstranění překážek bránící jeho výuce ve svých hodinách nezavedlo.

3) Pokud jste s předchozí otázkou souhlasil/a, zařadil byste do výuky programování, jehož výstupem by byla aplikace pro mobilní platformy?
Odpověď na tuto otázku nebyla povinná. Výsledky je možné shlédnout v kapitolách 3.2.2 („Výuku programování, jehož výstupem je mobilní či dotyková aplikace zrealizuji“), potažmo v kapitole 3.2.4 („Výuku programování, jehož výstupem je mobilní či dotyková aplikace nechci realizovat“).

3.4 Vyhodnocení dotazníku – společná závěrečná část

1) Pokud se chcete vyjádřit k tomuto tématu, dotazníku, či sdělit své zkušenosti, můžete tak učinit zde.

V následujícím tabulce jsou vloženy vybrané komentáře:

Tabulka 27 - Vybrané komentáře k dotazníkovému šetření

„Na ZŠ nejsou děti připraveny na programování. Jsme rádi, že je naučíme základy práce s MS Office a ani v tom nejsou nijak zdatné. Navíc je informatice věnován neskutečně malý prostor v RVP (1 h na 1. stupni, 1 h na 2. stupni).“
škola pouze s 1. stupněm
„Základy programování k výuce programování nestačí, je potřeba vzdělávání a podpora pedagogů v této oblasti. “
„Studenti, kteří dnes přicházejí ze základních škol neumí většinou dělit ani dvěma bez kalkulačky. Pokud se nezmění systém výuky na základních školách, nemá smysl řešit nějaké programování. Po VŠ jsem se studenty pár let programovala, pak jsem to už brala jako ztrátu času. “
„Bylo by dobré, kdyby vznikla nějaká učebnice, jako máme na Python ze Slovenska i pro programování aplikací pro mobilní telefony, tablety apod.“
„Programování na základní škole by pro řadu žáků bylo dost složité (někteří neumí ani malou násobilku nebo napsat větu bez chyby). Raději dáváme přednost prezentacím, hledáním informací na internetu nebo malování v různých programech.“
„Bylo by potřeba více hodin, více peněz na vybavení učeben a počítačů programy. ICT by se mělo stát hlavním předmětem jako je například zeměpis či přírodopis.“
Nejsme technická škola, žáci jsou slabí na matematiku, programování je pro ně obtížné a vadí jim
„Osobně se domnívám, že problematika programování přesahuje rozsah, který je IT ve vzdělávacím programu přidělen a přesahuje ve většině případů nejen potřeby, ale i možnosti žáků, které se na problematiku výpočetní techniky nespecializují.“
„Myslím, že by se s kódováním v nějaké formě mělo povinně začínat už na 1. stupni. V kroužku LEGO robotiky používáme jen LEGO Mindstorms, rád bych ale LEGO programoval také textovým kódem a začal pracovat s Arduinem. Problém je čas a zaučení. Díky kroužku se začalo několik lidí samostatně učit Python skrze účast v KSP-z a KSP.“

„Na některé otázky bylo nemožné odpovědět. Programování učíme v několika stupních - u malých se jedná zhruba o dva měsíce v sedmém a osmém ročníku. Jedná se o blokové programování, které má v dětech prostě jen vzbudit zájem. V prvním ročníků vytvářejí weby a zhruba tři měsíce tráví výukou JavaScriptu, opět pouze na bazální úrovni. Pro třetí a čtvrtý ročník gymnázia pak existuje pro zájemce seminář (dvouhodinová dotace každý rok), kde se učí Python, JS, PHP a MySQL.“
„Dělám to 30 let a pořád mě to baví, soudím, že studenty taky. (ale jde o volitelný předmět) Těžko říct, zda je výhodou či nevýhodou, že je třeba stále studovat nové věci, jazyky, prostředí, pojetí. (zatím od strukturovaného Algolu a Fortranu přes Pascal a Delphi k objektovému C#)“
„Programujeme v C,C++ z toho je transfer do PHP a wiringu, dále pak systémového programování. Pro pochopení základů je tady vhodné pochopit základy programování a podstatu překladu.... IDE je na obtíž; upustil jsem od něj a žáci se rychleji dostali dál, protože se nemuseli učit ovládání IDE. S tím se setkají až u Arduina.“
„Časová dotace na INF a materiální podmínky jsou takové, že opravdu není čas na programování...“
„Psát smysluplné aplikace s žáky, to potřebuje ze strany učitele obrovskou a náročnou přípravu a od žáka i vlastní domácí přípravu a velký zájem s motivací, což se na střední škole málokdy povede, ale je to možné. […]“
„Na střední škole jsme měli výuku programování - učila nás paní učitelka, která tomu sama nerozuměla, měli jsme jen stres, předmět byl velmi neoblíbený. V jiné skupině vyučoval programování učitel, který programování studoval na VŠ, žáky vzorně vedl, vše potřebné jim uměl vysvětlit a ve třídě bylo při výuce optimální klima. Koncem 80-tých let jsme se museli učit pouze programovací jazyk BASIC.“

V komentářích se objevují stížnosti na malou časovou dotaci pro ICT, slabé vybavení učeben hardwarem, případně obavy, aby žáci programování zvládli.

2) Pokud si přejete získat výsledky mé diplomové práce, vepište prosím svůj e-mail:

   Svůj e-mail vepsalo 46 respondentů.

3.5 Celkové zhodnocení dotazníku

Cílem tohoto dotazníku bylo přinést vhled do současné situace ve výuce programování. Tuto výuku realizovalo přesně 50 % respondentů, nejčastěji na vysokých školách, gymnáziích a středních odborných školách technického zaměření. Nejméně častá tato výuka byla na středních školách netechnického zaměření a na školách základních. Pedagogové, kteří ve svých hodinách programování nevyučuji, uvedli i důvody, proč programování ve svých hodinách nevyučují. Mezi nejčastěji zmiňované patřila nedostatečná zakotvenost programování v RVP, potažmo v ŠVP a také to, že si myslí, že tato výuka není potřeba (tabulka číslo X). V komentářích se objevovalo volání po větší časové dotaci pro hodiny informatiky a také se vyskytli obavy z náročnosti této výuky pro žáky. (strana X) Velký vliv na to, zda pedagog ve svých hodinách programování vyučuje má i to, zda během svých studií absolvoval základy programování. Zhruba 60 % ((tabulka X) z těch, kteří přípravu absolvovali, programování vyučuje. Oproti tomu programování vyučuje jen 17 % respondentů, kteří přípravu neabsolvovali.

Tato DP práce se věnuje srovnání IDE pro OS Android, proto v dotazníku bylo zjišťováno, zda je vyučováno i programování, jehož výstupem je mobilní aplikace. Kladně z těch, co vyučují programování opovědělo 15 % respondentů, nejčastěji je tato výuka realizována na středních odborných školách technického zaměření. Zhruba 23 % pedagogů tuto výuku plánuje zrealizovat, nicméně 37 % pedagogů v této výuce brání nějaké překážky, mezi které patří například nedostatečný hardware v učebnách, případně to, že není tento typ výuku v RVP dostatečně zakotven anebo také to, že nemají čas na přeučení se. Zmiňovaný problém nedostatečného hardwarového vybavení je velmi dobře řešitelný, neboť výuka programování, jehož výstupem je mobilní OS neklade při nasazení například App Inventoru větší nároky na hardware (kapitola 2.1.1), než výuka jiného typu programování.

V současné době je vyučování programování či algoritmizace již od prvního stupně diskutováno. (kapitola XX). Pokud by mělo být toto vyučování realizováno, bude dle výsledků dotazníku nutné:

1) Změnit RVP a potažmo ŠVP.

2) Zajistit vzdělání pedagogů v této oblasti společně s osvětou, proč je potřeba programování učit a zajistit přípravu budoucích učitelů na pedagogických univerzitách.

4 Testovací aplikace

V této kapitole jsou popsány zkušenosti získané při psaní jednoduché hry, zdrojové kódy jsou v přílohách.

Po začátku hry se začíná odpočítávat čas a hráčovým úkolem je stisknout pohybující se tlačítko / obrázek, čímž se přičte čas a skóre. Konec hry nastane, když vyprší předem stanovený čas.

Při psaní aplikací v jednotlivých IDE je nutné počítat s různými odchylkami přístupů. Například u App Inventoru je možné ošetřit kliknutí na obrázek, ve Visual Studiu toto není možné a je nutné nasadit tlačítko, které se případně obrázkem vyplní.

4.1 Pracovní list pro MIT App Inventor

Téma	
Tematický celek	Programování jednoduché hry
Počet žáků	5-15
Věk žáků	8+
Pomůcky	Viz kapitola 2.1 (MIT App Inventor 2)
Stručný popis aktivity	Tvorba jednoduché hry v prostředí MIT App Inventor 2
Cíle aktivity	Studenti zvládnou vytvoření jednoduché hry, která může být následujících hodinách rozvíjena.
Předchozí znalosti	Pro použití následujícího pracovního listu zná žák základy programování jako jsou podmínky, smyčky a prostředí Android Studia. Pokud nezná, je třeba žáky s MIT App Inventorem seznámit a základy programování například s pomocí pracovního listu vysvětlit.
Časový plán	Časový odhad uveden u každého bodu, nicméně napsání této aktivity může trvat zhruba 60 minut.
Návaznosti	Hra lze modifikovat navrženými způsoby uvedenými na konci pracovního listu.

Pracovní list:

1) Otevření vývojového prostředí a přihlášení se. (3 minut)

   V prohlížeči si otevřete adresu http://ai2.appinventor.mit.edu a přihlaste se Google účtem.

2) Vytvoření projektu. (1 minuta)

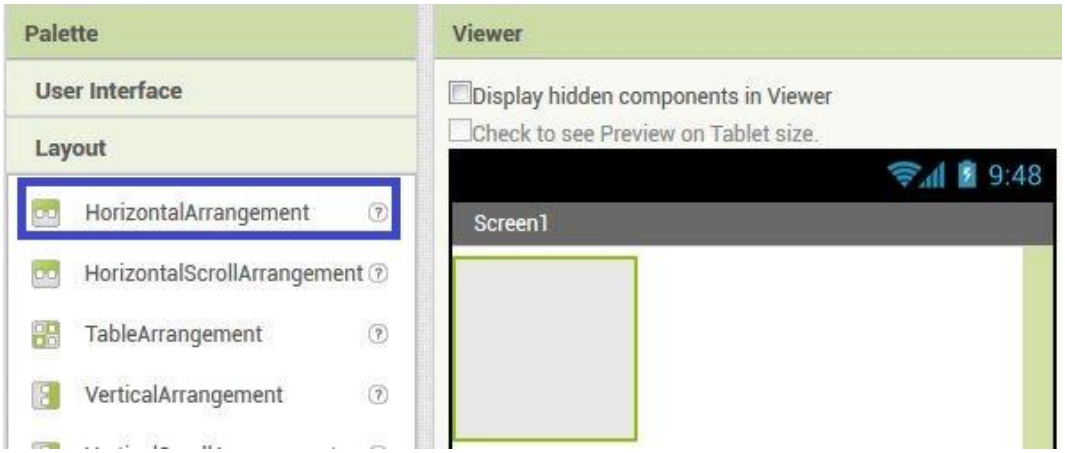
   V prostředí klikněte na „Start new project“ pojmenujte jej například „Game“.

3) Připojení telefonu do USB (Jiné možnosti testování aplikace popisuje kapitola 2.1.1), (2 minuty)

   Připojte telefon do USB s povoleným „USB laděním“, spusťte na PC aplikaci aiStarter.exe a následně klikněte v okně na „Connect“ → „USB“. Nyní se již všechny změny projeví na telefonu.

4) Přidání prvků „HorizontalArrangement“, Button a Label

   Na displej vložte z „Layout“ → „HorizontalArrangement“. (6 minut)



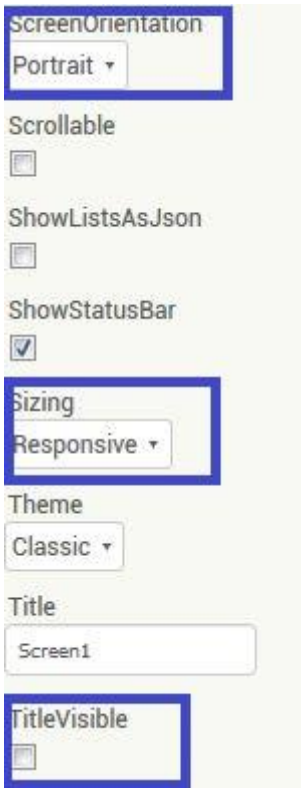
   Do tohoto šedého čtverce přidejte z „User Interface“ dvě tlačítka „Button“ a dva popisky „Label“ tak, jak je na obrázku.

5) Vložení prvku Canvas (3 minuty)

   Z „Drawing“ vložte do bílého místa kdekoliv na obrazovce prvek „Canvas“ a na Canvas vložte „ImageSprite“.

6) Úprava obrazovky (2 minuty)

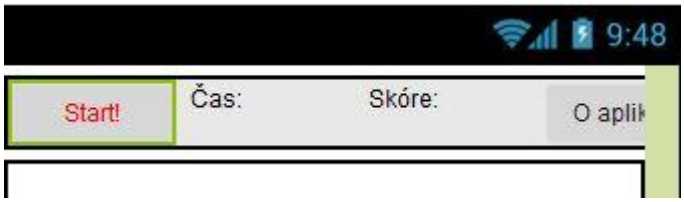
Klikněte v sloupečku „Components“ na „Screen 1“. Následně upravte vlastnosti „ScreenOrientation“ na „Portrait“, „Sizing“ na „Responsive“ a odškrtněte „TitleVisible“.



7) Responzivita prvků. (4 minuty)

Klikněte na „HorizontalArrangement1“ a ve vlastnostech upravte „Width“ na „Fill Parent“. Poté nastavte každému z prvků „Button1“, „Button2“, „Label1“ a „Label2“ „Width“ na 25% a v části „Text“ je popište tak, jak je na obrázku. Tlačítko „Button1“ je možné nastavit i červenou barvu.

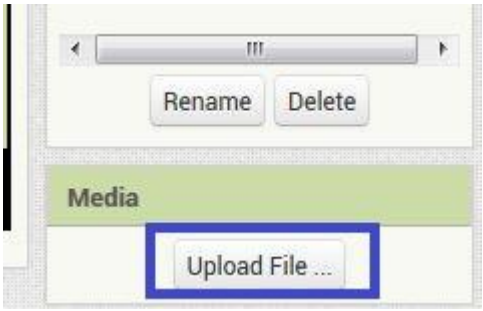
Prvku „Canvas“ nastavte „Height“ a „Width“ na „Fill parent“.



8) Stáhnutí a nahrání obrázku do aplikace. (5 minut)

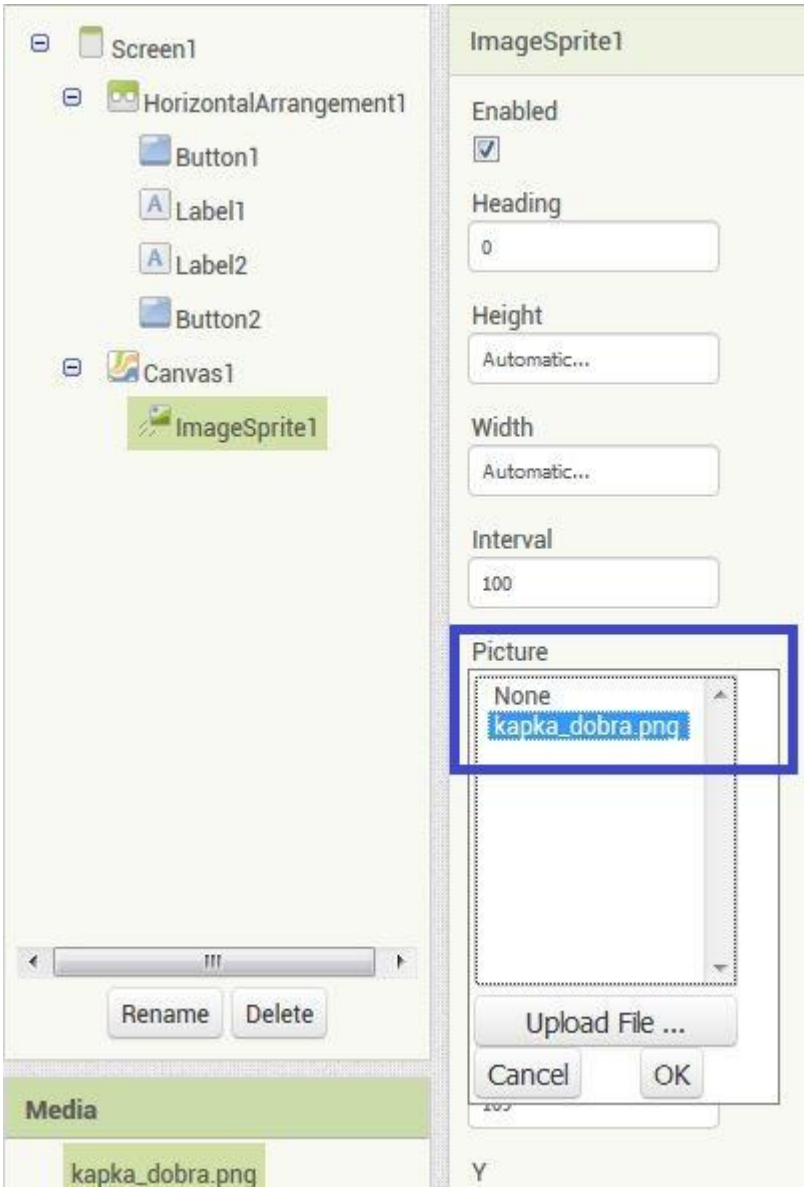
Na serveru flaticon.com zadejte frázi „waterdrop“ a stáhněte si libovolnou v co největším rozlišení ve formátu .png do počítače.

V části „Media“ klikněte na „Upload file“ a nahrajte obrázek.



9) Přidání obrázku do hry. (1 minuta)

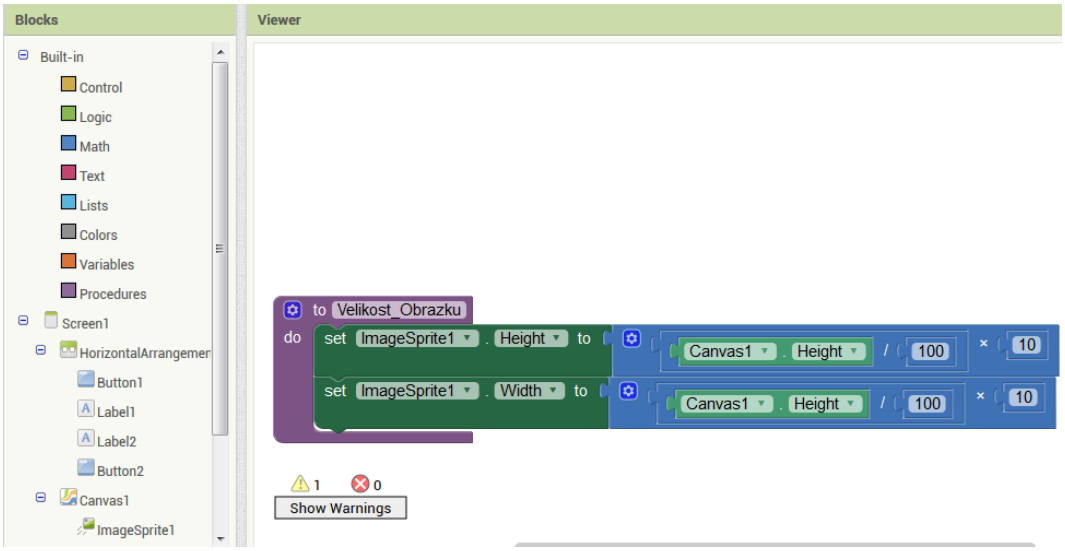
Klikněte na „ImageSprite“ a v jeho vlastnostech klepněte na „Picture“ a vyberte váš nahraný soubor.



10) Nastavení responzivnosti obrázku. (7 minut)

Poznámka: Tato část lze vynechat, pokud budete aplikaci používat jen na své velikosti telefonu. Pak můžete využít Nastavení vlastností „ImagePickeru“ v části „Width“ a „Height“.

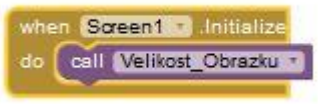
Přepněte se do části „Blocks“, kde vytvoříte proceduru „Velikost_Obrazku“.



Poznámka: Jak je vidět z obrázku, barvy korespondují se složkami, kde je lze najít. Tedy například samotná procedura je fialová, proto ji lze nalézt v části „Procedures“. Část tmavě modrá lze nalézt v sekci „Math“.

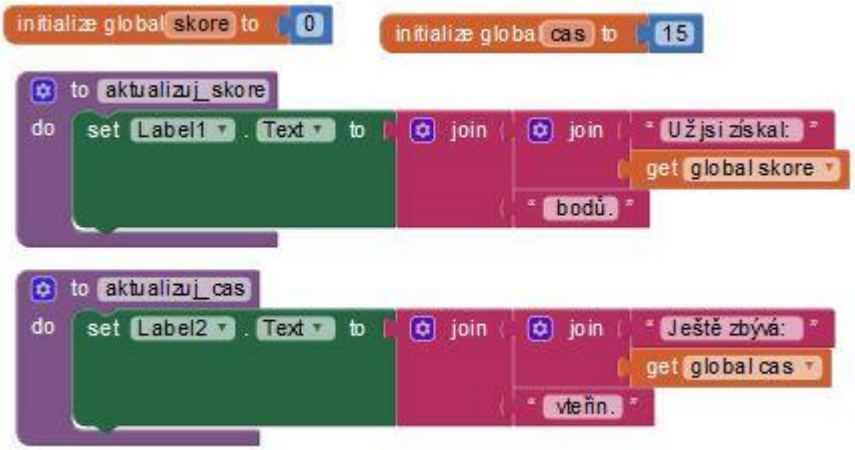
Tato procedura má za úkol nastavit desetiprocentní šířku a výšku obrázku vzhledem k šířce Canvasu. Proto na větším displeji bude obrázek větší, na menším menší. Cílem je zvýšení hratelnosti, neboť při velkém displeji by se těžce klikalo na malý obrázek.

Nyní přidejte ještě následující kus kódu.



Po inicializaci obrazovky by se již obrázek měl zmenšit na 10 % šířky Canvasu.

11) Nastavení ukazatele času a skóre. (7 minut)



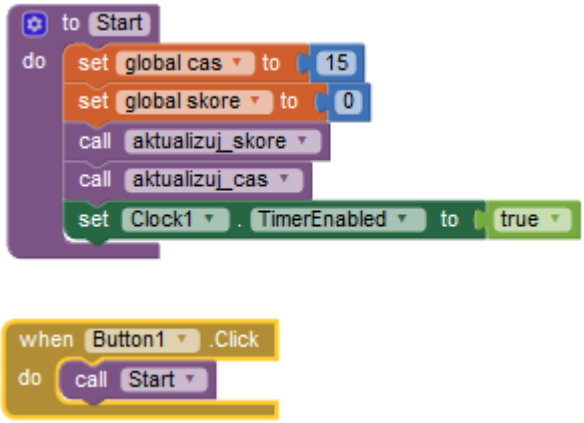
Proměnná čas je tedy nastavená na 15 vteřin, skóre samozřejmě na 0.

12) Přidání dalších komponent. (2 minuty)

V části „Design“ přidejte z „Sensors“ komponentu „Clock“ a nastavte ji „TimerInterval“ na 1000 (číslo 1000 je vyjádření v milisekundách) a odškrtněte „TimerEnabled“.

Dále přidejte dvakrát „Notifier“ z části „User Interface“.

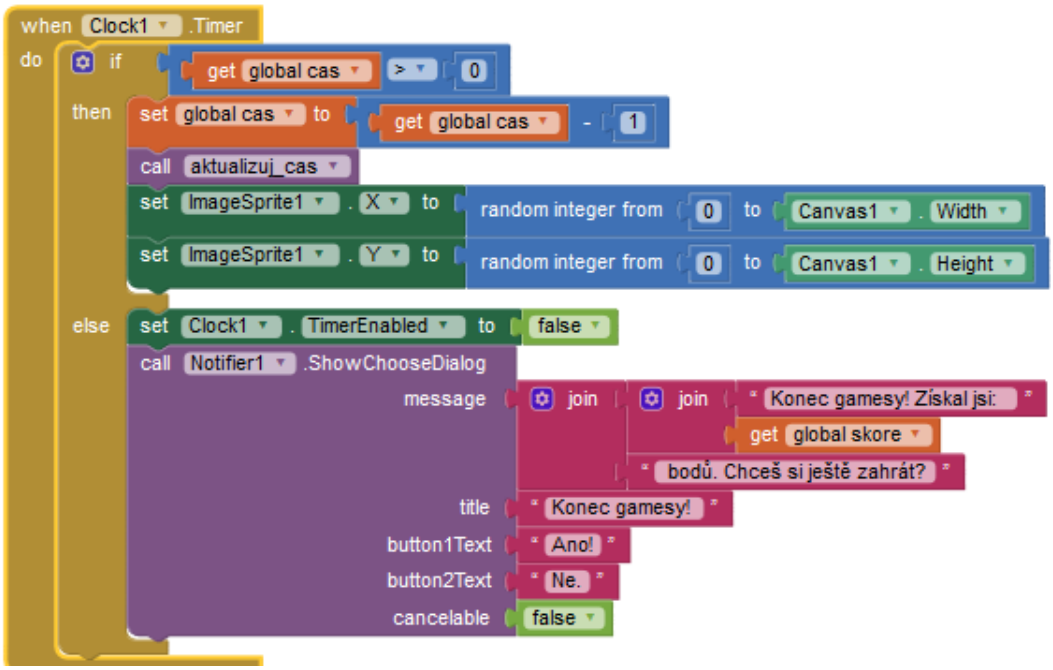
13) Přidání procedury „Start“. (3 minuty)



Tato procedura nastaví proměnné na jejich původní hodnoty a zaktualizuje skóre a zbývající čas. Následně povolí komponentu „Clock“.

Tlačítko „Button1“ tuto proceduru pak volá.

14) Hlavní část kódu (7 minut)

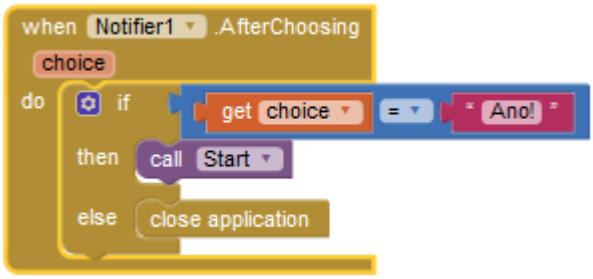


Když je zapnutý „Timer“, je tento kus kódu volán každou vteřinu (viz bod 12) a zjišťuje se podmínka, zda má hráč ještě čas, nebo jestli čas už vypršel.

Pokud má hráč ještě čas, odečte se vteřina od celkového počtu zbývajících vteřin, aktualizuje se čas a nastaví se náhodná pozice (zde jsou generovány čísla od 0 do šířky a výšky Canvasu. Tímto je ošetřena každá velikost displeje).

Pokud hráč už čas nemá, vypne se „Timer“ a vyvolá se „Notifier1“, ve kterém se vypíše skóre očekává se odpověď od uživatele – buď „Ano!“ anebo „Ne.“.

15) Ošetření odpovědi uživatele. (3 minuty)

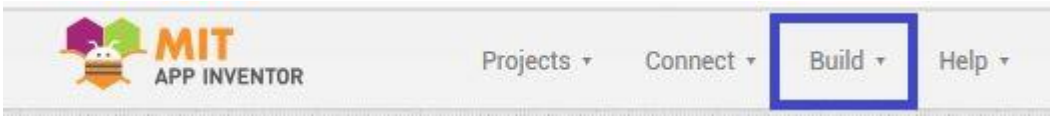


Pokud hráč klikne na „Ano!“, hra se znovu spustí, pokud klikne na „Ne“, hra se ukončí.

16) Ošetření tlačítka Button2 – O aplikaci (5 minut)

Po kliknutí se objeví libovolná hláška. Jelikož ale byl použit obrázek ze serveru flaticon.com, který podléhá citování, je jej zde nutné uvést.

17) Nyní již hotovou aplikaci lze stáhnout v části „Build“.



18) Rozšíření:

Umožnit „Landscape“ režim, aby hra fungovalo dobře i v tomto režimu displeje.

Možnost ukládat nejvyšší skóre do lokálního uložiště.

Možnost ukládat skóre do internetového uložiště, vytvořit tak žebříček nejlepších hráčů.

Přidat „špatnou kapku“ v červené barvě, která by hráči ubrala skóre a čas.

5 Závěr

V práci byl zkoumán současný stav výuky programování v českém školství. Bylo zjištěno, že programování je doménou středních škol, na školách základních je programování poměrně řídkým jevem. Programováním, jehož výstupem je mobilní aplikace, se ve svých hodinách zabývá také poměrně málo učitelů, nicméně velké procento plánuje, nebo by chtělo tuto výuku realizovat.

Při výběru IDE je zřejmě nejdůležitějším kritériiem úroveň žáků. Pokud s programováním začínají, jeví se jako nejlepší varianta IDE MIT App Inventor 2. Pokud jsou žáci pokročilejší, je vhodné využít Visual Studio, ve kterém je možné psát i desktopové, webové a další aplikace, případnou alternativou je Android Studio.

6 Přehled zdrojů:

1. Rámcové vzdělávací programy, Národní ústav pro vzdělávání. Národní ústav pro vzdělávání [online]. Copyright © [cit. 15.02.2018]. Dostupné z: http://www.nuv.cz/t/rvp

2. Školní vzdělávací program – Wikipedie. [online]. [cit. 15.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/%C5%A0koln%C3%AD_vzd%C4%9Bl%C3%A1vac%C3%AD_program

3. Nový projekt E-Bezpečí a společnosti O2 Czech Republic odstartoval ve Zlatých Horách . [online]. [cit. 15.02.2018]. Dostupné z: http://m.zurnal.upol.cz/nc/zprava/clanek/novy-projekt-e-bezpeci-a-spolecnosti-o2-czech-republic-odstartoval-ve-zlatych-horach/

4. Je to tady. Windows 10 je nejpoužívanějším systémem v Česku - Cnews.cz. Cnews.cz | Od tranzistorů až po PC sestavy [online]. [cit. 15.02.2018]. Dostupné z: https://www.cnews.cz/je-tady-windows-10-je-nejpouzivanejsim-systemem-v-cesku/

5. Operating Systems – Gemius Ranking [online]. Copyright © Copyright Gemius 2018 [cit. 15.02.2018]. Dostupné z: http://ranking.gemius.com/cz/ranking/systems/

6. Rámcové vzdělávací programy, Národní ústav pro vzdělávání. Národní ústav pro vzdělávání [online]. Copyright © [cit. 15.02.2018]. Dostupné z: http://www.nuv.cz/t/rvp

7. Rámcový vzdělávací program pro základní vzdělávání. Národní ústav pro vzdělávaní [online]. 2017 [cit. 15.02.2018]. Dostupné z: http://www.nuv.cz/uploads/RVP_ZV_2017_verze_cerven.pdf

8. Rámcový vzdělávací program pro gymnázia. Národní ústav pro vzdělávaní [online]. 2007 [cit. 15.02.2018]. Dostupné z: http://www.nuv.cz/file/159_1_1/

9. Rámcový vzdělávací program pro obor vzdělání 23 – 61 – H/01 Autolakýrník. Národní ústav odborného vzdělávaní [online]. [cit. 15.02.2018]. Dostupné z: http://zpd.nuov.cz/RVP/H/RVP%202361H01%20Autolakyrnik.pdf

10. Rámcový vzdělávací program pro obor vzdělání 82-47-M/01 82-47-P/01 Hudebně dramatické umění. Národní ústav odborného vzdělávaní [online]. [cit. 15.02.2018]. Dostupné z: http://zpd.nuov.cz/RVP_4_vlna/RVP_8247M01_Hudebne_dramaticke_umeni_8247P01_Hudebne_dramaticke_umeni.pdf

11. Rámcový vzdělávací program pro obor vzdělání 18-20-M/01 Informační technologie. Národní ústav odborného vzdělávaní [online]. 2008 [cit. 15.02.2018]. Dostupné z: http://zpd.nuov.cz/RVP/ML/RVP%201820M01%20Informacni%20technologie.pdf

12. Rámcový vzdělávací program pro obor vzdělání 82 – 44 – J/01 Ladění klavírů a kulturní činnost. Národní ústav odborného vzdělávaní [online]. 2009 [cit. 15.02.2018]. Dostupné z: http://zpd.nuov.cz/RVP_3_vlna/RVP%208244J01%20Ladeni%20klaviru%20a%20kulturni%20cinnost.pdf

13. Hypertext Markup Language – Wikipedie. [online]. [cit. 15.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/Hypertext_Markup_Language

14. 14. Programovací jazyky – Stránky k výuce informatiky. Stránky k výuce informatiky – určené nejen pro studenty Gymnázia Vlašim [online]. Copyright © 2018 [cit. 16.02.2018]. Dostupné z: http://www.ivt.mzf.cz/seminar/14-programovaci-jazyky/

15. Značkovací jazyk – Wikipedie. [online]. [cit. 16.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/Zna%C4%8Dkovac%C3%AD_jazyk

16. Robot Karel: vývojové prostředí. Robot Karel: vývojové prostředí [online]. Copyright © 2006 [cit. 16.02.2018]. Dostupné z: http://karel.oldium.net/

17. Logo (programovací jazyk) – Wikipedie. [online]. [cit. 16.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/Logo_(programovac%C3%AD_jazyk)

18. Scratch - Imagine, Program, Share. Scratch - Imagine, Program, Share [online]. [cit. 16.02.2018]. Dostupné z: https://scratch.mit.edu/

19. CodeCombat - Learn how to code by playing a game. CodeCombat - Learn how to code by playing a game [online]. [cit. 16.02.2018]. Dostupné z: https://codecombat.com/

20. STRATEGIE DIGITÁLNÍHO VZDĚLÁVÁNÍ DO ROKU 2020. Ministerstvo školství mládeže a tělovýchovy [online]. 2014 [cit. 16.02.2018]. Dostupné z: http://vzdelavani2020.cz/images_obsah/dokumenty/strategie/digistrategie.pdf

21. Průběžné hodnocení implementace - Strategie digitálního vzdělávání do roku 2020 (rok 2017) – MŠMT ČR [online]. Copyright © [cit. 16.02.2018]. Dostupné z: http://www.msmt.cz/uploads/zpra_va_SDV_vla_da_2017.docx

22. Zápis z kulatého stolu na téma „Informatika, roboti a informatické myšlení ve škole? Představení vize nového obsahu výuky informatiky a informatického myšlení od MŠ po SŠ“. Pedagogicke.info [online]. 2018 [cit. 16.02.2018]. Dostupné z: http://www.pedagogicke.info/2018/02/zapis-zkulateho-stolu-na-tema.html

23. Informatika - rámec očekávaných výstupů (prosinec 2017). Google Docs [online]. 2017 [cit. 16.02.2018]. Dostupné z: https://docs.google.com/spreadsheets/d/1GQCNaCjIJlspk5sTNsU0gV4DYvEETvrvHy6VYzlH674/edit#gid=1456952308

24. LMC. Jobs.cz – Inspirujeme k úspěchu – nabídka práce, volná pracovní místa, brigády i vzdělávání a rozvoj [online]. Copyright © 1996 [cit. 16.02.2018]. Dostupné z: https://www.jobs.cz/prace/programator/

25. FAQ | StatCounter Global Stats. StatCounter Global Stats - Browser, OS, Search Engine including Mobile Usage Share [online]. Copyright © StatCounter 1999 [cit. 16.02.2018]. Dostupné z: http://gs.statcounter.com/faq#methodology

26. Desktop vs Mobile vs Tablet Market Share Worldwide | StatCounter Global Stats. StatCounter Global Stats - Browser, OS, Search Engine including Mobile Usage Share [online]. Copyright © StatCounter 1999 [cit. 17.02.2018]. Dostupné z: http://gs.statcounter.com/platform-market-share/desktop-mobile-tablet/worldwide/#monthly-201702-201802-bar

27. StatCounter Global Stats - Browser, OS, Search Engine including Mobile Usage Share. StatCounter Global Stats - Browser, OS, Search Engine including Mobile Usage Share [online]. Copyright © StatCounter 1999 [cit. 17.02.2018]. Dostupné z: http://gs.statcounter.com/os-market-share#monthly-201702-201802-bar

28. Gartner Says Worldwide PC Shipments Declined 2 Percent in 4Q17 and 2.8 Percent for the Year - Gartner. [online]. [cit. 17.02.2018]. Dostupné z: https://www.gartner.com/newsroom/id/3844572

29. Gartner Says Worldwide Sales of Smartphones Recorded First Ever Decline During the Fourth Quarter of 2017 – Gartner. [online]. [cit. 17.02.2018]. Dostupné z: https://www.gartner.com/newsroom/id/3859963

30. App stores: number of apps in leading app stores 2018 | Statista. • Statista - The Statistics Portal for Market Data, Market Research and Market Studies [online]. Copyright © Statista 2018 [cit. 16.02.2018]. Dostupné z: https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

31. MIT App Inventor - Related Research | Explore MIT App Inventor. [online]. Copyright © 2012 [cit. 16.02.2018]. Dostupné z: http://appinventor.mit.edu/explore/research.html

32. WOLBER, David. App inventor. Brno: Computer Press, 2014. ISBN 9788025141953. –

Building android apps in easy steps: covers app inventor 2. 02. S. l.: In Easy Steps Limited, 2014. ISBN 1840786299.

33. Setting Up App Inventor | Explore MIT App Inventor. [online]. Copyright © 2012 [cit. 16.02.2018]. Dostupné z: http://appinventor.mit.edu/explore/ai2/setup.html

34. How Does my Android Device Connect Over Wifi? | Explore MIT App Inventor. [online]. Copyright © 2012 [cit. 17.02.2018]. Dostupné z: http://appinventor.mit.edu/explore/support/explain-wifi-connection-2.html

35. Connect your Phone or Tablet over WiFi | Explore MIT App Inventor. [online]. Copyright © 2012 [cit. 17.02.2018]. Dostupné z: http://appinventor.mit.edu/explore/ai2/setup-device-wifi.html

36. Connecting to a phone or tablet with a USB cable | Explore MIT App Inventor. [online]. Copyright © 2012 [cit. 17.02.2018]. Dostupné z: http://appinventor.mit.edu/explore/ai2/setup-device-usb.html

37. LEGO® MINDSTORMS® | Explore MIT App Inventor. [online]. Copyright © 2012 [cit. 17.02.2018]. Dostupné z: http://appinventor.mit.edu/explore/content/legomindstorms.html

38. App Inventor Java Bridge - MIT App Inventor. [online]. [cit. 17.02.2018]. Dostupné z: http://www.appinventortojava.com/login/

39. Books | Explore MIT App Inventor. [online]. Copyright © 2012 [cit. 17.02.2018]. Dostupné z: http://appinventor.mit.edu/explore/books.html

40. App Inventor 2 Book: Create Your Own Android Apps. Learn to build Android apps | Appinventor [online]. [cit. 17.02.2018]. Dostupné z: http://www.appinventor.org/book2

41. MCGRATH, Mike. Building Android Apps in easy steps: Covers App Inventor 2 Second Edition. 2nd. UK: In Easy Steps Limited Warwickshire, 2014. ISBN 1840786299.

42. GUTHALS, Sarah. Building a mobile app: design and program your own app!. Indianapolis, IN: Wiley, 2017. ISBN 1119376424.

43. ROLLKE, Karl-Hermann. Android Apps with App Inventor 2: Easy App Development for Everyone. Oberer Teckelsberg 1 D-59846 Sundern: CreateSpace Independent Publishing Platform, 2018. ISBN 1983965049.

44. App Inventor for Educators – MIT App Inventor Educators Community. App Inventor for Educators – MIT App Inventor Educators Community [online]. [cit. 17.02.2018]. Dostupné z: http://teach.appinventor.mit.edu/

45. Tutorials for App Inventor | Explore MIT App Inventor. MIT App Inventor | Explore MIT App Inventor [online]. Copyright © 2012 [cit. 17.02.2018]. Dostupné z: http://explore.appinventor.mit.edu/ai2/tutorials

46. E-learningový kurz - Moodle Západočeské Univerzity v Plzni [online]. [cit. 17.02.2018]. Dostupné z: https://phix.zcu.cz/moodle/course/view.php?id=1501

47. Learn to build Android apps in hours – appinventor.org. [online]. [cit. 17.02.2018]. Dostupné z: http://appinventor.org

48. MIT App Inventor Support Forum – Skupiny Google. [online]. [cit. 17.02.2018]. Dostupné z: https://groups.google.com/forum/#!forum/mitappinventortest

49. An update on Eclipse Android Developer Tools - Android Developers Blog. [online]. [cit. 17.02.2018]. Dostupné z: https://android-developers.googleblog.com/2015/06/an-update-on-eclipse-android-developer.html

50. Download Android Studio and SDK tools | Android Developers. Android Developers [online]. [cit. 17.02.2018]. Dostupné z: https://developer.android.com/studio/

51. Meet Android Studio | Android Developers. Android Developers [online]. [cit. 17.02.2018]. Dostupné z: https://developer.android.com/studio/intro/

52. Run apps on the Android Emulator | Android Developers. Android Developers [online]. [cit. 17.02.2018]. Dostupné z: https://developer.android.com/studio/run/emulator#requirements_and_recommendations

53. Intro (For Android Studio). Learn to build Android apps | Appinventor [online]. [cit. 17.02.2018]. Dostupné z: http://www.appinventor.org/content/java-bridge/introduction/intro-android-studio

54. TIOBE Index | TIOBE - The Software Quality Company. Home | TIOBE - The Software Quality Company [online]. Copyright © 2018 TIOBE software BV [cit. 18.02.2018]. Dostupné z: https://www.tiobe.com/tiobe-index/

55. Nejpoptávanější programovací jazyky v roce 2017 - Hello, World!. Hello, World! [online]. Copyright © 2015, Hello, World [cit. 18.02.2018]. Dostupné z: http://www.helloworld.cz/nejpoptavanejsi-programovaci-jazyky-v-roce-2017/

56. Support Ended for Eclipse Android – Android developers. Android Developers [online]. [cit. 18.02.2018]. Dostupné z: https://android-developers.googleblog.com/2016/11/support-ended-for-eclipse-android.html

57. LACKO, Ľuboslav. Mistrovství - Android. Přeložil Martin HERODEK. Brno: Computer Press, 2017. Mistrovství. ISBN 978-80-251-4875-4.

58. DIMARZIO, J. F. Programujeme hry pro Android 4. Brno: Computer Press, 2012. ISBN 9788025137543.

59. Documentation | Android Developers. Android Developers [online]. [cit. 18.02.2018]. Dostupné z: https://developer.android.com/docs/

60. Android Development. Eclipse, Android and Java training and support [online]. [cit. 18.02.2018]. Dostupné z: http://www.vogella.com/tutorials/android.html

61. Programování Android aplikací v Javě. itnetwork.cz - Ajťácká sociální síť a materiálová základna pro C#, Java, PHP, HTML, CSS, JavaScript a další. [online]. Copyright © 2018 itnetwork.cz.Veškerý obsah webu [cit. 18.02.2018]. Dostupné z: https://www.itnetwork.cz/java/android

62. Kotlin. itnetwork.cz - Ajťácká sociální síť a materiálová základna pro C#, Java, PHP, HTML, CSS, JavaScript a další. [online]. Copyright © 2018 itnetwork.cz.Veškerý obsah webu [cit. 18.02.2018]. Dostupné z: https://www.itnetwork.cz/kotlin

63. Xamarin - Xamarin and the Universal Windows Platform. Learn to Develop with Microsoft Developer Network | MSDN [online]. Copyright © 2018 Microsoft [cit. 18.02.2018]. Dostupné z: https://msdn.microsoft.com/en-us/magazine/mt790185.aspx

64. Vývoj Multiplatformních mobilních řešení v sadě Visual Studio | Microsoft Docs. [online]. [cit. 18.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/visualstudio/cross-platform/cross-platform-mobile-development-in-visual-studio

65. Sole, Alessandro, Del. Xamarin.Forms Succinctly. Morrisville NC: Syncfusion, 2017, 127 p.

66. Integrované vývojové prostředí (IDE) nástroje Visual Studio. Visual Studio IDE, Code Editor, VSTS, & App Center - Visual Studio [online]. [cit. 18.02.2018]. Dostupné z: https://visualstudio.microsoft.com/cs/vs/

67. Visual Studio 2017 – požadavky na systém | Microsoft Docs. [online]. [cit. 18.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/visualstudio/productinfo/vs2017-system-requirements-vs#visual-studio-2017-system-requirements

68. Visual Studio 2017 for Mac – požadavky na systém | Microsoft Docs. [online]. [cit. 18.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/visualstudio/productinfo/vs2017-system-requirements-mac

69. Xcode – Wikipedie. [online]. [cit. 18.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/Xcode

70. Požadavky na systém - Xamarin | Microsoft Docs. [online]. [cit. 18.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/xamarin/cross-platform/get-started/requirements#windows-requirements

71. Nainstalujte na malou šířkou pásma nebo nespolehlivé mezi sítě v prostředích | Microsoft Docs. [online]. [cit. 18.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/visualstudio/install/install-vs-inconsistent-quality-network

72. Soubory ke stažení | Integrované vývojové prostředí (IDE), Code a Team Foundation Server | Visual Studio. Visual Studio IDE, Code Editor, VSTS, & App Center - Visual Studio [online]. [cit. 18.02.2018]. Dostupné z: https://visualstudio.microsoft.com/cs/downloads/

73. Co je potřeba udělat před instalací sady Visual Studio? - Visual Studio. Visual Studio IDE, Code Editor, VSTS, & App Center - Visual Studio [online]. [cit. 19.02.2018]. Dostupné z: https://visualstudio.microsoft.com/cs/vs/support/need-installing-visual-studio/

74. Visual studio - error MSB4062: The "Xamarin.Forms.Build.Tasks.GetTasksAbi" task could not be loaded from the assembly - Stack Overflow. Stack Overflow - Where Developers Learn, Share, & Build Careers [online]. [cit. 19.02.2018]. Dostupné z: https://stackoverflow.com/questions/50249343/error-msb4062-the-xamarin-forms-build-tasks-gettasksabi-task-could-not-be-loa

75. Instalační program Xamarin Live Player - Xamarin | Microsoft Docs. [online]. [cit. 19.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/xamarin/tools/live-player/install?tabs=ios%2Cwindows

76. Xamarin Live Player - Apps on Google Play. [online]. Copyright ©2018 Google [cit. 19.02.2018]. Dostupné z: https://play.google.com/store/apps/details?id=com.xamarin.live

77. Instalační program Xamarin Live Player - Xamarin | Microsoft Docs. [online]. [cit. 19.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/xamarin/tools/live-player/install?tabs=ios%2Cwindows

78. Hyper-V - Wikipedia. [online]. [cit. 19.02.2018]. Dostupné z: https://en.wikipedia.org/wiki/Hyper-V

79. Požadavky na systém pro emulátor sady Visual Studio pro Android | Microsoft Docs. [online]. Dostupné z: https://docs.microsoft.com/cs-cz/visualstudio/cross-platform/system-requirements-for-the-visual-studio-emulator-for-android

80. Xamarin Android Player - Xamarin. [online]. Copyright © [cit. 19.02.2018]. Dostupné z: https://developer.xamarin.com/releases/android/android-player/

81. Instalace sady Visual Studio s použitím parametrů příkazového řádku | Microsoft Docs. [online]. [cit. 19.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/visualstudio/install/use-command-line-parameters-to-install-visual-studio#list-of-workload-ids-and-component-ids

82. Extensible Application Markup Language – Wikipedie. [online]. [cit. 19.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/Extensible_Application_Markup_Language

83. What Is Intellisense? - Definition from Techopedia. Techopedia - Where Information Technology and Business Meet [online]. Copyright © 2018 Techopedia Inc. [cit. 19.02.2018]. Dostupné z: https://www.techopedia.com/definition/24580/intellisense

84. Úprava kódu ve Visual Studio IDE | Visual Studio - Visual Studio. Visual Studio IDE, Code Editor, VSTS, & App Center - Visual Studio [online]. [cit. 19.02.2018]. Dostupné z: https://visualstudio.microsoft.com/cs/vs/features/ide/

85. Xamarin – Community Books on Xamarin Development [online]. [cit. 19.02.2018]. Dostupné z: https://blog.xamarin.com/community-books-xamarin-development/

86. Learn Xamarin - YouTube. YouTube [online]. [cit. 19.02.2018]. Dostupné z: https://www.youtube.com/playlist?list=PLDzXQPWT8wED1eXjcfjGndwGVzBF8U7uO

87. Vytváření mobilních aplikací s Xamarin.Forms knihy - Xamarin | Microsoft Docs. [online]. [cit. 19.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/xamarin/xamarin-forms/creating-mobile-apps-xamarin-forms/

88. Dokumentace pro Xamarin - Xamarin | Microsoft Docs. [online]. [cit. 19.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/xamarin/

89. Xamarin Community Forums. Xamarin Community Forums [online]. Copyright © [cit. 20.02.2018]. Dostupné z: https://forums.xamarin.com/

90. Visual studio - Why can not I choose a PCL Project in Xamarin Forms? - Stack Overflow. Stack Overflow - Where Developers Learn, Share, & Build Careers [online]. [cit. 20.02.2018]. Dostupné z: https://stackoverflow.com/questions/48492789/why-can-not-i-choose-a-pcl-project-in-xamarin-forms

91. Přehled sdílení kódu - Xamarin | Microsoft Docs. [online]. [cit. 20.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/xamarin/cross-platform/app-fundamentals/code-sharing#summary

92. Xamarin Forms – Xamarin University [online]. [cit. 20.02.2018]. Dostupné z: https://elearning.xamarin.com/forms/xam110/pcls/

93. Rychlý start Xamarin.Forms - Xamarin | Microsoft Docs. [online]. [cit. 20.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/xamarin/xamarin-forms/get-started/hello-xamarin-forms/quickstart?tabs=vswin

94. Dokumentace k rozhraní .NET | Microsoft Docs. [online]. [cit. 20.02.2018]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/index#pivot=docs&panel=getstarted

95. Největší český C# .NET portál a kompletní on-line kurzy. itnetwork.cz - Ajťácká sociální síť a materiálová základna pro C#, Java, PHP, HTML, CSS, JavaScript a další. [online]. Copyright © 2018 itnetwork.cz.Veškerý obsah webu [cit. 20.02.2018]. Dostupné z: https://www.itnetwork.cz/csharp

96. WYSIWYG – Wikipedie. [online]. [cit. 20.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/WYSIWYG

97. What are BlueStacks 4 system requirements? – BlueStacks Support. [online]. [cit. 20.02.2018]. Dostupné z: https://support.bluestacks.com/hc/en-us/articles/360004124812-What-are-BlueStacks-4-system-requirements-

98. Genymotion Cloud – Online Android Emulator. Genymotion Android Emulator | Cloud-based Android virtual devices | Develop - Automate your tests -

Validate with confidence [online]. Copyright © Genymobile 2014 [cit. 20.02.2018]. Dostupné z: https://www.genymotion.com/cloud/

99. Xamarin Android Player - Xamarin. [online]. Copyright © [cit. 20.02.2018]. Dostupné z: https://developer.xamarin.com/releases/android/android-player/

100. Emulátor pro aplikace Androidu | Visual Studio. Visual Studio IDE, Code Editor, VSTS, & App Center - Visual Studio [online]. [cit. 20.02.2018]. Dostupné z: https://visualstudio.microsoft.com/cs/vs/msft-android-emulator/

101. Seznam škol v ČR – Seznamškol. Cz. [online]. [cit. 20.02.2018]. Dostupné z: http://Seznamskol.cz

102. Střední školy – Seznam středních škol. [online]. [cit. 20.02.2018]. Dostupné z: http://Stredniskoly.cz

103. Anybody can learn | Code.org. Anybody can learn | Code.org [online]. [cit. 20.02.2018]. Copyright © Code.org, 2018. Code.org [cit. 14.08.2018]. Dostupné z: https://code.org/

104. Kodu | Home [online]. [cit. 20.02.2018]. Dostupné z: https://www.kodugamelab.com/about/

105. NetBeans – Wikipedie. [online]. [cit. 20.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/NetBeans

106. Lego Mindstorms – Wikipedie. [online]. [cit. 20.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/Lego_Mindstorms#Historie

107. Pascal (programovací jazyk) – Wikipedie. [online]. [cit. 20.02.2018]. Dostupné z: https://cs.wikipedia.org/wiki/Pascal_(programovac%C3%AD_jazyk)

7 Přehled příloh:

Přílohy jsou dostupné on-line v uložišti OneDrive: https://goo.gl/4KwHTJ