Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra Softwarového Inženýrství

Bakalářská práce

Animovaný avatar pro výukovou aplikaci Dráček

Pavel Podaný

Vedoucí práce: Ing. Jiří Chludil

10. ledna 2016

Poděkování

Rád bych poděkoval svému vedoucímu Ing. Jiřímu Chludilovi za jeho čas, rady a postřehy, které vedly k finální podobě a obsahu této práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 10. ledna 2016 .....

Odkaz na tuto práci

Podaný, Pavel. *Animovaný avatar pro výukovou aplikaci Dráček*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Aplikace Dráček představuje výukový program zaměřený na děti základních škol s poruchami učení. Tato práce se zabývá myšlenkou animovaných průvodců, rozborem jejich hlavních rysů a zkoumáním jejich úspěšnosti za účelem návrhu postavičky vlastní pro použití v nové mobilní verzi této aplikace. Práce uvažuje možnosti ovládání takovýchto pomocníků a problém volby vhodné reakce na příchozí podměty.
Získané informace jsou nakonec použity k návrhu a implementaci aplikace sloužící jako simulátor a grafický konfigurátor chování této postavy.

Klíčová slova výuková aplikace Dráček, avatar, průvodce, animace, návrh

Abstract

Application Dragon represents an educational software designed to be used by children affected by various learning disorders. The contents of this work discuss the idea of an animated character guide and analyse its strengths and features in order to design a character of its own to be used in a new mobile version of the application. The work examines the ways of controlling such characters and the problem of choosing an ideal reaction based on the application inputs.

The gathered information is then used for designing and implementation of an application to be used for configuration and simulation of the character's behavior.

Keywords educational application Dragon, avatar, guide, animation, design

Obsah

Seznam obrázků

Úvod

Projekt Dráček

Počátek projektu Dráček se datuje do začátku roku 2012, kdy byla Fakulta informačních technologií ČVUT kontaktována Základní Školou Smečno s žádostí o spolupráci na výukové aplikaci použitelné pro jejich specifické potřeby. Jedním z hlavních požadavků takovéto aplikace bylo zacílení se na děti s poruchami učení, jejichž výukou se ZŠ Smečno zabývá. Projektu se tehdy ujala skupina studentů z předmětu Týmový Projekt pod vedením pana Ing. Jiřího Chludila a brzy zjistila, že v aplikaci se nacházel mnohem větší potenciál než jaký byl od zadavatele požadován. Tým se tedy rozhodl tyto možnosti blíže prozkoumat a vznikl rozsáhlý projekt, na jehož dalších částech se pracuje dodnes.
Jednou z takovýchto částí je i varianta této aplikace na mobilní zařízení se systémem Android, jejíž vývoj se v současných chvílích plánuje. Tato aplikace se však řídí vzorem svého předchůdce a klade si mnohem vyšší cíle než být pouhou kopií již existující počítačové verze.
Výsledná aplikace by měla být schopna bez dalších nastavování stahovat a integrovat dodatečné přídavné moduly obsahující nová cvičení a zadání, komunikovat se serverem pro ukládání výsledků případně nabízet i takové možnosti jako nahlížení učitelem do stavu práce libovolného studenta v reálném čase pro možnost rychlé asistence.

Cíl práce

Jednou ze zmiňovaných novinek chystaných pro novou aplikaci Dráček se zabývá právě tato práce. Tvorba nového líbivého grafického rozhraní je nepochybně důležitým úkolem, tato činnost je však o to důležitější v případě, že koncovými uživateli jsou děti.
Udržení dětské pozornosti se často může ukázat být složitým, ne-li i nemožným

úkolem. Vizuální část takovéto aplikace se právě pro tento cíl stává jedním z nejmocnějších prostředků.
Pro tyto účely tedy bylo provedeno rozhodnutí o navržení animovaného avatara, který bude přítomný na obrazovce zařízení kde bude dynamicky reagovat na nejrůznější podměty spouštěním vhodných animací.
Samotné zpestření uživatelského rozhraní o pohybující se postavičku je velkým krokem k udržení pozornosti dítěte, avatar se však dá využít i mnoha dalšími způsoby, například jako průvodce a pomocník.

Rozbor zadání práce

Tato část pojednává o autorovu pochopení zadání práce a argumentuje směry, kterými se práce vydává.

Analyzujte aplikaci Dráček a vyberte vstupní informační kanály určené pro ovládání Avatara

Tento úkol zadává analýzu cílové aplikace a úvahu nad tím, jaké události lze použít pro ovládání reakcí navrhované postavičky. Lze sem zahrnout rozbor možných ovládacích prostředků, současných, již existujících řešení a případné další inspirace čerpané z provedení ovládacího rozhraní nejen cílové aplikace.

Analyzujte výrazové prostředky avatara (gesta, činnosti, atd.) vhodné pro výukovou aplikaci pro děti předškolního a školního věku

Výrazové prostředky avatara v tomto úkolu představují samotné animace, tedy reakce, které takováto postavička může obsahovat. V analýze se opět lze inspirovat avatary a průvodci, které již ve světě existují a na základě odhadu jejich popularity se zamyslet nad tím, které z vlastností je odlišují od ostatních a tedy proč jsou či nejsou populární.

Navrhněte modul Avatar použitelný aplikací Dráček pro vizuální interakci s uživatelem (prostřednictvím avatara)

Vzhledem k velmi ranému stádiu vývoje moderní verze aplikace v době psaní této práce není možné udělat modul součástí této (stále neexistující) aplikace. Zvolená varianta řešení se proto bude zabývat návrhem modulu schopného úprav možností chování avatara dle požadavků uživatele a jeho řízení.

Implementujte tento modul pod OS Android

Navržený modul bude naimplementován takovým způsobem, aby byl spustitelný na operačním systému Android se zaměřením na zařízení tabletů, které jsou cílovou platformou aplikace Dráček pod tímto systémem.

Otestujte uživatelskou přívětivost a srozumitelnost avatara v UX laboratoři na cílové skupině uživatelů

Úkol zadává provedení testů přívětivosti / jednoznačnosti uživatelského rozhraní výsledného modulu na uživatelích, u kterých se předpokládá jeho používání. Tyto testy budou provedeny v prostorách fakulty v laboratoři k tomu určené.

Kapitola 1

Analýza současných řešení animovaných průvodců a user agentů

Myšlenka více či méně inteligentních pomocníků pro používání nejrůznějších druhů software je na světě již od dob prvních GUI (Graphical User Interface) programů a do dnes se rozmanitými způsoby používá.
Jedním z raných příkladů takovéhoto pomocníka může být například animovaný pes Rover vyvinutý společností Microsoft v roce 1995 jako součást projektu *Microsoft Bob*.

Obrázek 1.1: Microsoft Bob, Rover

Tento projekt, jehož cílem bylo vyvinutí přívětivého rozhraní pro nové uživatele počítačů však bohužel selhal. Důvody proč tomu tak bylo mohly být například vysoká cena programu, který nenabízel téměř nic užitečného nebo pro tehdejší dobu vysoké hardwarové nároky.
Vývoj interaktivních pomocníků však nadále pokračoval a své uplatnění si

našel především v oboru video her, kde je nějaká jejich forma velmi často používána pro seznámení hráče s prostředím.

Tato kapitola se zabývá představením několika druhů již existujících a často známých postaviček, jejichž chování je poté podrobně rozebráno. Na základě této analýzy je nakonec sestavena tabulka, která se snaží objasnit důvod úspěchu (či neúspěchu) těchto postav.
Za tímto účelem je tedy ze všeho nejdříve nutno nějakým způsobem ustanovit zkoumané vlastnosti avatarů.

1.1 Ustanovení metrik kvality avatara

Určení kvality takovéhoto průvodce či pomocníka se na rozdíl od klasických technologií nedá považovat za exaktní vědu a jednoduše naměřit výkonnostní rozdíly. Jedná se spíše o z velké části subjektivní záležitost k jejíž analýze se musíme obrátit nejen k psychologii.
Je tedy třeba určit si vlastní měřítka kvality, na základě kterých se pokusíme posoudit co stojí za úspěchem jednotlivých avatarů.
Za podstatný faktor lze zajisté považovat vizuální stránku postavičky. Prvních několik málo vteřin kontaktu často rozhoduje zda-li si uživatel avatara oblíbí nebo ho naopak bude nenávidět. Lidé, především děti, mají tendenci snadněji si oblíbit věci, které považují nějakým způsobem za roztomilé, což lze také vidět na většině designů úspěšných avatarů. Co lze tedy považovat za roztomilé? Rakouský zoolog, etolog a ornitolog Konrad Lorenz charakterizoval[1] *roztomilost* těmito znaky

- Velká hlava v porovnání s tělem, zakulacená
- Velké čelo
- Velké oči v porovnání s obličejem posazené pod středem hlavy
- Zakulacené vyčnívající tváře
- Zakulacený tvar těla
- Měkký povrch těla

Za jeden z dalších znaků roztomilosti se dá v případě zvířat považovat také krátký čenich.

Dále se zaměříme na *pasivní chování animací* avatara za chodu aplikace, tedy zda-li postava nějakým způsobem reaguje pouze na přímou interakci nebo zda-li existují i jiné prostředky obohacující její chování mimo běžné vstupní akce (animace po spuštění aplikace, uplynutí delšího časového intervalu bez reakce uživatele atp.).

Jednou z klíčových vlastností pomocníka je zajisté jeho *užitečnost*, tedy jak přínosná přítomnost postavičky v aplikaci ve skutečnosti je a zda-li s sebou přináší zlepšení použitelnosti.

Nehledě na vzhled či užitečnost, žádný uživatel si avatara neoblíbí, pokud se bude snažit být přespříliš užitečný, tedy bude svým chováním *otravný či obtěžující*. Do této kategorie můžeme také zařadit do jaké míry se avatar umí přizpůsobit preferencím uživatele.

Posledním z testovaných faktorů bude možnost *přizpůsobení* vzhledu postavičky. Zde se zaměříme na to, do jaké míry lze avatara a jeho bezprostřední okolí přizpůsobit vizuálním požadavkům uživatele.

Ustanovené faktory tedy jsou

- Roztomilost dle Lorenzových kritérií
- Pasivní chování animací
- Užitečnost
- Potenciál pro obtěžování
- Možnosti přizpůsobení

Váhu významu těchto faktorů opět není možné exaktním způsobem určit, bylo pro ně proto ustanoveno měřítko splnění na stupnici od 1 do 5, kde více znamená lépe. Toto se netýká kritérií roztomilosti, která lze hodnotit jednoduchým „splněno / nesplněno“ způsobem.

1.2 Představení zvolených zástupců

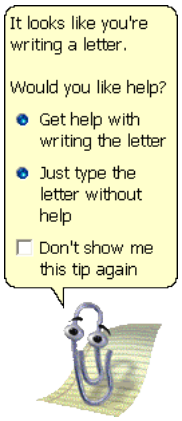
Od prvních nejistých kroků Rovera se na světě zrodilo nepřeberné další množství animovaných asistentů a průvodců. Typ jejich určení lze v podstatě rozdělit na dvě velké skupiny.

- *Postavičky, které jsou samostatnou součástí aplikace*
  Pomocníci z této skupiny jsou skutečně jen pouhými pomocníky. Jejich role není pro chod dané aplikace žádným způsobem podstatná a aplikaci by bylo možné kompletním způsobem používat i bez jejich přítomnosti. Funkce kterou v tomto případě zastávají se omezuje na akce jako je nápověda, užitečné tipy, částečná automatizovaná asistence atp. Klasickým příkladem této skupiny je neslavný Pan Sponka (anglicky také Clippit / Clippy).

- *Postavičky, které jsou integrální součástí aplikace*
  Pomocníci spadající do této skupiny jsou pro chod a funkci dané aplikace nějakým způsobem podstatní a nepostradatelní, tudíž se s jejich přítomností počítá již od samotných základů a aplikace je tomu přizpůsobena.
  Pokud by takovýto avatar v aplikaci chyběl, zásadním způsobem by to omezilo její funkčnost, například by nebylo možné vyvolat akce pro postup dále nebo by nebylo možné jak předat požadované instrukce. Příkladem takovéto postavičky je myšák Adam, z aplikace „*Jdu do školy!*“ vyvinuté fakultou informačních technologií ČVUT.

V této části si určíme a popíšeme několik vybraných exemplářů, které budou později použity pro celkovou analýzu žádoucích i nežádoucích vlastností.

1.2.1 Pan Sponka



Obrázek 1.2: Microsoft Office Assistant, Clippy (2000)

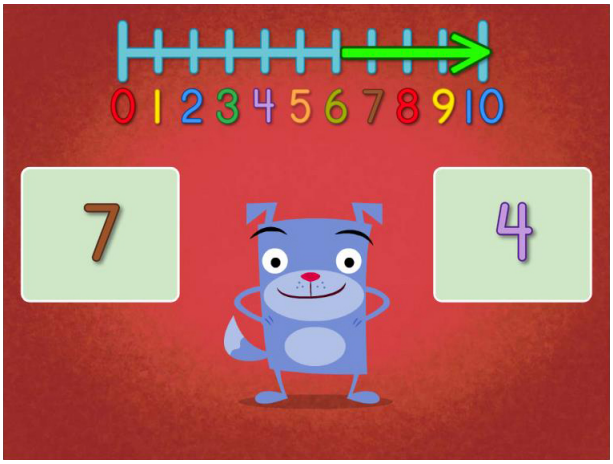
Pan Sponka byl poprvé uveden v roce 1997 jako interaktivní pomocník ke kancelářskému balíku Microsoft Office pro Windows. Asistent měl více možných podob a jmen, Clippy byl však nastaven jako defaultní volba vzhledem k jeho blízko k významu software.
Tento avatar si ve světě vedl lépe než jeho předchůdce Rover, přesto však vyvolával převážně negativní reakce. Důvodem proč tomu tak bylo se v roce 2003 rozsáhle zabýval student Stanfordské univerzity Luke Swartz ve své závěrečné práci „Why people hate the paperclip: labels, appearance, behavior and social responses to user interface agents“[2].
Vzhledem k zastaralosti a odstranění avatara ze současných verzí aplikace jsou informace o jeho chování z části čerpány z výše zmiňované práce.

1.2.2 Myšák Adam

Obrázek 1.3: „Jdu do školy!“, Adam [3]

Myšák Adam je průvodní postavičkou z dětské výukové hry „*Jdu do školy!*“, jehož úkolem je seznámení dětí s prostředím hry a úkoly jednotlivých mini her. Jelikož je aplikace zaměřena na děti předškolního věku, forma komunikace byla provedena pomocí namluvených dialogů vysvětlujících pravidla a postupy. Postava je přítomna po celou dobu hry, průběžně komentuje správnost uživatelových akcí a v případě potřeby poskytuje nápovědu.

1.2.3 Floyd



Obrázek 1.4: I'm Thinking of a Number, Floyd [4]

Floyd je součástí série dětských výukových her nacházejících se na internetových stránkách www.education.com, zabývajících se výukou dětí především

nižšího věku.
Tento animovaný pes se vyskytuje jako doprovodný prvek v některých z výše zmiňovaných krátkých hříček a jeho role je především vizuální. Postavička je zapojena do příběhu hry a někdy obsahuje i doprovodné animace k dění ve hře, instrukce a komentáře jsou však podávány hlasem třetí osoby.
Animované akce, které Floyd zrovna provádí jsou z velké části závislé na vstupu uživatele. Pro příklad, na obrázku 1.4 vidíme Floyda ve hře jménem I'm Thinking of a Number, kde vypravěč tvrdí, že si postavička myslí číslo, které je větší/menší než nějaké jiné číslo a dítěti jsou dány dvě možnosti. Pokud zvolí správně, Floyd s úsměvem přikývne, v opačném případě zklamaně zakroutí hlavou.

1.2.4 Kocour Tom

Obrázek 1.5: My Talking Tom, Tom [5]

Tom se poprvé stal slavným díky jednoduché mobilní aplikaci jménem Talking Tom Cat, prezentující postavu kočky jejíž přední vlastností byla schopnost mluvit, přesněji pozměněným hlasem opakovat uživatelova slova. Tom byl, zejména u dětí, velice populárním nápadem a vývojáři ho tedy přenesli ještě o několik úrovní výše pomocí aplikace My Talking Tom.
V této hře, velice připomínající starší zařízení tamagotchi (digitální zařízení simulující starání se o zvířátko), je hráči předána péče o právě narozené kotě (jehož jméno závisí na přání hráče) a s tím i všechny starosti a zodpovědnosti s tím související. Uživatel postavičku krmí, myje, hraje si s ní a celkově zajišťuje její plnou spokojenost. Pokud se mu to daří, kočka postupně získává vyšší úrovně, se kterými dospívá a odemyká nové možnosti.
Stejně jako u zvířátka tamagotchi se i zde jedná o časově dlouhodobou záležitost. Tom však v případě zanedbání na rozdíl od zmíněného zařízení nikdy nezemře.

1.2.5 Pou

Obrázek 1.6: Pou [6]

Jiným pojetím starání se o zvířátko je Pou. Aplikace si klade stejné cíle jako dříve zmiňovaný Tom, vše je však provedeno trochu jiným způsobem. Tam kde Tom vsází na graficky líbivé provedení plně využívající 3D výpočetní schopnosti moderních zařízení, tato aplikace se vydává směrem jednoduchých 2D kreseb, které preferují rychlost aplikace namísto pohlcující vizuální stránky.
Tato blíže nespecifikovaná mimozemská životní forma Pou opět spoléhá na zásahy uživatele, které mají za úkol zajišťovat panáčkovo pohodlí a spokojenost. Jako zástupce lze jmenovat například krmení, hraní mini her, starání se o zahradu a mnoho dalších.

1.3 Bližší rozbor

Každý z výše představených avatarů byl navržen s jistým záměrem pro jejich existenci, který se mezi nimi liší, všichni z nich se však snaží uspět a přilákat pozornost uživatelů. Čas a realita ukázali jak moc se jim tyto cíle podařilo naplnit.

1.3.1 Pan Sponka

Pravděpodobně nejslavnější z řad animovaných pomocníků a průvodců, Pan Sponka se ve světě stal téměř neslavnou legendou. S cílem hravé asistence pro nové a technologicky nezkušené uživatele softwaru Microsoft Office se tato postavička stala noční můrou mnohých.
Důvodů proč tomu tak mohlo být je mnoho, počínaje od nepříliš populárního

vzhledu postavičky až k nekonečně se opakujícímu obtěžování uživatele. Ty nejzávažnější z prohřešků se zde pokusíme blíže rozebrat.

- Příliš velká svoboda
  Jedním ze známých problémů tohoto avatara byla jeho schopnost svévolně interagovat se vstupem uživatele. Pokud uživatel například začal psát textový dokument a Clippy se rozhodl, že se jedná o dopis, ve svém pokusu být nápomocný upravil patřičně formátování dokumentu a případně vložil chybějící šablonový obsah.
  Rozzuřený uživatel v takovémto momentě musel strávit cenný čas vrácením dokumentu do původní podoby, jen aby jeho pokus o pokračování v práci byl přerušen novou „asistencí“ pana Sponky, což nás vede k bodu dalšímu.

- Nepřizpůsobení se uživateli
  Pokud asistent provedl určitou akci a uživatel s ní nebyl spokojen a ručně změny vrátil, asistent si nespokojenosti často nedovedl všimnout a změny provedl znovu, přestože byly nežádané.

- Složitá deaktivace
  Deaktivace pana Sponky, především v raných verzích aplikace vyžadovala ruční změnu aplikačních souborů a znalost tohoto kroku. To bylo v budoucích verzích aplikace podstatně zlepšeno možností deaktivace při změně instalace aplikace, stále se však jednalo o relativně obtěžující krok (zvláště v případech kdy pro tuto akci bylo vyžadováno instalační cd aplikace).

- Neužitečnost, opakující se tipy a obtěžování
  Přestože avatarovi rady mohly být užitečné pro začínající uživatele, pro ty z řad zkušenějších již často byly nepotřebné až obtěžující, zejména pokud se stále opakovaly a narušovali jejich práci a soustředění.

Dodatečné informace k testovaným vlastnostem
Pasivní chování animací - Clippy dokázal animacemi reagovat na některé podměty jiné než přímá interakce, tyto animace se však spustily pouze v případě, že akce byla provedena přes avatara
Možnosti přizpůsobení - Možnost změny vzhledu byla realizována pomocí schopnosti vyměnit celého asistenta za jiný typ (např. psa či kočku). Barevné provedení postaviček však přizpůsobit nešlo.

1.3.2 Myšák Adam

Role pana Sponky v jeho domovském software byla ta nezávislého pozorovatele a pomocníka a aplikace samotná s jeho přítomností pro svůj chod nijak

nepočítala. Jiného přístupu se dočkal Adam v dětské výukové aplikaci „*Jdu do školy!*“.
Postavička v této hře představuje klíčový prvek spojující řadu samostatných mini her do jednoho kompletního celku, čehož dosahuje především namluvenými monology. Zvolení hlasového výstupu bylo vzhledem k předpokládanému věku uživatelů a nutnosti konstantního předávání informací správným krokem, při testování však byla nalezena následující možná úskalí.

- Příliš časté komentáře
  Jak již bylo zmíněno, myšák krom vysvětlení vlastních pravidel jednotlivých her také poskytuje průběžné komentáře k akcím hráče. Tyto komentáře však často nastávají téměř po každém kliknutí a jsou vybírané z velmi omezené větné zásoby, tudíž se velmi často opakují a již po několika minutách mohou být obtěžující a to jak pro hráče, tak pro jeho okolí (např. rodiče).

- Nemožnost přeskočení vysvětlení pravidel a příběhu
  Pokud se hráč ve hře nachází poprvé, vysvětlení jednoduchých pravidel je velmi podstatné, jelikož by aplikace bez něj nedávala příliš smysl. Pokud si však v průběhu hry svůj postup uloží a znovu se vrátí nebo se rozhodne nějakou z her spustit znovu, je nucen znovu poslouchat často relativně dlouhé povídání které mu již nedokáže sdělit nic nového a pouze ho připravuje o čas.

- Nedostatečný vizuální doprovod
  Naprostá většina informací je v aplikaci z pochopitelných důvodů (určení pro děti předškolního věku) sdílena zvukovou podobou. Podávat takovéto informace textem by bylo nesmyslné, hra se však ve většině případů, krom ukázání jednoduchých příkladů fungování mini her, nesnaží povídání nějakým způsobem vizuálně obohatit a obraz zůstává neměnný. To, nejen u dítěte, může vést k jednoduché ztrátě pozornosti.

Dodatečné informace k testovaným vlastnostem
Možnosti přizpůsobení - Možnosti úpravy vzhledu avatara jsou v této aplikaci velice omezené až žádné (Adam se na začátku hry zeptá na barvu kalhot).

1.3.3 Floyd

Hlavní hrdina některých výukových online her na stránkách www.education.com se narozdíl od jiných nesnaží být ničím výjimečný. Jeho jediná role v příslušných aplikacích je být vizuální ozdobou, lákadlem pro děti a doprovodným prvkem pro tvorbu jednoduchého příběhu mini hry - a v těchto činnostech exceluje. Zkusíme se tedy blíže podívat na to, co Floyda vede k úspěchu.

- Pasivní a aktivní animovaná interakce
  Hlavním záměrem Floydovi existence je vizuální interakce s dítětem, které aplikaci používá. Postavička je přítomna na nějakém prominentním místě na obrazovce (obvykle ve středu) a příslušnými animacemi reaguje na zadaný vstup (správný / špatný). Mimo akce, které jsou přímo spojené s uživatelovým vstupem však nabízí také zásobu animací spuštěných z jiných zdrojů. Kupříkladu, při spuštění aplikace „I'm thinking of a number" postavička nejprve přiběhne ze vzdálené části obrazovky a teprve poté se kolem ní spustí samotná hra, v jiné hře zase během úvodního monologu vypravěče Floyd s úsměvem mává na uživatele.

- Neobtěžující přítomnost
  Mimo předem určených klíčových momentů Floyd na obrazovce nedělá žádné akce, které by mohly uživatele obtěžovat či rozptylovat. Reakce na zvolený vstup jsou také velmi jemné a neagresivní (krátký zvukový efekt spolu s animací namísto hlasového oznámení o chybné akci).

- Líbivý vzhled
  Přestože oblíbenost vzhledu postavičky je velmi ovlivněna individuálními pocity uživatele, existují určité zásady které lze následovat pro maximalizování šancí oblíbenosti avatara. Floydovo vizuální ztvárnění působí přátelským, roztomilým dojmem, který by měl ve většině případů mít na uživatele pozitivní vliv.

Dodatečné informace k testovaným vlastnostem
Užitečnost - Význam postavičky Floyda v této aplikaci neplní žádnou funkční roli, jeho úloha je však přesto velice důležitá. Přítomnost postavičky dodává celé hře určitou osobnost, čímž podporuje udržení pozornosti dítěte a šanci, že hru dítě vůbec spustí.
Možnosti přizpůsobení - Možnost úpravy vzhledu avatara v této aplikaci neexistuje.

1.3.4 Tom

Postavička této aplikace není pouze její podstatnou součástí, dalo by se totiž říci, že Tom je aplikací samotnou. Vše co tento software poskytuje je úzce svázáno s nějakým způsobem interakce s tímto kotětem a nemá žádný jiný účel. Oblíbenost této postavy je zde tedy extrémně důležitým prvkem. Pokud se uživateli avatar nějakým způsobem nelíbí, ztrácí pro něj aplikace význam. Soudě dle recenzí a populárnosti aplikace [5] byl však Tom lidmi přijat převážně v pozitivním světle. Čím se tomu tedy zasloužil?
Vizuální stránka postavičky je v takovémto případě velice podstatná, přináší však s sebou jisté problémy. Lidský vkus je velmi složitý a rozmanitý a je nemožné ho vystihnout jen pomocí jednoho fixního stylu. Nejen proto aplikace přináší možnost bohatého vizuálního přizpůsobení avatara vkusu uživatele,

včetně přizpůsobení jakéhokoli vybavení místností.
Toho lze dosáhnout použitím jakéhosi šatníku kotěte, ve kterém lze, za splnění jistých předpokladů, vybírat z velkého množství vzhledů srsti, očí, oblečení a kostýmů, případně zvolením úprav místnosti, ve které se uživatel zrovna nachází. Zmiňované předpoklady obnáší současnou úroveň Toma a dostatek virtuální měny pro koupi doplňku. Této měny lze nabít buďto plněním úkolů, které jsou hráči zadávány a postupným nasbíráním dostatečného množství nebo použitím reálných peněz.
To nás přivádí k jedné z nepříliš atraktivních vlastností aplikace. Software je poskytován zdarma ke stažení a je tedy prostoupen reklamami zobrazovanými při hraní a častým pobízením hráče k použití mikrotransakcí, kterými lze mnohé z věcí podstatně urychlit.
Přestože povrchní vzhled Toma lze libovolně a bohatě měnit, rozložení jeho těla však po čas hry zůstává stále stejné (nebereme-li v potaz postupné dospívání, které vzhled příliš nemění). V tomto ohledu bylo nutné zajistit zvolení vhodného návrhu postavičky a i zde se, na první pohled roztomilý charakter, setkává s úspěchem, možná proto, že splňuje všechny z Lorenzových kritérií roztomilosti.
Chování postavy uvnitř hry samotné je relativně pasivní a nerušivé, ačkoli všechny Tomovi urgentní potřeby jsou zdůrazněny příslušnou animací (unavenost, špatná nálada, hlad...). Při starání se o kotě jsou všechny jeho reakce také patřičně zanimovány (vrnění při gestu hlazení, spaní...).
Hra však od uživatele očekává dlouhodobou časovou investici a pro chvilkovou zábavu toho nemusí příliš nabízet. Nepříjemné pro někoho mohou být vyskakující notifikace na zařízení pravidelně uživatele informující o potřebách zvířátka, pokud zrovna nějaké má, nebo nutnost dlouhého čekání, než se kotě vyspí. To se však dá považovat za princip hry a jiní uživatelé tyto věci mohou zase ocenit.
Za zmínku nakonec stojí také atraktivní 3D provedení celé vizuální stránky aplikace, které však může nadměrně zatěžovat pomalejší zařízení.

1.3.5 Pou

Tato aplikace funguje na stejném principu jako Tom a velká část činností a reakcí je téměř totožná. Za hlavní odlišnost by se dalo považovat především jiné vizuální provedení celé aplikace.
Zatímco u Toma se uživatel může setkat s propracovaným 3D prostředím plného textur, tato hra se vydala cestou jednoduchého obrázkového umění s jednobarevným pozadím místností, čímž se podstatně snižují hardwarové nároky na zařízení. I přes svou jednoduchost však vizuální stránka hry zůstává stále atraktivní.
Stejně jako předchozí zástupce, i tato aplikace umožňuje rozmanité úpravy barev postavičky a předmětů v místnostech, přestože na jednodušší úrovni. Zpoplatnění doplňků reálnou či virtuální měnou zůstává stejné.

Vítanou funkcí je zde možnost deaktivace notifikací o potřebách postavičky.

1.4 Shrnutí výsledků analýzy

Informace získané v předchozích částech textu lze shrnout do závěrečné tabulky určující do jaké míry jednotlivý zástupci vyhovují na začátku stanoveným metrickým údajům.
Zatímco výsledky vyhovění stanoveným kritériím roztomilosti lze zapsat pomocí jednoduchého ano/ne, ostatní části již vyžadují drobnější měřítko úspěšnosti.
Pro následující tabulku si tedy zavedeme bodový systém od 1 do 5, kde více bodů znamená větší úspěšnost. Z důvodu konzistence bodování si také údaj „Potenciál pro obtěžování“ převedeme na „Nerušivost“.

	Sponka	Adam	Floyd	Tom	Pou
Pasivní chování animací	3	1	4	5	4
Užitečnost	2	4	3	5	5
Nerušivost	1	3	5	4	5
Možnosti přizpůsobení	2	1	1	5	4
Celkem bodů	8	8	13	19	18
Popularita	2	1	3	5	5
Velká zakulacená hlava	✗	✓	✗	✓	✓
Velké čelo	✗	✗	✓	✓	✗
Velké oči vůči obličeji	✗	✗	✓	✓	✓
Zakulacené vyčnívající tváře	✗	✗	✓	✓	✓
Zakulacený tvar těla	✓	✓	✗	✓	✓
Měkký povrch těla	✗	✓	✓	✓	✓

Z tabulky vidíme, že Tom si vedl velice dobře. Jeho jediným nedostatkem v zavedeném systému metrik byla možnost obtěžování skrze notifikace, jež nelze deaktivovat. Můžeme si také všimnout, že vzhled postavičky splňuje všechny z Lorenzových kritérií.
Velice blízko za ním je stejně provedený Pou, který nabízí jednodušší animace a méně možností přizpůsobení, nahrazuje to však možností deaktivace notifikací.
Třetí místo obsadil pes Floyd, jehož hlavním neduhem oproti oponentům byla chybějící možnost uzpůsobení. Dalo by se však argumentovat, že v kratičkých nesouvislých mini hrách by se jednalo o relativně zbytečný prvek. Naopak za největší sílu lze v jeho případě považovat nerušivou, příjemnou přítomnost.

Pro možnost snadného porovnání s realitou je pod součtem získaných bodů přidáno také hodnocení reálné popularity postaviček. Toto hodnocení nemá samo o sobě žádnou váhu, jedná se však o zajímavé porovnání teoretických vlastností a reálného přijetí.
Celkově lze z tabulky vypozorovat, že úspěšnější avataři splňují více předpokladů roztomilosti, než jejich méně úspěšní kolegové. Může se zde jednat o náhodu, vzhledem k malému počtu testovaných exemplářů, a vzhled samotný rozhodně nezaručuje úspěšnost postavičky, přesto je toto zjištění užitečné a dodržení kritérií při návrhu může vést pouze k pozitivnímu efektu.

Kapitola 2

Analýza vstupních informačních kanálů pro ovládání avatara

Nehledě na to jaké animace a funkce jednotlivé postavičky poskytují, jejich role v aplikaci by byla bezvýznamná, kdyby tyto prvky nebylo možné správným způsobem ovládat.
Tato část se tedy bude zabývat identifikací částí aplikace, které lze použít pro řízení interakce s avatarem, tedy především pro spouštění vhodných animací. Ze zkušeností lze předpokládat, že většinová část podmětů pro reakci bude pocházet z nějaké interakce uživatele s postavičkou. Jelikož je aplikace zaměřena na děti a běžné uživatele, pro její ovládání bude použito grafické uživatelské rozhraní. Ke zjištění možností, které takové rozhraní nabízí je proto vhodné se nad ním blíže zamyslet a na tomto vytvořit kvalifikovaný odhad toho, co lze od výsledné aplikace očekávat. Jelikož nová verze rozhraní aplikace Dráček prozatím neexistuje, je k získání co nejbližšího závěru o jeho podobě provedena analýza návrhu uživatelských rozhraní zaměřených na dětské uživatele.
V kapitole je nakonec uvažováno několik dalších, předem nezmíněných, vstupních událostí a vše je poté shrnuto do závěrečného seznamu.

2.1 Analýza uživatelského rozhraní aplikací

Velká část vstupů použitých k ovládání avatara pochází z uživatelského rozhraní samotné aplikace, které je s postavičkou nějakým způsobem propojeno. Možných prostředků jak tohoto dosáhnout existuje více a v dalších částech se nad nimi blíže zamyslíme, zde se však podíváme na možnosti samotného rozhraní.
Pro analýzu možností interakce s aplikací a následného vyvolání výstupní reakce postavičky lze rozebrat ovládací mechanismy běžně používaného software navrženého pro dotyková zařízení a aplikace, které s existencí avatarů již od zárodků počítají (viz. kapitola 1).

Mimo jiné je také nutno počítat s tím, že výsledná aplikace, pro kterou je tento avatar navrhován je zaměřena na děti od předškolního věku pro něž je třeba při návrhu rozhraní brát speciální ohledy.

2.1.1 Obecné elementy rozhraní

Proces návrhu kvalitního uživatelského rozhraní je vědní obor sám o sobě a v jeho poli se pohybuje velké množství ostřílených expertů, jejichž názory se nemusí vždy shodovat. Co se však hlavního rozložení aplikačního interface týče, velké množství aplikací pro své potřeby používá časem prověřená a ustálená řešení se kterými se v nějaké formě lze setkat v téměř každém moderním software.
Hlavní rysy těchto ovládání můžeme shrnout do následujících bodů.

- Graphical User Interface (GUI)
  Tato vlastnost je pravděpodobně jasná a samozřejmá, přesto je však pro budoucí úvahy velice podstatná. Interakce s aplikací ovládanou pouze textovým vstupem (*CLI*) je velmi odlišná a návrh by v jejím případě šel jiným směrem. Většina moderních aplikací navržených pro běžné uživatele se vydává touto cestou.

- Dotykové zařízení
  Jelikož je výsledná aplikace určena pro mobilní zařízení, je nutné předpokládat vstupy uživatele zadávané pomocí doteků, gest a virtuální klávesnice přímo na displej zařízení. Tímto je také eliminována možnost použití některých možných vstupních akcí, jakou je například přejetí myši přes položku (mouseover).

- Hlavní menu
  Nalézající se na vstupní obrazovce aplikace ihned po jejím spuštění nebo vyvolané na přání uživatele, tato součást představuje křižovatku mezi jednotlivými součástmi aplikace.
  Nejběžnější provedení sestává z množiny tlačítek po jejichž stisknutí aplikace přejde do žádaného stavu.

- Položka nastavení
  Téměř každá alespoň trochu komplexní moderní aplikace obsahuje sekci nastavení, ve které lze upravit chování aplikace a jejích součástí.

- Přechod aplikace do/z pozadí
  Zejména na mobilních zařízeních se systémem Android se aplikace velice často dostane do stavu, kdy se momentálně neocitá na obrazovce zařízení, přesto je však v pozadí spuštěná, případně zapauzovaná. Takovouto změnu stavu lze v aplikace detekovat a zadefinovat požadované reakce.

Ovládací prostředky aplikací často obsahují velké množství dalších vlastností, s těmito se však lze setkat v téměř každém případě, proto je s nimi nutno počítat při samotném návrhu vstupních prvků avatara.
Další podstatným předpokladem je v tomto případě věková kategorie uživatelů. Jelikož aplikace Dráček je zaměřena na děti, je s tímto nutno počítat v návrhu a rozhraní patřičným způsobem přizpůsobit. Tímto problémem se na svém webu zabývá designer Luke Wroblewski v článku „*Touch-based App Design for Toddlers*“ [7]. Přestože se ve svém textu zaměřuje na návrh pro ty opravdu nejmenší, lze jeho postřehy bez problémů uplatnit také v aplikacích pro děti o trochu starší. Body popsané v článku jsou následující.

- Vyhnutí se častým načítacím obrazovkám
  Dlouhotrvající úvodní logo a video, načtení aplikace následované načtením hlavního menu po kterém dochází k načtení herní úrovně samotné. Čím déle je dítě nuceno na začátek hry čekat, tím větší je pravděpodobnost, že ho to omrzí a přesune se k jinému druhu zábavy. Stejné tvrzení platí o dlouhém či častém načítání dalších částí aplikace během hry samotné.

- Zamezení ostrého kontrastu mezi aplikací a menu
  Velké množství dnešních aplikací, obzvláště těch, které jsou cíleny na uživatele z řad dětí, volí pro své uživatelské prostředí barevně pestré kombinace interaktivních obrázků s jejichž pomocí lze dosáhnout požadovaných cílů. Pro uživatelský zážitek dítěte není poté nic nepříjemnějšího, nežli když stiskne jedno z krásných barevných tlačítek a na obrazovce aplikace se objeví klasické systémové menu v černé barvě.

- Přizpůsobení dotykových gest
  Široké spektrum možností interakce se zařízením pomocí kombinace doteků nemusí být v případě dětí, jejichž ruce jsou stále ještě malé a jejichž motorické schopnosti nemusí ještě být na úrovni dospělého uživatele, tou nejžádanější vlastností.
  Luke ve svém článku především zmiňuje reakci aplikace v případě, že se uživatel pokouší nějakou položku zvolit poklepáním, má však zároveň již na nějakém místě displej zařízení stisknutý (část dlaně se např. dotýká rohu obrazovky). Jako řešení proto doporučuje podporu více stisků najednou a odolnost proti takovýmto případům.

- Odolnost vůči několikanásobnému poklepání
  Dětské chování nemusí vždy splňovat logické předpoklady chování uživatele, proto je třeba při vývoji počítat s extrémními případy užívání. Jedním z takovýchto případů je ten, kdy dítě na jednu nebo více položek rapidně poklepá v rychlém sledu za sebou, např. 10-krát během jedné vteřiny. Pokud by aplikace proti tomuto nebyla správně ošetřena, způsobila by takováto akce opakované spuštění události, playbacku nebo

jiné reakce, jejíž proběhnutí je v dané chvíli zamýšleno jen jedenkrát. Pro zamezení překrývajících se zvuků a jiných, pro chod aplikace tragičtějších, situací je třeba poskytovat ochranu proti spuštění nové události před tím, než ta předchozí úspěšně dokončila svůj běh.

Mimo samostatné články které se tímto tématem zabývají, lze další informace nalézt v řadách vývojářů, kteří se již někdy věnovali tvorbě aplikace zaměřené především na mladé uživatele.
Jedním z doporučení, nalezených na internetových diskuzních fórech pro vývojáře, je vyvarování se použití zbytečně velkého množství interaktivních tlačítek na obrazovce v případech, kdy toto není opravdu nezbytné. Všudypřítomné odkazy na sociální sítě, které uživatele z aplikace kompletně vyvedou jsou v dnešním dni běžně provozovanou normou, pro tento typ aplikace jsou však zcela nevhodné. Stejné lze říci o velmi populárním trendu vkládání výdělečných reklam do obrazovek aplikace, které často zabírají podstatné procento celkové plochy a lze je při neopatrnosti náhodně stisknout.
Pokud je nějaké tlačítko na obrazovce mimo hlavní menu opravdu potřebné, je vhodné toto tlačítko zajistit jakousi obdobou dětské pojistky běžně používané na mnoha výrobcích. Jednou z fungujících metod pro toto používaných je požadavek dvojího stisknutí tlačítka s minimální odezvou, které je navíc graficky nevýrazné. To znamená, že pokud je tento prvek poprvé stisknutý, s aplikací to v podstatě nic neudělá. Feedback by v tomto případě měl být opravdu minimální, aby dítě nepobízel k další akci, tedy například nepatrné zvětšení tlačítka na dobu jedné vteřiny. Pokud je oblast během této krátké chvíle stlačena znovu, teprve v tu chvíli se vyvolá požadovaná akce.
V případě, že však přesto nějakým způsobem dojde k tomu, že je uživatel z aplikace přesměrován do systému, měla by mu být zajištěna možnost rychlého a plynulého návratu. Tomuto se věnuje další z tipů, který poukazuje na fakt, že pokud se dítě chce vrátit do nějakého předchozího stavu aplikace, například začít hru od znovu, častokrát namísto komplikované série zpětných tlačítek zvolí návrat do systému zařízení a znovuzapnutí aplikace. Aplikace by tedy měla být svižná a rychle reagující jak na normální tak extrémní případy zacházení, jelikož, ze všeho nejdůležitěji je třeba mít na paměti, že dětská trpělivost je velice omezená.
Vyhnutí se jakémukoli nepotřebnému zdržování či rozptýlení uživatele je tedy klíčové. Z pohledu avatara samotného lze tedy říci, že jeho animace by měly být rychlé a neblokující. Z technického hlediska je také třeba myslet na velikost zdrojových souborů a složitost kódu, které mohou aplikaci dodatečně zdržovat a tím jí tedy uškodit.
Shrneme ty tyto informace do bodů, získáváme následující doporučení.

- Vyvarování se použití nepotřebných tlačítek mimo hlavní obrazovku
- Dvojí stisk pro systémová tlačítka mimo menu

- Nevkládat odkazy na reklamy a sociální sítě
- Rychlé znovunačtení aplikace po náhodném vypnutí
- Jednoduchý návrat do minulých položek / ukončení aplikace
- Svižná a rychle reagující aplikace
- Neblokující animace

2.2 Aplikace Dráček

2.2.1 Původní verze aplikace v jazyce Java

Obrázek 2.1: Hlavní menu, hra skládání slabik

Při návrhu ovládacích prvků avatara se lze do jisté míry inspirovat předchůdcem navrhované aplikace, jež funguje na systému Microsoft Windows. Ovládací rozhraní, které aplikace poskytuje se velice podobá nespočtu dalších aplikací podobného stylu a lze předpokládat že se ve své podstatě nebude příliš lišit ani v budoucí verzi Dráčka. Jeho bližší rozbor tedy poskytuje vítané informace ohledně možností použitých pro vstupy postavičky.
Uživatel je ihned po spuštění prezentován s obrazovkou výběru svého herního profilu, po jehož zvolení se dostává do hlavního menu celé aplikace. V této části si poté volí z několika možností, mezi něž patří výběr ze seznamu dostupných výukových her, přehled dosažených výsledků a několik sekcí s informacemi (historie Smečenské školy, výukové poruchy, pověst o Smečenském draku). Již během těchto částí se naskytuje množství možností použitelných pro vstup avatara.

- Spuštění aplikace
- Zvolení profilu hráče
- Přechod do hlavního menu

- Prohlížení informačních sekcí
- Stav herních výsledků uživatele
- Přechod do seznamu dostupných mini her

Spektrum možností se po přechodu do seznamu mini her ještě podstatně rozšiřuje. Každá z těchto her má své unikátní chování a cíle, frekvence a typ poskytnutých vstupních prvků se tedy mezi nimi bude lišit. Obecně však lze říci, že se jedná jak o hry, u nichž dochází k průběžnému vyhodnocování výsledků a tak o ty, kde jsou hráčovi výsledky vyhodnoceny až na konci herního kola. Rozborem samotných her a možností, které poskytují se bude zabývat budoucí část textu.

2.2.2 Nová verze aplikace pro systém Android

Vývoj nové verze aplikace pro mobilní zařízení s sebou přináší řadu nových možností. Mimo odlišné chování systému samotného lze také uplatnit unikátní hardwarovou výbavu tabletů a chytrých telefonů, jež se na klasickém počítači nenachází.
Jednou z novinek, které vývojový tým aplikace zamýšlí uplatnit je použití tzv. metadat, pomocí nichž lze shromažďovat nejrůznější druhy informací. V současném návrhu se konkrétně jedná o

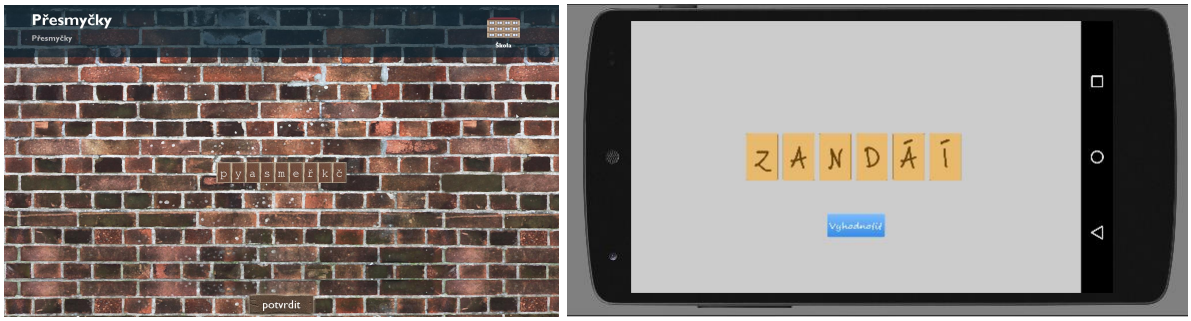
- Délka doby uplynulé od posledního přihlášení
- Podrobnější záznamy o výsledcích hráče

Mezi další diskutované změny patří implementace učitelského rozhraní aplikace, pomocí nějž by vyučující mohl provádět administrativní akce, jakými je například přidávání nového obsahu nebo prohlížení výsledků dětí.
Co se již zmiňovaných možností zařízení týče, lze pro naše účely uvažovat následující

- Přechod aplikace z pozadí do popředí
  Pokud se uživatel navrátí ze systému zařízení zpět do aplikace, která mezitím nebyla ukončena, ale pouze suspendována, lze na toto odpovědět reakcí postavičky.
- Použití akcelerometru zařízení
  Přestože kombinace drahých mobilních zařízení a dětí s nimi divoce mávajících nemusí být tou nejšťastnější volbou, je dobré uvažovat s možností využití specifičtějšího hardwaru těchto zařízení. Mimo akcelerometru lze použít i např. senzor osvětlení nebo jiné součásti hardwaru.

2.2.3 Modul Přesmyčky nové aplikace Dráček

Ve chvíli psaní této práce je z androidí verze aplikace Dráček hotov pouze modul Přesmyčky [8], založený na stejnojmenné součásti původní aplikace. Tento modul byl vyvinut čtyřčlenným týmem studentů v rámci předmětu BI-SP1 [9], Michal Bureš, Patrik Pavelec, Ondřej Filip a Miroslav Mazel, kteří se poté rozhodli rozsah zadání rozšířit na své bakalářské práce a ve vývoji aplikace pokračovat.



Obrázek 2.2: Původní verze modulu (nalevo), nová verze modulu (napravo)

Současný prototyp se, na rozdíl od původní verze, nachází ve formě samostatné aplikace, která po spuštěné nabízí možnost nové hry a nastavení. Grafické zpracování je vzhledem k fázi vývoje v současné chvíli minimální.
Položka nastavení nabízí uživateli volbu obtížnosti (lehká, střední, těžká) a délky slov (minimální a maximální počet znaků). Po spuštění nové hry jsou tyto nastavení uplatněna a je vybráno jedno z vhodných zadání.
Nabídnutá písmena může uživatel libovolně přesouvat na pozice ve slově, dokud není s výsledkem spokojený. Stisknutím tlačítka vyhodnotit se poté provede ověření výsledku a písmenka na správné pozici jsou uzamčena a označena modrou barvou. Nesprávně umístěná písmena může uživatel znovu nadále přesouvat a pokusit se o splnění znovu.
Možnosti výsledku jsou tedy tři

- Žádné písmeno není na správné pozici, vše je špatně
- Část z písmenek je umístěna správně
- Celé slovo je správně

Jakmile jsou všechna písmenka správně umístěna a odpověď je vyhodnocena, aplikace vygeneruje zadání nové. Po úspěšném odevzdání obtížností definovaného množství slov je cvičení ukončeno.

2.3 Typy vstupních akcí

Uživatel běžně během svého působení v aplikaci vykonává množství akcí, jejichž účelem bývá ovlivnění stavu aplikace samotné nebo vkládání nějakého druhu vstupních dat.
Z těchto vstupů je třeba vybrat takové, které lze výhodně použít pro zajištění reakcí postavy tak, aby zaručovali dostatečnou výstižnost dané situace a případně podněcovali k dalším reakcím uživatele.
Na základě analýzy grafického uživatelského rozhraní a obou verzí aplikace Dráček v této kapitole provedené lze tyto vstupní informační kanály rozdělit na dvě hlavní skupiny.

- **Akce, které *zaručují přímou reakci* avatara**
  Do této skupiny spadají vstupy takové, na které lze zareagovat přímým a okamžitým způsobem. Pokud tedy uživatel v tomto případě použije daný prvek aplikace, ihned se mu od postavičky dostane vhodně zvolené animace nebo jiných příhodných reakcí.
  Jako příklad lze uvést postavičku psa Floyda uvedeného v kapitole 1.2.3. V několika z mini her, ve kterých se Floyd nachází, jsou uživateli představeny volby (odpovědi na úkol či otázku) z nichž ne všechny jsou správné. V závislosti na výběru uživatele z těchto možností je poté spuštěna animace buďto zklamaného zamračení, v případě špatné odpovědi, nebo šťastného přikývnutí, v případě správné.

- **Akce, které reakce avatara ovlivňují *nepřímým způsobem***
  Správná definice akcí patřící do této kategorie je již podstatně složitější než v případě předchozím. Jedná se o akce, jejichž provedení samo o sobě buďto nezaručuje žádnou výstupní reakci nebo se jedná o reakce zpožděné v závislosti na tom co uživatel udělal, případně neudělal.
  Jedním z možných příkladů může být reakce postavičky na dlouhodobou neaktivitu uživatele. Pokud po delší, předem specifikovanou, dobu nedojde k žádnému typu interakce s aplikací, spustí se kupříkladu animace, ve které postavička upadne do spánku nebo provede nějakou jinou tzv. idle akci.
  Do této skupiny můžeme také zařadit vstupy, které spustí reakci pouze v případě splnění určitých předpokladů. Pro příklad můžeme uvést postavičku Toma (kapitola 1.2.4), kde uživatelovi akce postupně naplňují zkušenostní ukazatel. Akce provedená za normálních okolností by způsobila pouze svou přímou reakci, pokud se však tento ukazatel jejím provedením naplnil na požadovanou hodnotu, Tom postoupí o úroveň výše což mohou doprovázet doprovodné animace, např. růst postavičky.

K získání ještě lepší představy o formě těchto vstupních událostí se lze blíže podívat na zástupce výukových her nalezených v originální aplikaci Dráček.

Další užitečné informace lze poté nalézt také v zástupcích již testovaných postaviček, tedy v jejich domovských aplikacích.

2.4 Rozbor některých mini her z původní aplikace

Část původní aplikace zvaná Škola obsahuje otočný seznam s ikonami příslušných her. Tyto hry a jejich obsah poskytují hlavní a nejdůležitější zdroj pro ovládání reakcí avatara, je tedy nutné se na jejich průběh blíže podívat a identifikovat jejich klíčové události. Mezi tyto hry patří i hra Přesmyčky, která byla v jedné z předchozích částí (2.2.3) blíže rozebrána. Zde se stručněji podíváme na některé z dalších exemplářů.

2.4.1 Doplňování do textu

Tato hra se vyskytuje v několika variantách, jednou z nichž je doplňování ě/je do předem určených míst ve slovech. Volbu lze v průběhu kola libovolně měnit, výsledek je určen až po stisknutí závěrečného tlačítka, čímž se zobrazí výsledek a ukončí hra. Na stejném principu funguje i varianta s určováním podmětu a přísudku ve větách. V těchto verzích je pro reakci vhodná pouze část se závěrečným vyhodnocením.
Naopak ve variantě s doplněním s/z je kontrola správnosti slova provedena okamžitě s případnou korekcí a penalizací. Rozšiřující možností je zde tedy

- Okamžitá kontrola po zvolení možnosti

2.4.2 Vybarvování

Na začátku tohoto modulu je hráči předložena paleta barev, linkovaný obrázek a ručně psaný seznam úkolů. Cílem dítěte je pomocí poskytnutých barev a instrukcí vybarvit pouze příslušné části obrázku správně zvolenou barvou.
Vyhodnocení hry je provedeno až po stisknutí příslušného tlačítka, platí zde tedy stejné zásady reakcí jako v hrách předchozích, mimo to se však naskytuje možnost reakce avatara na hráčovu interakci s prostředím, tedy zvolení barvy z poskytnuté palety a následné vybarvení části obrázku. Tam kde se dříve probrané vstupy zabývali sdělením nějaké informace o stavu, zde by cílem bylo zvýšení zábavnosti plnění a zlepšené udržení pozornosti dítěte.

- Doprovodná animace při zvolení barvy
- Doprovodná animace při vybarvení obrázku

Zbylé z her jsou sice obsahově odlišné, z hlediska možného ovládání postavičky však již nenabízí nic podstatného.

2.5 Ohlédnutí na aplikace analyzovaných avatarů

Jednou z možností nalezení prostředků použitelných pro ovládání animované postavičky je bližší pohled na to, jak to dělají již zaběhnuté aplikace. Pro tento účel se lze podívat na dříve zvolené zástupce, zejména ty co byly v testech nejúspěšnější a jsou mezi lidmi nejvíce populární.

2.5.1 My Talking Tom

Jelikož je celá existence této aplikace založena na úspěšnosti avatara, je nezbytné, aby pro tento účel poskytovala dostatečně rozmanité ovládací prostředky. Jaké situace tedy zaručují získání nějaké reakce od postavičky?

- Stisknutí tlačítka pro uspokojení potřeb postavičky
  Nejjednodušší ze způsobů starání se o Toma obnáší pouhé stisknutí příslušného tlačítka a sledování jeho reakce (např. spánek, wc).
- Přímý dotek avatara
  Postava samotná je rozdělena na několik interaktivních částí, každá z nichž přináší po doteku jiný způsob reakce (podražení nohou, šťouchnutí to hlavy).
- Typ doteku postavičky
  Výše zmíněná reakce je také závislá na tom, jak často byl dotek v krátkém časovém intervalu zopakován a jaký byl jeho typ. V případě, že uživatel Tomovi několikrát přejede prstem po břiše tam a zpět, dostane se mu reakce spokojeného vrnění. Pokud však stejnou akci provede pouhým ťuknutím, postavička na to reaguje animací zcela odlišnou. V případě jednoho ťuknutí do hlavy je Tom pouze zmatený, pokud se toto však rapidně opakuje, spadne z toho zem.
- Interakce s jiným objektem na obrazovce
  Tom poskytuje reakce i v případě, že s ním akce uživatele zrovna přímo nesouvisí. Pokud ten například poklepe na předmět či zem, postava očima přejede na zasažené místo a sleduje co se stalo. Podobný efekt nastává i pokud se uživatel přiblíží s ikonkou jídla poblíž Tomovi pusy, jenž předmět celou dobu sleduje očima a v případě přiblížení ho sní.
- Změna oblečení postavy
  Během procházení možností oblečení postavičky a dalších možných přizpůsobení na to ona sama reaguje animacemi ve kterých se prohlíží a poskytuje reakce o tom, jak se jí daný předmět zamlouvá.
- Stav ukazatelů potřeb
  Pokud se některý nebo více z indikátorů Tomovi spokojenosti nachází na nedostatečné hladině, je toto uživateli zdůrazněno pomocí příslušných opakujících se animací.

- Sledování pohybu prstu po obrazovce zařízení

- Reakce na mluvení uživatele
  Schopnost opakovat to, co mu uživatel poví je pravděpodobně jedna z Tomových nejznámějších vlastností. Reakce na toto zahrnuje animaci pro indikaci zahájení naslouchání, která se spustí ihned při zaznamenání hlasového projevu.
  Další animace jsou poté připraveny pro samotné mluvení po ukončení nahrávání.

- Idle animace
  Pokud uživatel delší dobu neprovedl žádnou akci, postavička na to zareaguje zamáváním.

2.5.2 Microsoft Office

Svým způsobem zaměření odlišný od Toma, lze u tohoto pomocníka nalézt několik dalších reakčních prvků, které se v předchozím případě nemusí vyskytovat.

- Rozpoznání vzorce chování uživatele
  Aplikace v tomto případě průběžně sleduje chování uživatele při jejím používání a pokud zaregistruje známý vzorec chování, zavolá asistenta pro poskytnutí pomoci.

- Provedení akce přes asistenta
  V případě, že je některá z předdefinovaných akcí provedena pomocí user agenta, namísto klasického postupu, je spuštěna doprovodná animace, např. odesílání pošty.

- Provádění činnosti v aplikaci
  Použití některých součástí aplikace, jako např. položky ovládacího panelu může přivolat postavičku poskytující tipy k danému nástroji.

2.5.3 Education.com

Zástupce postaviček výukových her, Floyd se vyskytuje v několika takovýchto aplikacích, jejichž délka je obvykle velmi krátká, jejich průběh však poskytuje další užitečné informace.

- Zapnutí/vypnutí aplikace
  Po dokončení načtení aplikace je přehrána animace přibíhajícího Floyda, okolo nějž se poté rozloží herní interface. Podobná reakce se stane těsně před tím, než se aplikace ukončí.

- Volba možnosti a její správnost
  Při hraní je poskytována zpětná odezva v několika formách, jedna z nichž je animace Floyda vyjadřující správnost volby. Tato reakce je spuštěna ihned po výběru odpovědi na otázku.

- Doprovodné animace mezi úkoly
  Pro pomoc přechodu mezi jednotlivými otázkami je někdy použita rychlá doprovodná animace avatara. Pro příklad lze uvést hru Counting in the Kitchen s počítáním pokrmů, kde Floyd vždy po správném výběru navaří nové porce. Tato akce je rychlá a trvá pouze kolem jedné vteřiny.

Tento rozbor však stále ani zdaleka nevystihuje všechny možné události, které lze pro potřeby postavičky použít. Následující kapitola se proto zabývá některými z exotičtějších možností.

2.6 Úvaha dalších možností vstupních akcí

V dosavadních částech textu bylo nalezeno velké množství použitelných technik pro ovládání postavičky avatara. Tyto však stále ještě nevystihují všechny příležitosti, které se vývojáři nabízí, jelikož počet těchto akcí není žádným způsobem limitovaný a záleží pouze na schopnostech a kreativitě vývojáře.
Z informací získaných v rozborech a úvaze schopností zařízení lze jako zástupce dalších použitelných možností uvést např.

- Kalendářní datum
  Specifické kalendářní události jako jsou svátky mohou být uvažovány za vhodný vstup pro reakci avatara na daný den (např. vánoce).

- Načítací obrazovky
  Jak již bylo zmíněno v analýze rozhraní, dětská trpělivost má své hranice a dlouhé či časté načítací pauzy mohou být odrazujícím faktorem. Tento problém lze alespoň z části omezit přípravou vhodné animace pro zabavení uživatele v průběhu nahrávání aplikace.

- Věk a pohlaví hráče
  Reakce na určité situace lze obměnit v závislosti na stáří a pohlaví hráče, jelikož ne všechny animace mohou být všeobecně vhodné.

- Reakce na gesta
  Komplexnější gesta (kupříkladu gesto pinch, běžně používané pro přiblížení) mohou také najít své uplatnění pro spuštění dalšího feedbacku.

- Rychlost uživatelových reakcí
  Pokud se hráč potýká se hrou, jež je zaměřena na testování rychlosti rozhodnutí a reflexů lze frekvenci správných odpovědí použít jako informaci pro provedení vhodné animace.

- Listování v seznamu mini her
  Dle toho, jakým způsobem je takovýto seznam provedený může být možné zajistit doprovodné animace dle stylu hry, který je na obrazovce zrovna vybraný.

- Počasí a teplota okolí
  Postava může být schopna spouštět krátké reakce dle toho, jaké je zrovna v okolí uživatele počasí.

- Rozšíření herní databáze
  Obsah velkého množství aplikací není fixní a může být v průběhu života aplikace dále rozšiřován o nové moduly a jiná vylepšení. Na průběh takovéto změny lze upozornit vhodnou reakcí postavičky.

2.7 Souhrn možných událostí a technická stránka věci

Předchozí části této kapitoly se věnovaly širokému pokrytí a dedukci možných vstupních akcí, které by nějakým způsobem mohly být použity k ovládání animovaného pomocníka. Informace získané z těchto rozborů mohou nakonec být shrnuty do jednoho seznamu, ve kterém lze navíc diskutovat způsob technického provedení těchto možností.

- Spuštění aplikace
  Reakce či animace postavy ve chvíli, kdy se aplikace plně načte. Vývojové nástroje platformy Android obyčejně obsahují nějakou variantu metody „onCreate“ volané právě ve chvíli, kdy je aplikace spuštěna.

- Výběr profilu hráče
  Obrazovka volby profilu jako taková je vhodná pro různé typy animací. V této chvíli lze reagovat buď přímo na stav aplikace, tedy „aplikace je na obrazovce volby profilu“, nebo na samotnou volbu tohoto profilu (na základě osobních nastavení, úspěchů atd.) Jedním z údajů použitelných pro další rozhodování o zvolené animaci může být také například počet dnů od posledního přihlášení uživatele. Tyto údaje lze reprezentovat pomocí proměnné int nebo položky data.

- Změna v hlavním menu
  Výběr některé z položek jako jsou Nová Hra nebo jakékoli jiné akce provedené v hlavním menu jsou dobrým způsobem vyvolání reakce postavičky. Tyto reakce lze jednoduše vyhodnotit zavoláním příslušných funkcí při stisknutí tlačítka či interaktivního objektu tuto nabídku reprezentující.

- Stisknutí speciální položky
  Zvláštní druhy tlačítek či ovládacích prvků (např. posuvník) které při

změně vyvolají požadovanou akci. Opět lze implementovat jednoduše pomocí listenerů [10].

- Změna stavu aplikace
  Přechod aplikace z jedné obrazovky na druhou. Tato akce může vyvolat reakci jak při opuštění tohoto stavu, tak při vstupu do dalšího. Lze nastavit volání funkce do příslušné části kódu.

- Načítací obrazovka
  Doprovodné animace během načítání aplikace či jejích částí. Tyto animace mohou být spuštěny provedením nezbytných volání při ukázání této obrazovky (tj. funkce se zavolá v části kódu za načítání zodpovědné).

- Přizpůsobení postavy
  Změna v nastavení avatara, ať již jeho chování nebo nastavení vzhledu. Reakční animace se poté týkají položky, která byla právě změněna (opět například přes tlačítko či ID objektu).

- Neaktivita uživatele
  Pokud uživatel po stanovenou dobu neprovedl žádnou akci, postavička na to může zareagovat zvolenou animací. Tyto animace mohou být různé či náhodné v závislosti na času. Uplynulý čas lze sledovat implementací časovače [11], který začne odpočítávat pokaždé, když uživatel skončí nějakou akci. Pokud tento časovač dosáhne zadaného limitu, provede příslušná volání animace.
  Podobné techniky lze použít také pro měření rychlostí reakcí uživatele a prodlevy mezi akcemi uvnitř mini her a přizpůsobit výsledné reakce těmto údajům.

- Přechod do / ze systému
  Zejména v případě, že aplikace byla spuštěna v pozadí a její průběh je obnoven je možné zareagovat vhodnou animací. Tato změna stavu aplikace je obvykle detekovatelná funkcemi typu „onResume“, které se ve vývojových nástrojích nacházejí.

- Dotyková gesta
  Detekce typu gesta a reakce dle vyhodnocení. Například lze implementovat gesto „otočení“, rozpoznat jeho provedení, rychlost a úhel a provést příslušné změny s natočením hlavy avatara. Podobné akce lze provést například pro gesta „přiblížení“ či „cvrnknutí“. Nástroje pro detekci těchto gest jsou v systému dostupné [12].

- Poklepání na obrazovku
  Jedno či vícenásobné poklepání na obrazovku zařízení lze jednoduše vyhodnotit a použít ke spuštění vhodné akce.

- Dotknutí se avatara
  Přímá interakce s postavičkou. Podobné gestům či poklepání (zde opět možné) avšak v prostoru přítomnosti avatara. Při detekci kolize doteku a postavičky lze provést vyhodnocení typu doteku a zvolení vhodné animace. Graf scény scene2d [13] umí tyto kolize detekovat.

- Speciální sekce aplikace
  Části aplikace které vedou ke spuštění specifických animací k této části určené. Postavička může například reagovat na přítomnost seznamu her nebo jiných údajů.

- Celkový stav úspěchů uživatele
  V případě ukládání podrobných údajů výsledků hráče v celém či omezeně určeném časovém období lze tyto údaje využít k výběru vhodné reakce a jejím spuštění kdykoli v průběhu aplikace. Toto lze reprezentovat například průměrem všech výsledků v proměnné int.

- Výsledky poslední hry
  Stejné jako předchozí případ, svým výskytem však lokálnější. Lze použít například při vybrání stejného typu hry. Výsledek ve formátu int, 0 - 100%.

- Výsledky poslední akce
  Akce provedené přímo ve hrách, ať již detailní (vyhodnocení po každé změně) či celkové (na konci hry). K tomuto vyhodnocení lze použít vše od jednoduchých bool hodnot (true / false) po procentuální úspěch či uplynulého času.

- Využití hardwaru zařízení
  Hardwarové možnosti specifické v mobilním zařízením, například akcelerometr. Údaje z těchto součástí mohou být použity k vyvolání vhodné reakce.

- Doprovodné animace k akcím uvnitř mini hry
  Ke hře specifické animace závislé na akcích uživatele uvnitř této hry. Pokud například ve hře malování uživatel namočí štětec do barvy je možné, aby postavička udělala akci stejnou. Implementace pomocí onClick / onChange listeneru.

- Interakce s objektem
  Možné interaktivní objekty v pozadí aplikace, s nízkým či žádným významem určeny pouze pro zabavení dítěte. Detekce onClick listenerem či true / false kontrolou stavu.

- Způsob provedení akce
  Postava může na různé styly provedení stejné akce odpovídat jiným způ-

sobem (například spuštění hry dvěma možnými způsoby). Provedení voláním odlišných funkcí.

- Vnitřní stav aplikace
  Data, která uživatel nemá schopnost přímo ovlivnit. Může se jednat například o ukazatele stavu říkající že postava je unavená. Tyto údaje mají číselnou (např. procentuální) formu a lze na ně reagovat kontrolou překročení nastavených limitů.

- Zachycení / analýza řeči
  Avatar teoreticky může být schopen rozpoznat řeč a provést na ni vhodným způsobem vybranou odpověď. I pokud význam zvuku rozpoznán není, je stále možné znázornit jeho zachycení.

- Detekce vzoru chování
  Mnoho různých vzorů a stylů chování lze zachytit a zareagovat. Jako příklad lze uvést určitou posloupnost akcí, opakující se situace nebo detekci záměru uživatele.

- Výplňové animace
  U her skládající se z více částí nebo v jiných vhodných místech lze zajistit vyšší zajímavost aplikace využitím vhodné animace pro přechod části či časové prodlevy mezi úkoly. Tyto reakce lze zavolat při provedení specifické akce (konec úkolu, restartování stavu atd.)

- Kalendářní datum
  Svátek, významný den, narozeniny uživatele a mnoho dalších akcí lze vyvolat při detekci vhodného data dne pomocí informací zařízení a v aplikaci uložených informací. Implementace položkou Date nebo celočíselně (int).

- Osobní data uživatele
  Údaje jakými jsou věk, výška, pohlaví a jiné mohou hrát roli v nastavení vzhledu a chování postavičky a to buď pomocí přednastavených profilů (kluk / dívka, pod / nad určitou věkovou hranici...) nebo údajů samotných (výška avatara závislá na výšce či věku uživatele).

- Rozšíření aplikace
  Podstatné změny v aplikaci například při její aktualizaci lze zdůraznit zavoláním speciální animace (například dodané s aktualizací) na tyto změny upozorňující. Informaci o nutnosti provedení této reakce lze zajistit pomocí jednoduché bool proměnné (novinky - true / false).

Kapitola 3

Možnosti výstupních reakcí postavičky

Jakmile je ujasněno jaké části aplikace lze pro ovládání avatara použít, je nutné rozebrat, jakým způsobem by postavička měla na jednotlivé podměty odpovídat.
Typy vhodných výstupných animací opět není možné určit nějakým přesným, vědeckým způsobem a je proto se nejprve zamyslet nad tím jaký je typ cílových uživatelů a nad jejich psychologickým profilem. K tomuto účelu kapitola obsahuje rozsáhlou část věnující se nástrahám dětské mysli, způsobu jakým ji lze nejlépe motivovat a pokud možno napomoci zájmu o užívání celé aplikace. Druhá část kapitoly je poté již techničtějšího typu a jsou v ní rozebírány možnosti zachycení a zpracování vstupních událostí se závěrečným seznamem možných animací pro tyto události.

Nejdříve se tedy pojďme podívat na psychologickou stránku věci.

3.1 Chování a motivace dítěte

Návrh správných reakčních animací postavičky není jen podstatným prvkem pro úspěch modulu samotného, ale může mít také klíčový vliv na popularitu celé aplikace. Avatar může při správném provedení vystupovat jako motivační, odměňující faktor díky němuž se dítě do hry opětovně vrátí.
Pro toto, stejně jako pro předejití možných nástrah dětské mysli je třeba nejprve pochopit alespoň základní vzorce dětského chování a časem prokázaná pozorování oborových expertů.
Prvním z důležitých kroků pro pochopení myšlenkového pochodu uživatelů, a to nejen těch nejmladších, je přijetí faktu, že většinová část dětí nebude projevovat nadšení vůči jakýmkoli výukovým prostředkům, které jim byť jen z části připomínají školu a s tím spojené nepříjemné povinnosti. Tento efekt

bohužel u aplikace, jejímž cílem je prohloubení intelektuálních znalostí dětí, nelze nijak obejít, použitím správným technik se ho však lze pokusit minimalizovat.
Pro toto je třeba blíže pochopit co dokáže dítě motivovat a jemnou formou ho přimět k tomu, co by za normálních okolností ve svém volném čase nikdy nedělalo.

3.1.1 Druhy motivace

Motivaci k nějaké činnosti lze dle psychologů rozdělit do dvou odlišných kategorií, lišících se nejen v tom, proč něco děláme, ale také jaké z toho máme potěšení. Tyto kategorie jsou

- Vnější motivace
  Zde spadají činnosti, které si člověk z vlastní vůle přeje vykonávat, přesto se k nim vždy vážou nějaké další externí důvody. Osoba se k těmto aktivitám ubírá, protože jsou pro ni nějakým způsobem užitečné, jejich výkon jí však nemusí přinášet žádné osobní potěšení.
  Jako dobrý příklad vnější motivace lze považovat studenta, který zdokonaluje své znalosti v daném oboru za účelem vyhnutí se špatné známce.

- Vnitřní motivace
  Do kategorie vnitřní motivace spadají činnosti více osobní, ty které si člověk přeje vykonávat bez jakýchkoli vnějších důvodů, čistě pro své vlastní potěšení. Specifické aktivity jsou velice individuální a nelze je univerzálně rozřadit, existují však kroky, které lze podniknout pro zvýšení pravděpodobnosti vyvolání tohoto druhu motivace. Autoři Coon a Mitterer [14] ve své knize používají takovouto definici „*Intrinsic motivation occurs when we act without any obvious external rewards. We simply enjoy an activity or see it as an opportunity to explore, learn, and actualize our potentials.*“
  Jako příklad může posloužit učení se hraní na hudební nástroj pouze z lásky k hudbě a touze umění ovládnout.

Z charakteristik vyplývá, že přestože je vnější motivace velmi užitečnou vlastností, její udržení vyžaduje značné přesvědčení a sílu vůle. Pro naše účely by nejlépe posloužil případ, kdy má dítě silnou osobní motivaci činnost vykonávat, aniž by ho do toho nutila třetí strana.
Takovéhoto efektu se lze pokusit dosáhnout zaměřením se na emoční stránku uživatele či představením nestresující výzvy, kterou by rád pokořil. A právě zde vchází do obrazu postavička dráčka, jejíž účel je přesně takový. Jak by se tedy avatar měl chovat, aby jeho akce vyvolali dostatečně silnou pozitivní reakci v mysli dítěte, jenž svou přítomností překryje nepříjemné vzpomínky na poslední domácí úkol z daného tématu mini hry?

3.1.2 Nástrahy odměn

Jednou z očividných technik jak člověka přimět k něčemu nepříjemnému obyčejně bývá odměna. Tohoto lidstvo již dlouhou dobu využívá a své potomky za složité činnosti také patřičně odměňuje. Je tato činnost však opravdu přínosná?
Tímto problémem se v roce 1973 zabývali psychologové Mark R. Lepper a David Greene ze Stanfordské a Michiganské univerzity. Pro svůj experiment shromáždili tři skupiny dětí předškolního věku, z nichž všechny rády kreslili. Tyto skupiny byly od sebe oddělené a jedné z nich bylo řečeno, že po nich bude něco požadováno a na konci dostanou odměnu. Druhé a třetí skupině o odměně nebylo zmíněno nic, jedna z nich jí však jako překvapení také měla obdržet. Třetí skupina tedy zůstala naprosto bez odměny. Tyto děti byly poté požádány o nakreslení obrázku a v příslušných případech dostaly své odměny, zdobené diplomy s mašlí.
Následné pozorování chování těchto dětí odhalilo, že ti, kterým byla od počátku přislíbena odměna projevili zhruba padesáti procentní pokles motivace k dalšímu samostatnému kreslení v porovnání se zbývajícími dvěma skupinami. Závěr tohoto experimentu ustanovil, že pokud je člověk za činnost, kterou rád sám od sebe vykonává, nějakým způsobem znatelně odměňován, začne činnost vnímat jako takovou, která k provádění potřebuje motivaci ve formě odměn, tedy je sama o sobě nepříjemná.

3.1.3 Emoce a paměť

Spojení mezi pamětí a emocemi nemusí být na první pohled patrné, přesto se však aplikuje na každého z nás. Životní události při nichž člověk zažívá zvýšené emoce nějakého druhu bývají velmi dobře zaznamenány do paměti takovým způsobem, že si je někdy lze živě vybavit i o mnoho let později. Tento jev ve svém článku rozepisuje doktor Rick Nauert [15], který tvrdí, že za ním stojí část mozku zvaná amygdala.
Tato část, zodpovědná za určování motivační a emocionální důležitosti věcí je více aktivní při pohledu na emocionálně zajímavé obrázky. Při výzkumu tímto se zabývajícím bylo zjištěno, že věci, které jsou pro nás emočně zajímavé vnímáme jasněji než jiné. Na intenzitě tohoto vnímání lze poté odhadovat sílu vzpomínek okamžiku se týkajících.
Z informací lze dedukovat, že vyvolání pozitivních emocí u dítěte pomocí roztomilé postavičky a jejích reakcí může mít pozitivní vliv na to, jak silně si dítě v paměti uchová to, co se v aplikaci naučilo.

Za další informace spojené s pamětí, zde konkrétně dětskou, lze vzít článek z deníku Psychological Science, který na svém webu rozebírá Dr Jeremy Dean [16]. V tomto článku se autor zabývá výzkumem, při němž byla zjištěna návaznost mezi informacemi, které dětská mysl přijímá do paměti a časovou

prodlevou.
V experimentech bylo zjištěno, že pokud se dítěti v krátkém časovém období podají dvě podobné informace, tyto informace se navzájem vyeliminují a dítě zapomene. Pokud je však mezi podáním podobného druhu informací uvedena časová prodleva, pochopení dítěte se zlepšuje, protože mezitím mělo čas zpracovat část první.

Jedním z kritických bodů pro vyhodnocení správné reakce je poté také způsob jakým dítě samotný obraz vnímá. Tomuto se věnuje následující podsekce.

3.1.4 Vizuální percepce

Obrázek 3.1: Delfín, www.eyecanlearn.com [17]

Vizuální percepce vyjadřuje schopnost mozku zpracovat to co vidí oči. Dobrá úroveň této schopnosti je podstatná pro velké množství každodenních činností jakými jsou psaní, čtení, řešení hlavolamů či oblékání se. Dítě jehož vizuální percepce není dostatečně rozvinutá může mít například problém s rozlišením mezi písmeny „b“ a „d“. Nízká úroveň nebo absence této schopnosti může tedy mít negativní dopad na život dítěte a to nejen ve stránce jeho sebevědomí.
Aplikace Dráček, mimo jiné, obsahuje součásti, které se touto problematikou zabývají, tj. mini hry soustředěné na rozeznávání obrázku nebo věcí, které nevyhovují určitému vzoru. Potíže se správně vyvinutou percepcí mohou být mimo to indikovány také věcmi jako jsou obtíže s určením začátku stránky, rozlišení pravé strany od levé nebo koncept prostoru jako takový (venku/uvnitř). Mezi další indikátory patří také potíže se soustředěním se na jedno slovo na stránce a snížené organizační schopnosti.
Vizuální percepce může být rozdělena do několika klíčových skupin [18]

- Sensory Processing
  Schopnost přesného zpracování a reakce na smyslové vjemy
- Visual Attention
  Schopnost vizuálního soustředění se na důležité elementy okolí a odfiltrování těch nepodstatných

- Visual Discrimination
  Schopnost rozlišit rozdíly v objektech založené na tvaru, barvě, velikosti atd.

- Visual Memory
  Vzpomínky na vizuální vlastnosti objektů

- Visual Spatial Relationships
  Pochopení vztahů objektů a prostředí

- Visual Sequential-Memory
  Schopnost vzpomenout si na sekvenci objektů ve správném pořadí

- Visual Figure Ground
  Rozpoznání objektů ve složitějším pozadí

- Visual Form Constancy
  Rozpoznání stejného tvaru i při změně jeho velikosti nebo rotace

- Visual Closure
  Schopnost rozpoznat objekt, který není úplný (viz. obrázek 3.1)

Nedostatečné schopnosti v některé z těchto oblastí mohou mít velice negativní následky jak na normální tak akademický život dítěte. Naštěstí však existují prostředky, pomocí kterých se tyto schopnosti dá pokusit zdokonalit. Za jednu z možností se dají považovat vizuální hlavolamy. Hříčky s tvořením obrázku spojováním teček, dokončování rozdělaných tvarů, detekce skrytých obrázkových vzorů a další vizuální hádanky k tomuto rozvoji napomáhají. Navrhnutí takových her je navíc povětšinou velice jednoduché, jelikož inspirace ve světě existuje nepřeberné množství. Jejich přítomnost v aplikaci vede nejen ke zdokonalování vjemových schopností, ale také může detekovat problémy dítěte v dané oblasti.
Mimo samotná cvičení, vykonávaná dítětem existují také kroky, kterých se může ujmout dospělá osoba a tím napomoci jeho orientaci.

- Směrové šipky
  Problémy se směrovou orientací lze vyřešit přidáním orientačních šipek v požadovaném směru

- Vizuální nápovědy
  Použití grafické či barevné značky ke zvýraznění začátku textu a dalších problémových lokací

- Mřížkovaný papír
  Psaní na papír se čtverečky napomáhá udržet velikost a vzdálenost písmen

- Zvýraznění hranic a linek
  Rozpoznání hraničních oblastí může pro dítě s touto poruchou být obtížným úkolem, jejich zvýraznění s tímto dokáže pomoci

- Předlohy písmen a číslic
  Pro ověření správného tvaru a orientace písmene

- Eliminace rozptýlení
  Rozptylující elementy jakými jsou blízká okna (ve třídě) či nepotřebné tabulky a tlačítka (aplikace) mohou ničit koncentraci dítěte a narušovat jeho schopnosti správného splnění úlohy

- Jasná a jednoduchá cvičení
  Pochopení cílů a zadání by mělo být rychlé a bezproblémové

Všechny tyto body se dají použít jako doporučení pro účely grafického rozhraní aplikace, designu mini her i avatara samotného, je tedy dobré na ně při návrhu pamatovat.

3.1.5 Další doporučení

Kromě již zmíněných doporučení pro udržení pozornosti a zvýšení motivace a zábavnosti se dá použít ještě mnoho dalších metod. Je však třeba brát ohled na to, že lidská psychologie, obzvláště dětská, je komplexní obor plný neznáma a každý člověk je unikátní, nelze tedy vždy jednoznačně určit co je přínosné a co negativní.
Mezi relativně neutrální techniky, jejichž použití by s sebou nemělo nést velká rizika, patří neustálé povzbuzování (odměřené, ne otravující). Toho lze za pomoci avatara dosáhnout použitím motivujících a odměňujících animací. Přímo navazující je téma zdůrazňování silných stránek dítěte. To mu dopomáhá cítit zadostiučinění z dobře odvedené práce a může motivovat ke zvýšení snahy k dosažení dalších úspěchů. Případné zdůraznění slabin dítěte má efekt zcela opačný a je třeba se mu vyvarovat.

Tímto lze analýzu dětské mysli uzavřít a informace zde získané použít jako jeden z faktorů pro návrh vhodných animací výsledné postavičky. Nyní se již můžeme vrátit k částem technickým.

3.2 Způsoby předávání požadavků aplikace

V této chvíli již víme, v jakých případech může aplikace po avatarovi požadovat reakci, podstatné však také je, jak se tato informace k postavičce vůbec dostane, tedy jakým způsobem aplikace o provedení reakce požádá.
Tohoto lze dosáhnout několika způsoby, jejichž podstatu a individuální výhody si v této kapitole blíže rozebereme.

3.2.1 Synchronní vysílání požadavků

V tomto případě se ovládání postavičky provádí pomocí předem nadefinovaného ovládacího rozhraní modulu. Tento modul v sobě obsahuje metody, jenž v okamžiku zavolání spustí požadovanou animaci.
Pokud je tedy v nějakém případě či stavu aplikace požadována reakce postavičky, do příslušné části kódu je vloženo volání metody jenž tuto reakci zajišťuje.
Jako příklad si vezměme celkové vyhodnocení výsledků nějaké z mini her. Ve chvíli, kdy uživatel stiskne příslušné tlačítko, je provedena část kódu, jenž zjistí jakým způsobem si uživatel vedl. Tato informace je poté použita pro zvolení vhodné doprovodné animace a ta je zavolána.
Za hlavní charakteristiky této metody lze považovat

Klady

- Okamžitá reakce postavičky
  Za předpokladu, že ve chvíli zavolání neprobíhá žádná další animace, jejíž přerušení není žádané, je reakce avatara na zavolání okamžitá a chování předvídatelné.

- Vysoká kontrola nad chováním
  V danou chvíli se provede opravdu ta animace, jenž si v příslušné situaci přejeme

- Nezatěžuje aplikaci
  Předání požadavku po aplikaci nevyžaduje žádné další prostředky, které by zatěžovaly její paměť nebo výpočetní výkon.

Zápory

- Nutnost specifického dosazení do kódu
  V každé části aplikace, kde si vývojář přeje reakci avatara je nutná implementace výběru druhu animace na základě vstupních dat a její následné zavolání.

- Složité řetězení animací

- Změna způsobu reakce vyžaduje zásah do kódu

3.2.2 Použití fronty požadavků

Na rozdíl od případu minulého, kde veškerou práci zastával zdrojový kód, tato metoda pro svůj chod předpokládá aktivní spolupráci aplikace. Princip spočívá ve vytvoření a udržení fronty požadavků, do které aplikace dle potřeby zasílá vstupní data využitelná avatarem. Tato fronta je modulem aktivně monitorována a příchozí informace jsou v daném pořadí zpracovány. Na základě

výsledků je nakonec zvolena a provedena vhodná reakce postavičky.
V této podobě se aplikace nestará o volbu animací, ty jsou vybrány uvnitř modulu na základě zpracování vstupních dat. Frontu je však možné použít ještě druhým způsobem, v němž se do ní zařazují již konkrétní požadavky na reakce a ty jsou postupně prováděny.
Hlavní rozdíl mezi těmito způsoby spočívá v tom, jaká část aplikace provádí nejvíce práce. Pokud se výběr animace provádí uvnitř modulu, práce a implementace jsou provedena tam. V opačném případě se vše odehrává v hlavní části aplikace. Tuto metodu lze v případě potřeby ještě nadále rozšířit na použití prioritní fronty, díky které lze ustanovit reakce postavičky, jejichž provedení má přednost před všemi ostatními.

Klady

- Možnost řetězení požadavků
  Do fronty může být zařazeno libovolné množství animací, které se postupně vykonají, jakmile na ně přijde řada

- Jednodušší implementace volby reakcí
  Analýzu vstupních dat a následnou volbu vhodné animace lze provést na jednom místě znovupoužitelného kódu. Metoda synchronního vstupu v porovnání vyžaduje ruční specifikaci reakce pro každý možný případ.

Zápory

- Složitější implementace

- Využívá prostředky aplikace
  Pro uchování požadavků a alokaci fronty je třeba využívat určité množství paměti zařízení, stejně tak musí aplikace použít část dostupné výpočetní síly k monitorování a správu fronty

- Změna způsobu reakce vyžaduje zásah do kódu

3.2.3 Konfigurační soubor avatara

Dalším způsobem ovládání, přinejmenším z jisté části, je využití externího souboru obsahujícího definice událostí a jejich reakcí. Tento soubor je při startu aplikace načtený do paměti a modul avatara začne aplikaci monitorovat v pokusu detekovat takovou událost. Pokud se tak stane, je ze seznamu vybrána příslušná reakce a ta je ihned provedena.
Jelikož však rozpoznání některých událostí bez další pomoci může být obtížným nebo dokonce nemožným úkolem, tato metoda není příliš vhodná pro samostatné použití. Pokud tomu ale situace vyhovuje, může se jednat o velice užitečný dodatečný způsob ovládání postavičky.

Klady

- Jednoduchost změny reakcí
  V případě, že si uživatel přeje změnit nějakou z používaných animací, stačí změnit příslušný konfigurační soubor bez nutnosti rekompilovat aplikaci.
- Celkový přehled chování
  Konfigurační soubor nabízí jednoduchý souhrn všech možných událostí a jejich reakčních animací

Zápory

- Samostatně nemusí být plně použitelná

3.3 Jiné druhy výstupů

Základním způsobem reakce na zachycení podmětu je v případě postavičky jednoduchá animace. Při návrhu však není nutné se na toto omezovat. Celkový zážitek lze nadále obohatit využitím dalších možností interakce s uživatelem, jejichž použití může mít pozitivní vliv i na celkovou přívětivost aplikace.

- Částicové efekty
  Jednoduché animace lze v určitých případech obohatit o líbivé částicové efekty s cílem vylepšení atmosféry aplikace. Možnosti použití sahají od jiskřiček u úst dráčka až po komplexní kouř zahalující okraje obrazovky. Po technické stránce lze volit ze dvou možných způsobů implementace, prvním z nichž je počítání a renderování aplikací v reálném čase. Toto řešení se vyznačuje vysokou kvalitou a různorodostí efektů, jeho nároky na zařízení však mohou být značné.
  Druhou možností je využití již předrenderovaných animací poskytnutých v souborech aplikace. Nároky na prostředky se v takovém případě drasticky sníží, stejně tak však klesá kvalita. Výsledky jsou navíc pevně dané a neměnné.
- Dynamické informace
  Existenci avatara lze využít i jako určitý druh prostředku pro předávání informací uživateli. Tyto informace nejsou pevně dané a jejich obsah závisí na volbě vývojáře. Například si lze představit, že se postavička začne tvářit zamyšleně a nad její hlavou se objeví obláček uvnitř nějž lze zobrazit libovolný obsah, např. ve formě textu nebo obrázku.
- Audio výstup
  Každá vizuální prezentace může pomocí správného použití zvukových

efektů nabrat zcela nových rozměrů. V tomto případě lze vybírat ze širokého spektra možností jako jsou zvuky postavičky, namluvené scény či doprovodné efekty aplikace (např. pípnutí při špatné volbě).

- Vibrace
  Většina chytrých telefonů a některé tablety poskytují možnost feedbacku pomocí vibrací. Ty mohou nastat například při zmáčknutí tlačítka nebo jiných interaktivních událostech.

- Barevné záblesky
  Indikace určitých stavů lze provést pomocí chvilkového barevného záblesku okrajů obrazovky pro zdůraznění reakce (např. červená barva pro nesprávnou volbu).

- Zvýraznění částí interface
  Postavička může také plnit roli pomocníka, k čemuž jí může napomáhat možnost zvýraznění jednotlivých částí obrazovky, případně vkládání vlastních obrázků a elementů pro vizuální doprovod událostí.

3.4 Rozbor možných reakcí avatara

Množství možných reakcí a animací je závislé pouze na představivosti a prostředcích vývojáře, existuje jich tedy nepřeberné množství. Tyto varianty tedy nelze všechny vyjmenovat, tato kapitola slouží k nastínění možností a určení druhů reakcí, které lze v daných případech aplikovat.

- Spuštění či ukončení aplikace
  Uvítací/loučící se animace. Mezi možné realizace patří mávání, euforické nadšení při příchodu, smutné shrbení při odchodu. Dráček také může při startu aplikace přiletět a přistát na obrazovce, vyklubat se z vajíčka nebo magicky zjevit z oblaku kouře. Stejné v opačném provedení platí pro chvíli, kdy uživatel aplikaci ukončí.

- Zvolení profilu hráče
  V takovémto okamžiku může dráček animací změnit vzhled z defaultní postavičky na tu, kterou si hráč uzpůsobil. Za jiné použitelné možnosti lze považovat potlesk, jásavé poskočení, odlet z obrazovky (s příletem do následného menu), reakce individuální k hráčovu profilu - pokud má například hráčův profil ikonku čaroděje, může se na postavičce objevit kouzelná hůlka a klobouk doprovázené příslušným stylem animace, obdobně pro princeznu atd.
  Klíčovým prvkem může být také doba uplynulá od posledního přihlášení, ovlivňující intenzitu reakcí.

- Přechod do hlavního menu
  Animace následující po výběru profilu. Záleží převážně na reakci z předchozí obrazovky, tedy přílet do menu, teleportace, vyskočení z vajíčka aj. Po zaujmutí pozice se postavička může rozhlížet kolem sebe po jednotlivých položkách, případně na nějakou specifickou zaostřit pohled v případě výběru.

- Prohlížení informačních sekcí
  Původní aplikace obsahovala sekce s historií a zajímavostmi o městě. Pokud se uživatel v této části nachází, dráček může vytáhnout knihu a číst si také. Jiná z možností je otočení postavičky směrem k textu a čtení z displeje samotného. Možné jsou také individuální animace v závislosti na obsahu textu.

- Stav výsledků uživatele
  Obrazovka s výsledky jednotlivých mini her a celkovým skóre. Je možné uplatnit všechny klasické animace nálad (šťastný, smutný) stejně jako reakce specifické ke hrám, ve kterých má uživatel nejlepší výsledky (pokud se dítěti dobře vede ve hře přesmyčky, postavička si může vytáhnout kamínky s písmenky a skládat si slova).
  Dále lze uvažovat výsledky a správnosti rozhodnutí hráče v reálném čase v průběhu hry samotné. Pokud zvolí správně, avatar s úsměvem souhlasí, pokud špatně zatváří se varovně.

- Prohlížení seznamu mini her
  Postavička zde může sledovat jednotlivé výběry či podávat stručné informace o hře (myšlenková bublina s obrázky hry). Dalším faktorem mohou být dlouhodobé výsledky v jednotlivých modulech a s nimi spojená změna výrazů či vzhledu (dítě zvládlo všechny součásti hry na 100%, postavička dráčka je při výběru této hry oděná v rouchu krále).

- Přechod aplikace z pozadí do popředí
  Dráček může v případě minimalizované aplikace spát nebo být úplně mimo obrazovku. Pokud spí, tak se při návratu probudí, v případě že je pryč tak se vhodným způsobem vrátí (přílet, vyskočení ze spodní části obrazovky, zjevení v plamenech...).

- Použití akcelerometru a jiného hardware zařízení
  Dítě může zařízením zatřepat a dráčkovi se z toho zamotá hlava nebo při prudkém pohybu ztratí rovnováhu. Při významné změně GPS polohy (např. vstup do Francie) lze aplikovat kulturně sladěnou aplikaci (pojídání bagety, prohlížení Eiffelovi věže...). Stejné techniky lze provést pro počasí či teplotu okolí (postavička se bojí bouřky, klepe zimou, chytá sněhové vločky).

- Doprovodné animace
  Tyto animace se vztahují přímo k jednotlivým mini hrám. Pro příklad, když dítě hraje hru s vybarvováním obrázku, tak postavička na počátku vytáhne štětec a při zvolení barvy si ho namočí do barvy, poté při vybarvení části obrázku štětcem krátce maluje po malé části obrazovky (barva za okamžik mizí).

- Přímá interakce s avatarem
  V případě, že se se hráč postavičky dotkne, ta může zavrávorat nebo uskočit a poté se vrátit. Také lze sledovat pohyb prstu po obrazovce zařízení.

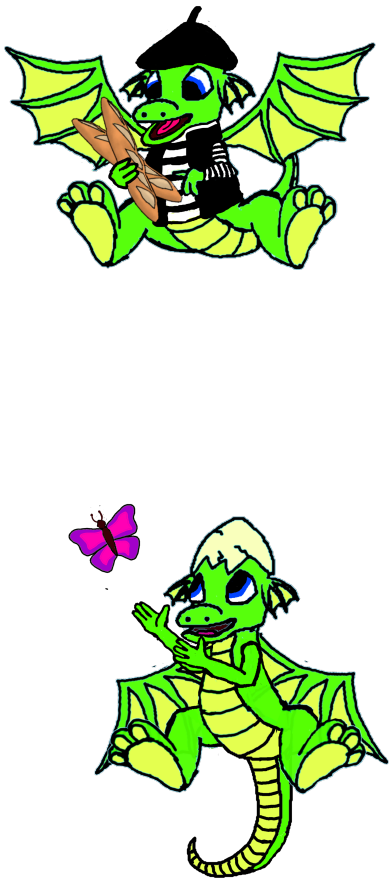
- Idle animace
  Ve chvíli kdy uživatel neprovedl delší dobu nějakou akci, dráček může usnout a probudit se až poté, co je zadán nový vstup. Toto lze prokládat výplňovou zábavní animací jako je nahánění motýla po obrazovce či tancování.

- Datum a události
  Speciální animace ke svátkům a narozeninám. V den vánoc si postavička může rozbalovat dárky, pokud má dítě narozeniny tak může mít narozeninový čepec a rozhazovat barevné papírky.

- Načítací obrazovky
  Animace podobné idle akcím, prováděné pro zabavení uživatele. Mohou i nemusí mít něco společného s tím, jaká část aplikace se načítá. Pro příklad lze uvést žonglování dráčka s míčky.

- Gesta a rychlost reakcí
  Opět závislé na situaci. Obecně však pokud je rychlost vstupů velmi vysoká, může se postavička na okamžik tvářit zmateně a zamotaně. Reakce na gesta je možná například na gesto otočení, kdy spolu s ním může postavička otáčet hlavou (do jistých povolených hranic).

- Rozšíření herní databáze
  Postavička může na nově dostupné prvky a moduly upozorňovat animací toho se týkající, například ukazovat na seznam mini her nebo tuto hru rovnou zjednodušeně hrát.



Obrázek 3.2: Koncepty možných animací, autor: Karel Kovařovic

Kapitola 4

Návrh prototypu avatara Dráčka

Dosavadní části práce se zabývaly rozborem velkého spektra možností a doporučení použitelných pro implementaci takového postavičky. Reálná implementace veškerých těchto vlastností je však daleko nad rámec této práce. Tato kapitola se proto zabývá zvolením omezeného množství prvků a vhodným způsobem jejich realizace.
Na implementační stránce práce se také z velké části podílí další student, Karel Kovařovic, v roli grafika. Veškeré nákresy a animace postavičky v této práci byly realizovány právě jím.

4.1 Koncept vzhledu postavičky

Grafikem vytvořený prvotní návrh postavičky Dráčka. Koncept byl vytvořený s ohledem na výsledky testů provedených v kapitole analýzy avatarů a s důrazem na splnění Lorenzových kritérií roztomilosti (kapitola 1).

Obrázek 4.1: Návrh vzhledu Dráčka

4.2 Funkční a nefunkční požadavky prototypu

Funkční požadavky

- **Schopnost práce s konfiguračním souborem**
  - Načítání dat
    Aplikace musí být schopna načíst a zpracovat data konfiguračního souboru a upravit podle nich svůj stav
  - Ukládání dat
    V předešlé části načtená data musí být možné znovu uložit v podobě v jaké se v aplikaci zrovna nachází (soubor se přepisuje)
- **Simulátor**
  - Rozšiřitelnost vstupních typů
    Aplikaci musí jít rozšířit o další druhy vstupů dle přání vývojáře (tzn. přidání simulací novách událostí).
  - Nastavení parametrů simulace
    Každá z událostí bude obsahovat libovolné typy parametrů, které u ní lze nastavovat (pro účely simulace, tyto údaje se nebudou ukládat)
  - Volba příslušné animace
    Aplikace na základě současného stavu rozhraní (parametrů) rozhodne o zvolení vhodné animace k přehrání
  - Přehrání zvolené animace
    Zvolená animace, vyhodnocena v předchozím bodě, bude přehrána na obrazovce zařízení
- **Konfigurátor**
  - Úprava a nastavení parametrů vstupů
    Parametry potřebné k vhodné volbě animace (např. časová prodleva pro reakci) jsou ukládány v konfiguračním souboru a tato část aplikace bude umožňovat jejich manipulaci
  - Přehled nastavených animací
    Aplikace bude zobrazovat jaké animace jsou pro daná nastavení zvoleny
  - Změna animace vstupu
    Zobrazené údaje o animaci z předchozího bodu bude možné na přání změnit

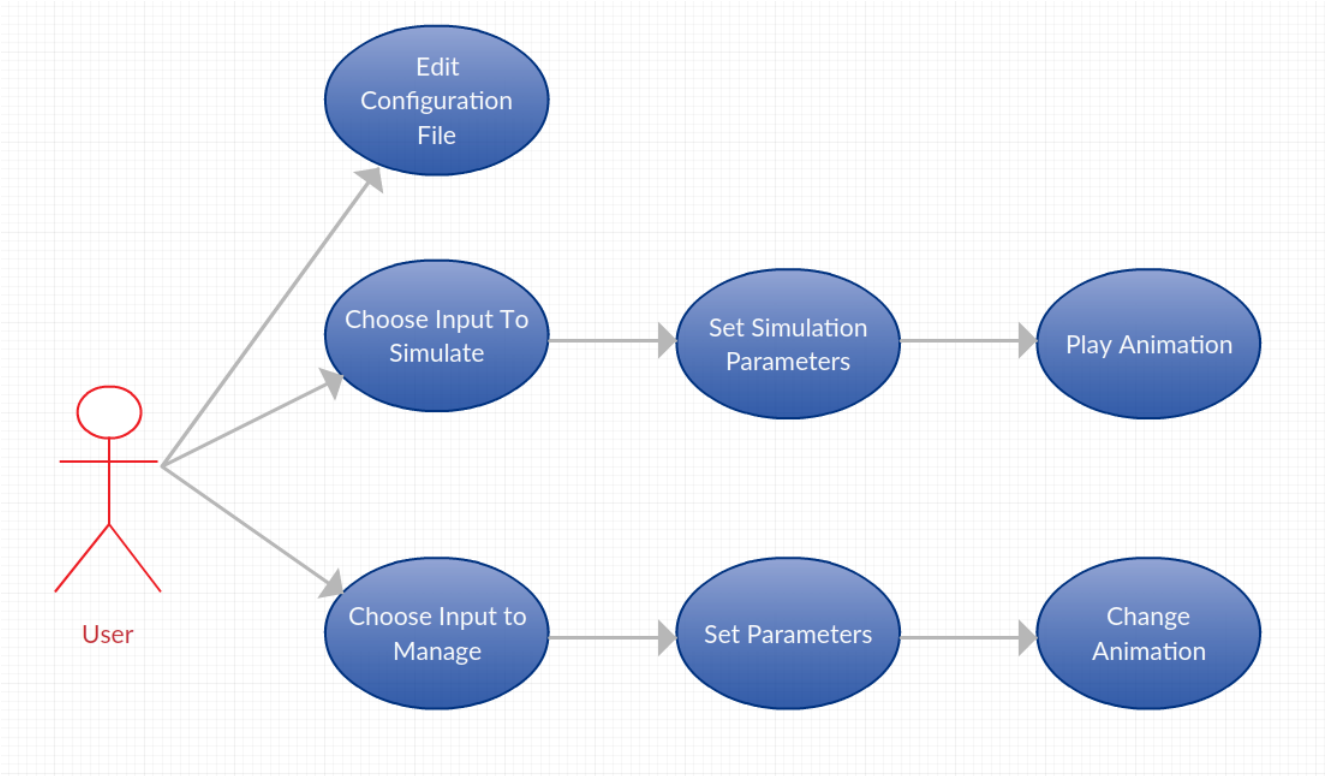
- **Vstupní typy pro implementaci v prototypu**
  - Ano/Ne odpověď
    Část simulující odpověď na úkol či otázku kde je správnost odpovědi jednoznačně určitelná
  - Částečná odpověď
    V tomto případě lze správnost odpovědi rozdělit na procentuální úspěch a reakce pro tyto intervaly mohou být jiné
  - Neaktivní uživatel
    Vstup simulující uživatele, který nebyl aktivní po určitý (nastavitelný) čas
  - Změna profilu
    Tento vstup bude jednat jako údaj ze chvíle, kdy uživatel zvolil nějaký profil

Nefunkční požadavky

- Grafické uživatelské rozhraní
- Systém Android
- Jazyk Java
- Konfigurační soubor formátu JSON
- Použitelné na tabletu

4.3 Případy užití

Uživatel může aplikaci použít třemi hlavními způsoby, které se od sebe odlišují. Tyto způsoby popisuje následující Use Case Diagram (graf případů užití, obrázek 4.2).



Obrázek 4.2: Use Case Diagram

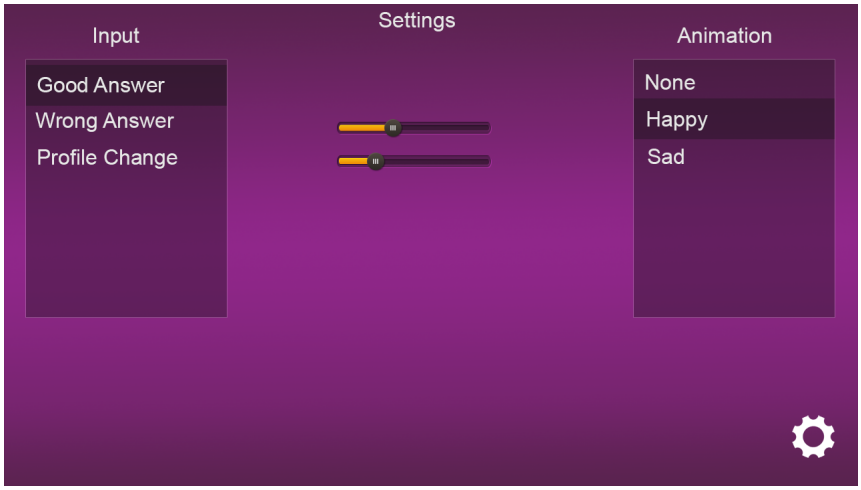
4.4 Simulátor chování avatara v závislosti na vstupu

Vzhledem k tomu, že aplikace, pro kterou je postavička navrhována je v době psaní této práce stále ještě v raných fázích vývoje, zabývá se implementační stránka práce návrhem a vývojem simulátoru volby animací.
Tato aplikace bude pracovat s konfiguračním souborem obsahujícím informace o nastavení vstupních událostí a o tom, jaké animace mají tyto události přiřazené. Výsledná aplikace tedy bude sloužit nejen jako simulátor, ale také jako grafické rozhraní pro úpravu konfiguračního souboru avatara.
Aplikace se bude skládat ze dvou hlavních částí

- **Jádro aplikace**
  Hlavní část aplikace, implementovaná autorem práce, umožňující výběr z dostupných vstupních signálů a přehrání k němu přiřazené animace postavičky. Tyto animace bude možno libovolně měnit ze seznamu dostupných možností a dále upravovat vlastnosti událostí (např. velikost časové prodlevy před reakcí na neaktivitu uživatele).
- **Knihovna animací**
  Druhá součást aplikace, implementována grafikem, Karlem Kovařovic, bude obsahovat implementaci požadovaných animací postavičky. Tyto animace budou volány a přehrávány z jádra aplikace v případě jejich zavolání.

4.5 Wireframy prototypu

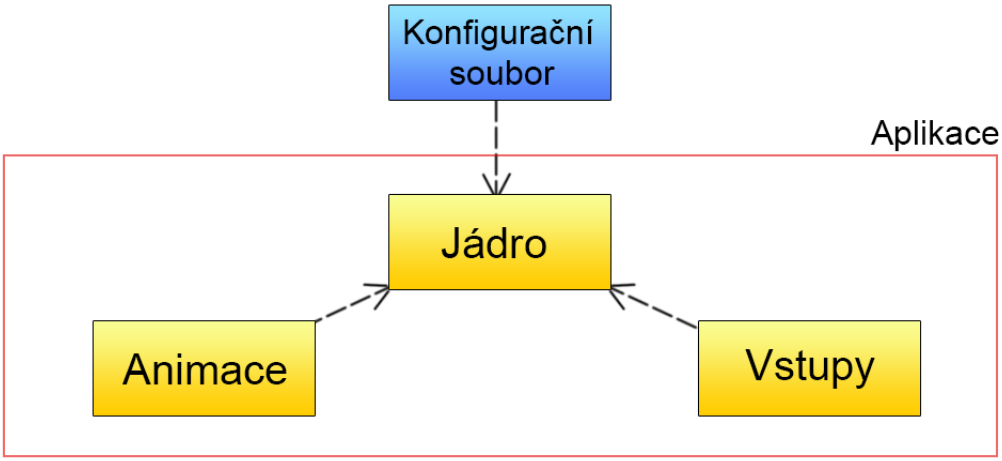
Obrázek 4.3: Prvotní koncept vzhledu aplikace, hlavní obrazovka



Obrázek 4.4: Prvotní koncept vzhledu aplikace, obrazovka nastavení

4.6 Architektura aplikace

Grafické znázornění složení součástí aplikace (obrázek 4.5). Jádro využívá jednotlivé třídy vstupů, jejichž data načítá z konfiguračního souboru. Volané animace jsou nakonec dodány samostatným modulem.



Obrázek 4.5: Architektura aplikace

4.7 Zvolené vstupy a výstupy prototypu

4.7.1 Kategorie vstupů

Základní myšlenkou této aplikace je nastavitelnost libovolné animace k dané události. Vstupy jsou tedy rozděleny do kategorií, které sdílejí stejné parametry a uživatelské rozhraní.

Typ vstupu	Parametry
Ano/Ne odpověď	Čas od poslední odpovědi Počet odpovědí v řadě (0 - 5) Úroveň dokončení cvičení (0 - 100%)
Částečná odpověď	Čas od poslední odpovědi Správnost odpovědi (0 - 100%)
Neaktivní uživatel	Stav aplikace Uplynulý čas (1 - 200 sekund) Časovač pro reakci (1 - 200 sekund)
Změna profilu	Významný den Poslední přihlášení před (0 - 365 dny)

4.7.2 Parametry vstupů

Priorita parametrů je u jednotlivých vstupů v tabulce seřazena sestupně. Vyhodnocení je provedeno tak, že se nejprve zkontroluje parametr s nejvyšší prioritou a pokud ten vyhověl a zároveň má nastavenou animaci, pak se jádru odešle id této aplikace a vyhodnocování se ukončí. V případě, že taková situace nenastane, vyhodnotí se stejným způsobem parametr s prioritou nižší.

4.7.3 Navrhované animace prototypu

Seznam animací pro nákres a implementaci, realizovaný grafikem. Tyto animace budou v aplikaci vypsány pod svými jmény a volány jednoznačným identifikátorem, pro který bylo zvoleno číslování od 1 nahoru.

Název	ID	Popis
Happy	1	Dráček je šťastný
Jumps happily	2	Dráček skáče nadšením
Cries	3	Dráček je smutný a pláče
Cries desperately	4	Dráček je velice smutný
Butterfly chase	5	Dráček nahání motýla
It snows!	6	Dráček chytá vločky sněhu
Waves	7	Dráček zamává
Aye	8	Dráček s úsměvem přikývne
Nay	9	Dráček odmítavě zakroutí hlavou

4.8 Návrh formátu konfiguračního souboru

Aplikace bude pro uchovávání dat o nastavení používat soubor ve formátu JSON. Frameworkem používaný serializér používá tento formát bez použití uvozovek pro jména a hodnoty a bez čárek na konci řádků.
Soubor obsahuje pole tříd, které dědí informace od nadtřídy AppInput. Každý ze vstupů tedy musí mít minimálně tyto proměnné (hodnoty jsou ukázkou vstupu Good Answer)

```
class: com.preview.app.AnswerInput      // Trida
type: answerInput                       // Typ
inputId: 1                              // ID vstupu
btnText: Good Answer                    // Text pro zobrazeni
```

Forma dalších položek je závislá na implementaci daného typu vstupu. Pro třídy implementované v této práci se jedná o následující proměnné (výsledná forma se může mírně lišit). Hodnoty jsou náhodně zvolené.

Ano/Ne odpověď

```
fastAnswerTime: 33            // Rychla odpoved do
fastAnswerAnimationId: 1      // ID animace rychle odpovedi
completenessInterval1Max: 70 // Vrchni hodnota 1. intervalu
completenessInterval2Max: 90 // Vrchni hodnota 2. intervalu
answerStreak0: 1              // ID animaci
answerStreak1: 2
answerStreak2: 1
answerStreak3: 2
answerStreak4: 1
answerStreak5: 2
interval1AnimationId: 1
interval2AnimationId: 1
interval3AnimationId: 3
```

Neaktivní uživatel

```
timer1: 85              // Hodnoty casovacu
timer2: 60
timer3: 50
animationId1: 1         // ID animaci
animationId2: 2
animationId3: 3
```

Změna profilu

```
event1AnimationId: 2          // ID animaci udalosti
event2AnimationId: 1
event3AnimationId: 4
interval1AnimationId: 4       // ID animaci uplynule doby
interval2AnimationId: 3
interval3AnimationId: 2
interval1Max: 100             // Vrchni hodnota 1. intervalu
interval2Max: 300             // Vrchni hodnota 2. intervalu
```

Částečná odpověď
Viz. příloha Ukázková implementace třídy vstupu (B), část B.4.

Kapitola 5

Implementace simulátoru

Simulátor chování avatara je samostatně spustitelná nezávislá aplikace zaměřená na tablety se systémem Android. Tato aplikace čerpá data z externího konfiguračního souboru, který určuje stav aplikace při zapnutí a umožňuje jeho změnu za pomoci jednoduchého grafického rozhraní.

5.1 Použité prostředky a technologie

Pro vývoj aplikace byl použit framework LibGDX [19], jehož cílem je snadný a efektivní vývoj 2D her. Framework pracuje v jazyce Java, tedy v jazyce nativním pro vývoj Android aplikací, a je multiplatformní, umožňující překlad aplikace jak pro systémy stolních počítačů, tak pro mobilní zařízení (Android, iOS, BlackBerry). V případě potřeby nabízí také podporu HTML5 a webGL. Tento framework obsahuje také vlastní graf scény, Scene2d [13], jehož použití je volitelné. Tato součást byla v aplikaci použita pro vytvoření a správu ovládacích prvků grafického rozhraní (tlačítka, posuvníky atd.).

Aplikace byla napsána za pomocí vývojového prostředí Android Studio 1.3 [20] využívajícím Java SE Development Kit 8 [21]. Pro ukládání dat do souboru je použit formát JSON.

Sticky Butterscotch UI Kit [22] pro vzhled posuvníků rozhraní.

5.1.1 Pomocné nástroje

GDX Texture Packer [23]
Samostatně stažitelný nástroj pro LibGDX, umožňuje zformování velkého množství malých obrázků do jednoho velkého spolu s vytvořením souboru obsahujícím souřadnice těchto obrázků.

Hiero [24]
Nástroj pro tvorbu bitmapové verze fontu ve zvolené velikosti a barvě. Takovýto font lze poté použít pro vykreslování textu.

5.2 Vysvětlivky k frameworku

Mimo vlastní metody, vysvětlené v dalších částech používá LibGDX také své vlastní proměnné, které jsou v programu často používané. Z důvodu lepšího objasnění parametrů volaných metod jsou zde proto tyto a některé další třídy vysvětleny.

- Batch
  Třída starající se o vykreslování objektů na obrazovku. Vše co se v LibGDX vykresluje se kreslí do batche. Jelikož se jedná o masivnější strukturu, je doporučeno používání pouze jedné instance této třídy.

- Sprite
  Datová struktura obsahující texturu spolu s dalšími informacemi o pozici, velikosti atd.

- Stage
  Struktura používaná grafem scény Scene2d. Shromažďuje všechny použitelné objekty (Actors) a dovoluje složitější operace nad nimi. V této aplikaci je stage použitý pouze pro ovládací prvky rozhraní.

- Atlas
  Grafický objekt obsahující data více textur v jednom nebo více velkých obrázcích. Pomocí .atlas souboru dokáže určit názvy a souřadnice obsažených textur a na požádání je dodat.

- Skin
  Struktura obsahující textury a skiny pro tlačítka a další ovládací prvky. V této aplikaci se jedná v podstatě o stejná data jako v atlasu.

- BitMapFont
  Obsahuje obrázek fontu a souřadnice jeho znaků. Používá se pro tisk textu na obrazovku.

5.3 Způsob implementace

5.3.1 Jádro aplikace

Hlavní část aplikace, umožňující načtení dat ze souboru a jejich ukládání, stejně jako manipulaci s jednotlivými třídami vstupních typů a zobrazování získaných animací.

Všechny pracovní soubory (textury, fonty, data) jsou uloženy ve složce aplikace v podadresáři *android/assets*. Zde se i nachází soubory *textures.png* a *textures.atlas*, které zajišťují načtení většiny potřebných grafických elementů v jednom obrázkovém souboru, což zvyšuje výkon aplikace a přehlednost kódu.
Aplikace samotná se skládá ze dvou obrazovek. V první, simulační, si uživatel nejprve vybere požadovaný vstup pomocí tlačítek šipek. Tyto vstupy jsou naimplementovány v podobě tříd, uvnitř kterých je zajištěna veškerá pro funkci potřebná logika. Aplikaci je tedy možné volně rozšiřovat o další vstupní typy s minimálním zásahem do kódu jádra. V tom je pouze třeba zajistit rozšíření načítací funkce loadInputsFromJSON() o řádek schopný na tuto třídu načtený vstup přetypovat (viz. příklad implementace).
Na obrazovce druhé, nastavení, lze nalézt nástroje pro přiřazení animací k událostem a jejich konfiguraci. V této části se nachází dva seznamy, jeden obsahující všechny načtené vstupy, druhý obsahující všechny načtené animace. Seznamy jsou vytvořeny a spravovány jádrem aplikace, třídám vstupů je poté předána reference na seznam animací pro možnost další práce.

5.3.2 Třída vstupu

Jednotlivé typy vstupů jsou řešené jako samostatné třídy, používané jádrem aplikace. Pro tento účel bylo použito polymorfismu s děděním od hlavní třídy AppInput, která uchovává údaje o typu, id a textu vstupu.
Každá z následných podtříd může uchovávat libovolné údaje a implementovat libovolné metody, musí však vždy rozšiřovat třídu AppInput a implementovat všechny její metody.
Tyto třídy jsou zodpovědné za vykreslování své soukromé části grafického rozhraní, uchovávají si tedy uvnitř sebe další třídy implementující tyto prvky.
Metody tříd jsou volány pomocí přetížení metod mateřské třídy, čímž se zajišťuje stejný kód jádra aplikace pro libovolný počet rozšíření.

5.4 Součásti jádra

- public void create ()
  Základní funkce frameworu LibGDX, zavolána ve chvíli, kdy je aplikace spuštěna. Tato funkce je použita pro inicializaci globálních proměnných a celkového nastavení aplikace do úvodního stavu, ze kterého je možné její další používání.
- public void render ()
  Hlavní funkce použitá pro vykreslování na obrazovku. Vytvořena a spravována frameworkem, tato funkce je volána v nekonečné smyčce, v níž při každém zavolání překreslí obsah obrazovky svým současným obsahem.
- public void resize(int width, int height)
  Funkce implementována pro potřeby frameworku, zajišťující správné roz-

lišení a souřadnice obrazovky. Její implementace je kritická pro správnou funkci grafu scény. Obsah funkce upravuje nastavení kamery a viewportu aplikace na současné hodnoty rozlišení okna (width, height).

- setUpOutputTable(), setUpInputTable()
  Metody, které vytvářejí a zaplňují seznam vstupů / animací

- loadInputsFromJSON, saveInputsToJSON
  Metody pro načtení a uložení dat ve třídách vstupu. Tyto třídy krom vlastních dat obsahují také struktury s tlačítky a ovládacími prvky, které je nevhodné ukládat. O toto se stará metoda destroyUI, která je před každým uložením zavolána. Rozhraní je poté znovu vybudováno zavoláním metody initUI.

Další v jádru nalezené metody se starají o věci, jako jsou nastavení vzhledu a souřadnic tlačítek, přiřazení listenerů či animování součástí rozhraní pomocí úpravy souřadnic a alfa hodnot v závislosti na uplynulém čase od započetí akce (delta time).

5.5 Součásti třídy událostí (vstupů)

Jak již bylo v předchozí části zmíněno, tyto třídy dědí od třídy AppInput, jejíž metody je nutné pro správnou funkci implementovat.

- public void initUI(Skin skin, List animationList, float worldWidth, float worldHeight, ArrayList<animationData>animationData, BitmapFont fontRef)
  Metoda sloužící k inicializaci všech ovládacích prvků třídy (tlačítka, posuvníky, hodnoty) a dalších nastavení. Tato metoda je zavolána vždy při spuštění aplikace a poté po stisknutí uložení, při němž se veškeré rozhraní ve třídách rozebírá. Parametry jsou reference na strukturu se skinem prvků, seznam načtených animací, rozměry kamery, reference na strukturu obsahující jména a id všech dostupných animací a nakonec reference na strukturu fontu.

- public void setupMainUI(Stage stage)
  Pokud je aplikace na hlavní obrazovce simulátoru a je změněn typ vstupu (tlačítko šipky), je zavolána tato metoda. Její funkcí je přidání příslušných ovládacích elementů do Stage rozhraní a tím zahájení jejich vykreslování.

- public void setupSettingsUI(Stage stage)
  Stejné jako v přechozím případě, pouze se volá na obrazovce nastavení.

- public void clearUI(Stage stage)
  Volá se pro odstranění ovládacích prvků současné třídy z obrazovky. Volá se před změnou vstupu.

- public void destroyUI()
  Nastavení nepotřebných datových struktur na hodnotu null zabrání jejich uložení do souboru JSON. Ty je po uložení znovu třeba inicializovat metodou initUI.

- public void requestAnimation()
  Vyhodnotí současný stav ovládacích prvků aktivní třídy a vrátí příslušnou animaci. Volá se při stisknutí tlačítka pro přehrání animace na hlavní obrazovce.

- public void drawUIText(Batch batch)
  Vykreslení textu k současnému rozhraní hlavní obrazovky. Font k použití byl nastaven v metodě initUI.

- public void drawUISettingsText(Batch batch, float worldWidth, float worldHeight)
  Stejné jako předchozí, pouze pro obrazovku nastavení.

- public void selectCurrentAnimation()
  V tabulce animací v nastavení změní vybranou animaci při změně parametrů nebo vstupu. Tato animace vždy reprezentuje zvolený výstup pro daný vstup. Volá se při změně v tabulce vstupů (nastavení) nebo při změně v ovládacích prvcích třídy (nastavení).

- public void setCurrentAnimation(int id)
  V datech třídy změní nastavení id animace náležící současnému nastavení ovládacích prvků na hodnotu id. Volá se při změně v tabulce animací (nastavení).

5.6 Ukázková implementace třídy vstupu

Vzhledem k rozsáhlosti kapitoly byla tato část přidána do sekce příloh (B).

Kapitola 6

Testování

Po dokončení implementace je třeba aplikaci otestovat ve vhodném prostředí na cílové skupině uživatelů pro zjištění závad a nedostatků napravitelných v dalším vývoji.
Nejdříve ze všeho je tedy nutné ustanovit personu cílového uživatele na kterém bude testování probíhat. Na základě tohoto lze poté sestavit nezbytné dotazníky a scénáře testování.

6.1 Charakteristika cílového uživatele

Hlavní účel aplikace spočívá v úloze úprav konfiguračního souboru v jednoduchém grafickém prostředí. Této funkce tedy nejvíce využijí zejména *vyučující* dětí, které finální aplikaci Dráček používají. Ze získaných informací lze sestavit takovouto fiktivní personu.

Jan Černý, vyučující na základní škole v Praze
Jan se narodil v roce 1981 a od mala se mohl chlubit zvídavou povahou. Jediné co mu přinášelo více potěšení než získávání znalostí bylo jejich šíření mezi ostatními podobné povahy. Při výběru střední školy se tedy rozhodl následovat kariéru učitele a brzy úspěšně dokončil střední i vysoká studia se zaměřením na výuku dětí.
Své uplatnění našel na základní škole v Praze, kde již úspěšně vyučuje po dobu téměř deseti let. Ve svém povolání nevyžaduje velké technické znalosti, přesto však moderní technologie dokáže ovládat na úrovni průměrně znalého uživatele.

Za další skupinu uživatelů, kteří se s aplikací mohou setkat lze považovat *studenty* technických oborů, kteří se budou zabývat dalším vývojem aplikace Dráček nebo postavičky samotné.

6.2 Druh a průběh testování

Testy aplikace budou provedeny v prostorách fakulty uvnitř připravované laboratoře zaměřené na testování uživatelského rozhraní. Přestože tato laboratoř není v současné chvíli stále ještě plně funkční, její zařízení jsou pro účely testování dostačující.
Testování samotné bude probíhat následovně. Skupina uživatelů specificky vybraných pro testování bude v určený den pozvána do prostor laboratoře, kde jim budou předloženy úvodní dotazníky, které na místě vyplní. Poté budou uživatelé po jednom vstupovat do prostor laboratoře, kde jim bude předložen scénář akcí, které mají pomocí aplikace provést a bude sledována úroveň s jakou se jim podaří akci dokončit.
Nakonec uživatelé vyplní závěrečný dotazník týkající se jejich zážitku s aplikací a případných komentářů. Tím pro ně testování končí. Tato data budou po celkovém skončení testu vyhodnocena a navržena možná zlepšení.

6.3 Dotazníky

Tato kapitola shrnuje otázky vhodné pro použití v požadovaných testovacích fázích. Výslednou formu zhotovených dotazníků lze nalézt přílohách práce.

6.3.1 Dotazník před testem

Tento dotazník má za úkol zhodnotit technické schopnosti uživatele a jeho vhodnost pro toto testování.

- Jméno / ID
- Povolání: student / vyučující / jiné
- Vaše zkušenosti se systémem Android: 1 (velké) - 5 (žádné)
- Jak často používáte tablet: 1 (často) - 5 (nikdy)
- Jak často používáte notebook či stolní počítač: 1 (často) - 5 (nikdy)
- Znalost datového formátu JSON: 1 (velké) - 5 (žádné)

6.3.2 Dotazník po testu

V této části uživatel zodpoví dotazy týkající se postřehů a pocitů z testování samotného. Dotazy tedy musí směřovat k testovaným scénářům a aplikaci. Pro otázky s odpovědí na stupnici od 1 do 5 platí hodnocení 1 nejlepší, 5 nejhorší.

- Provedení úkolů uvnitř aplikace bylo snadné: 1 - 5
- Manipulace s konfiguračním souborem byla snadná: 1 - 5

- Grafické rozhraní aplikace je lehce pochopitelné: 1 - 5
- Význam ikon aplikace je zřejmý: 1 - 5
- Práce se simulační částí aplikace je příjemná: 1 - 5
- Práce s konfigurační částí aplikace je příjemná: 1 - 5
- Formát konfiguračního souboru považuji za vhodný: 1 - 5
- Barevné schéma aplikace mi vyhovuje: 1 - 5
- Volitelný komentář k příjemnosti užívání

6.4 Scénáře

Tato část navrhuje scénáře průběhu testování, tedy jaké akce se po uživateli budou v aplikaci požadovat.

6.4.1 A

V simulační části aplikace by jste si rádi přehráli animaci přiřazenou ke změně profilu hráče za předpokladu, že jsou zrovna vánoce. Využijte připraveného tabletu a pokuste se takto učinit.

6.4.2 B

Rádi by jste pomocí aplikace změnili nastavení animace pro vstup špatné odpovědi (Wrong Answer) v případě, že tento vstup nastane ve chvíli, kdy uživatel potřetí špatně odpověděl (Answer Streak). Nastavte pro takovouto situaci libovolnou z dostupných animací.

6.4.3 C

Rozhodli jste se, že Vám současné nastavení intervalů splnění úlohy pro vstup částečné odpovědi (Partial Answer) nevyhovuje a rádi by jste nastavení změnili na podporu dvou intervalů, 0 - 50% a 51 - 100% s výběrem libovolné z dostupných animací pro každý z nich.
Použijte připravený notebook s otevřeným konfiguračním souborem aplikace k tomu, abyste tyto změny provedli. Předpokládejte, že víte, že typ vstupu se nazývá *partialAnswerInput* a intervaly se definují nastavením hodnoty jejich vrchní hranice (spodní se nastavuje automaticky).

6.5 Výsledky testů

Připravené úkoly byly předloženy skupině uživatelů a testy proběhly dle stanoveného postupu. Přestože ne všichni z testovaných oplývali zkušenostmi s používám tabletu, úvodní dotazník odhalil, že každý z nich disponoval nejméně průměrnou zkušeností se systémem Android.

Nejběžnějším z pozorování během průběhu testování byla počáteční nutnost orientace uživatele v prostředí aplikace, jehož součásti nemusí být na první pohled zřejmé. Přesto však tito uživatelé po získání přehledu o významu aplikace neměli větší problémy s vykonáním zadaných úkonů.
Jedním z hlavních nedostatků, které byly během testování odhaleny byla složitost rozlišení rozevíracího menu od textového pole.
Čas od času bylo také pozorováno zmatení při identifikaci simulační a konfigurační části aplikace, které postrádají jasné označení.
Byl také zjištěn problém s předpoklady uživatele o chování tlačítka „Zpět“, které namísto návratu do simulátoru (z nastavení) ukončí aplikaci.

Celková spokojenost uživatelů s prostředím a přívětivostí aplikace byla kladná, stejně tak nebyl problém ani s přímou editací konfiguračního souboru (včetně uživatelů neznalých použitého formátu).

6.6 Reakce na výsledky testování

Současnou verzi aplikace je zajisté možno vylepšit zjednodušením orientace uživatele při prvním spuštění. Toho je možno dosáhnout přidáním dodatečných popisků, nadpisů a případným tlačítkem nápovědy obsahující stručný návod k aplikaci.

Vylepšení je dále zapotřebí s označením vysunovacích nabídek tak, aby jejich účel byl na první pohled zřejmý. Přidání znaku šipky, nebo jiné pro tyto účely běžně používané ikony, vedle takového položky by mělo být dostatečným řešením.

Na tlačítko „Zpět“ systému Android (ve frameworku libGDX odpovídajícímu hodnotě„Keys.BACK“) je možno přiřadit listener, který, pokud je aplikace v sekci nastavení, namísto jejího vypnutí přejde do minulé části.

Závěr

O práci

Tato práce se ve svém obsahu věnovala detailnímu rozboru možností animovaných postaviček, které nějakým způsobem jednají jako průvodci či pomocníci ve svých domovských aplikacích.
V práci provedený rozbor se dá rozdělit zejména na dvě hlavní kategorie. Těmi jsou

- Události, které nějakým způsobem zajišťují reakci postavičky (vstupní data)
- Reakce avatara (animace) zvolené na základě vstupních či jiných dat

Tyto kategorie jsou v práci do detailu rozebrány a na základě získaných informací jsou ustanoveny avatarem použitelné možnosti.

Některé z těchto možností jsou následně naimplementovány v prototypu simulační aplikace, která dovoluje úpravu konfiguračního souboru avatara. Tento soubor je zodpovědný za způsoby chování této postavičky na předem definované reakce.

Pro aplikaci byly nakonec vytvořeny testy uživatelského rozhraní, které byly také vykonány. Údaje tímto způsobem nabyté lze následně použít jak pro další vývoj a ladění aplikace, tak pro zjištění vhodnosti formátu konfiguračního souboru.

Úroveň splnění zadání práce

Finální ohlédnutí na zadané úkoly a diskuze nad způsobem jejich splnění.

- **Analyzujte aplikaci Dráček a vyberte vstupní informační kanály určené pro ovládání Avatara**
  Této části se věnuje kapitola 2, jež rozebírá možnosti staré verze aplikace Dráček, shrnuje dostupné záměry její nové verze a na základě rozboru uživatelských rozhraní, které je určeno jako jeden z hlavních vstupních prvků, se pokouší předpokládat (a doporučit) vhodný směr vývoje této aplikace. Ze získaných informací je poté vytvořen seznam ideálních ovládacích prvků.

- **Analyzujte výrazové prostředky avatara (gesta, činnosti, atd.) vhodné pro výukovou aplikaci pro děti předškolního a školního věku**
  Kapitola 1 volí a zkoumá zástupce již existujících avatarů (více i méně populárních) a pokouší se vytvořit obraz toho, jak by taková postavička, implementována v aplikaci Dráček, měla vypadat (grafikou i chováním).

- **Navrhněte modul Avatar použitelný aplikací Dráček pro vizuální interakci s uživatelem (prostřednictvím avatara)**
  Modul byl navržen s ohledem na práci s konfiguračním souborem chování avatara, který je v budoucnosti použitelný pro potřeby aplikace Dráček v případě implementace takového postavičky do hotové aplikace.

- **Implementujte tento modul pod OS Android**
  Implementace byla provedena za pomoci frameworku libGDX a vytvořena s možností překladu pro desktopové systémy pro možnost rychlého ladění.

- **Otestujte uživatelskou přívětivost a srozumitelnost avatara v UX laboratoři na cílové skupině uživatelů**
  Sekce Testování (6) charakterizuje personu cílového uživatele pro v této práci vyvinutou aplikaci a na ní staví příslušné testovací dotazníky a scénáře. Tyto testy byly vzhledem k nedokončenému stavu laboratoře provedeny v improvizovaných podmínkách a v případě dalšího rozvoje aplikace by bylo vhodné je zopakovat v již dokončené laboratoři.

Možná vylepšení simulátoru

Aplikace, která byla v této práci naimplementována je v závěru práce stále ještě ve stádiu prototypu. To znamená, že disponuje mnoha nedokonalostmi a také nabízí množství možných rozšíření.
Za jedno z takovýchto rozšíření se dá považovat možnost přímého přístupu

do textové verze konfiguračního souboru z aplikace samotné a jeho následná editace. To je v aplikaci v této chvíli nabízeno jen formou grafického rozhraní, které nemusí podporovat úpravy pokročilejších typů parametrů.

Lákavým prvkem také je možnost rychlého náhledu animace při jejím nastavení v konfigurační části aplikace, případně možnost volby různých parametrů stejných animací (barvy částí postavičky, rychlost pohybu a gest atd.).

Aplikace jako taková by nakonec, ve finální podobě, mohla být použitelná jako vnitřní součást aplikace Dráček, dostupná vyučujícím pro formu detailní konfigurace chování avatara.

Budoucnost projektu

Budoucnost projektu Dráček je široká a nabízí se nespočet možných rozšířeních a adaptací. Jednou z těchto adaptací je plánovaná webová verze aplikace, sloužící pro snadný přístup k aplikaci odkudkoli.

Verze software pro systémy Android je stále ve svých zárodcích a již nyní se dočkává rozšíření, která byla pro původní aplikaci nemyslitelná, jedním z nichž lze považovat právě postavičku zde navrhovanou.

Co se týče budoucnosti avatara samotného, vždy se nabízí možnost rozšíření o nové animace, návrh a implementace nových modifikací postavy či kupříkladu obohacení avatara o hlas s využitím namluvených scénářů.

Ideální podoba avatara

Verze dráčka implementující všechny v této práci zmiňované (a další) vlastnosti je dalece nad možnosti i mateřské aplikace samotné, nedá se tedy předpokládat, že by se někdy stala realitou. Přesto se však lze zamyslet nad tím, jak by mohla postavička vypadat, kdyby tomu tak bylo.
Autor práce si představuje bohatě kreslenou postavičku dráčka, s níž se děti aplikaci užívající dokáží sblížit a která je svou přítomností jak provede skrze méně zábavné úkoly, tak obohatí celkové zážitky z částí příjemných.
Postavička podporuje rozmanité způsoby přizpůsobení jejího vzhledu, ať již přes oblečky a barevná vylepšení, tak i možností použití zcela jiného modelu pro pokrytí větší škály chutí uživatelů.
Tyto možnosti jsou hráči povoleny při splnění jistých podmínek, jako je například pokoření těžkých úkolů nebo překonání bodové hranice cvičení a chovají se svou přítomností jako odměny pro ty nejlepší a nejpilnější.
Dráček má v této podobě již také plně funkční hlas, jehož pomocí je dítěti schopen poskytnout ještě lepší možnosti podpory a nápovědy a který mu přidává

na charakteru. Tímto způsobem je schopen přilákat široký okruh uživatelů, kteří se díky jeho pomoci nejen něco nového naučí, ale ze všeho nejpodstatněji, se budou skvěle bavit.

Literatura

[1] CUPCHIK, Gerald C., Janos László.*Emerging Visions of the Aesthetic Process: In Psychology, Semiology, and Philosophy.*Cambridge University Press.1992

[2] SWARTZ, Luke.*Why people hate the paperclip: labels, appearance, behavior and social responses to user interface agents* [online].[cit. 2015-10-03]. Dostupné z:
http://xenon.stanford.edu/ lswartz/paperclip/paperclip.pdf

[3] TechSophia.*Jdu do školy!.*[software].[přístup 3. října 2015].Dostupné z:
`https://play.google.com/store/apps/detailsid=cz.techsophia.schoolreadiness&hl=cs`

[4] www.education.com. *I'm Thinking of a Number.*[software].[přístup 16. října 2015].Dostupné z:
`http://www.education.com/games/thinking-of-a-number/`

[5] Outfit7.*My Talking Tom.*[software].[přístup 16. října 2015].Dostupné z:
`https://play.google.com/store/apps/details?id=com.outfit7.mytalkingtomfree`

[6] Zakeh.*Pou.*[software].[přístup 16. října 2015].Dostupné z:
`https://play.google.com/store/apps/details?id=me.pou.app`

[7] WROBLEWSKI, Luke. *Touch-based App Design for Toddlers.* 18.srpna 2010.[online].[přístup 24. října 2015]. Dostupné z:
`http://www.lukew.com/ff/entry.asp?1179`

[8] gitlab.fit.cvut.cz.*Přesmyčky.*[online].[přístup 14. prosince 2015]. Dostupné z: `https://gitlab.fit.cvut.cz/buresm11/Dracek3`

[9] Edux.*BI-SP1.*[online].[přístup 14. prosince 2015].Dostupné z:
`https://edux.fit.cvut.cz/courses/BI-SP1/`

[10] Badlogic Games.*Scene2d, Event Listener.*[online].[přístup 21. prosince 2015]. Dostupné z:
`https://libgdx.badlogicgames.com/nightlies/docs/api/com/badlogic/gdx/scenes/scene2d/EventListener.html`

[11] Badlogic Games.*libGDX, Timer.*[online].[přístup 10. ledna 2016]. Dostupné z:
`https://libgdx.badlogicgames.com/nightlies/docs/api/com/badlogic/gdx/utils/Timer.html`

[12] Badlogic Games.*libGDX, GestureListener.*[online].[přístup 10. ledna 2016]. Dostupné z:
`https://libgdx.badlogicgames.com/nightlies/docs/api/com/badlogic/gdx/input/GestureDetector.GestureListener.html`

[13] Badlogic Games.*Scene2d* [software].[přístup 14. prosince 2015] Dostupné z: https://github.com/libgdx/libgdx/wiki/Scene2d

[14] COON, Dennis, Mitterer O. John. *Psychology: Modules for Active Learning*. Brock University.13th edition.2012.832p

[15] Nauert, Rick. *Emotion Can Heighten Memories.* 21.srpna 2012.[online].[přístup 7. listopadu 2015]. Dostupné z:
`http://psychcentral.com/news/2012/08/21/emotion-can-heighten-memories/43437.html`

[16] Dean, Jeremy. *Children's Memories Work In A Surprising Way.* 16.října 2015.[online].[přístup 7. listopadu 2015]. Dostupné z:
`http://www.spring.org.uk/2015/10/the-surprising-way-that-childrens-memories-work.php`

[17] Eye Can Learn. *Visual Closure.*[online].[přístup 9. listopadu 2015].Dostupné z:
`http://www.eyecanlearn.com/perception/closure/`

[18] Kid Sense. *Visual Perception.*[online].[přístup 7. listopadu 2015]. Dostupné z:
`http://www.childdevelopment.com.au/visual-processing`

[19] Badlogic Games.*libGDX* [software].[přístup 23. listopadu 2015] Dostupné z: http://libgdx.badlogicgames.com/index.html

[20] Google.*Android Studio* [software].[přístup 14. prosince 2015] Dostupné z: http://developer.android.com/sdk/index.html

[21] Oracle.*Java SE Development Kit 8* [software].[přístup 14. prosince 2015] Dostupné z: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

[22] Thomas.*Sticky Butterscotch UI Kit.*[online].[přístup 21. prosince 2015]. Dostupné z: `https://dribbble.com/shots/306239-Sticky-Butterscotch-UI-Kit-Free-PSD`

[23] Badlogic Games.*GDX Texture Packer* [software].[přístup 7. ledna 2016] Dostupné z: https://code.google.com/p/libgdx-texturepacker-gui/

[24] Badlogic Games.*Hiero* [software].[přístup 7. ledna 2016] Dostupné z: https://github.com/libgdx/libgdx/wiki/Hiero

Příloha **A**

Seznam použitých zkratek

GUI Graphical User Interface

CLI Command Line Interface

UI User Interface

JSON JavaScript Object Notation

BI-SP1 Softwarový týmový projekt 1

ID Identity

GPS Global Positioning System

HTML5 HyperText Markup Language 5

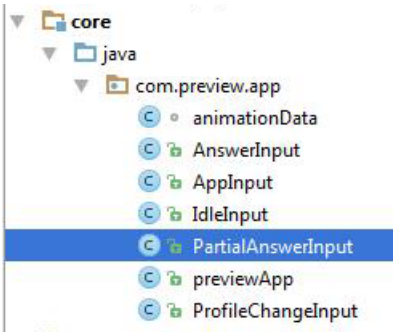
UX User Experience

Příloha B

Ukázková implementace třídy vstupu

Pro detailní vysvětlení funkce třídy a návod na implementaci třídy nové se tato kapitola bude věnovat názorné implementaci jednoho ze čtyř implementovaných druhů vstupů, přesněji třídy částečné odpovědi.
Tento typ vstupu reprezentuje případ, kdy uživatelovu odpověď ve hře nelze klasifikovat jako zcela dobrou nebo zcela špatnou, ale lze určit přesný procentuální výsledek, tedy 0 až 100%.
Tato třída bude navíc mít jeden parametr s vyšší prioritou, čas rychlé odpovědi, který bude vyhodnocen pokud uživatel odpoví dříve, než vyprší stanovený čas.

B.1 Tvorba a registrace třídy



Obrázek B.1: Struktura projektu

V adresáři *core*, ve složce aplikace uvnitř frameworku se nejprve vytvoří .java soubor třídy požadovaného jména, v našem případě „*PartialAnswerInput*“. Tento adresář obsahuje všechny součásti kódu nezávislé na platformě a

provádí se v něm většina potřebné práce.
Tato třída musí nutně dědit od mateřské třídy AppInput, která je použita pro uchování dat v jedné struktuře a pro správnou funkci aplikace musí implementovat všechny nezbytné metody. Výsledná kostra třídy poté vypadá následovně

```
public class PartialAnswerInput extends AppInput{


public void initUI(Skin skin, List ref, float worldWidth,
    float worldHeight, final ArrayList<animationData> ad,
    BitmapFont fontRef){     }

public void destroyUI(){     }

public void setupMainUI(Stage stage){    }

public void setupSettingsUI(Stage stage){     }

public void clearUI(Stage stage){     }

public void drawUIText(Batch batch){     }

public void drawUISettingsText(Batch batch, float worldWidth,
     float worldHeight){     }

public void selectCurrentAnimation(){     }

public void setCurrentAnimation(int id){     }

public int requestAnimationId(){ return -1; }

}
```

Takto předpřipravenou třídu je poté nutno zaregistrovat do načítací funkce jádra aplikace, aby bylo možné správně přetypovat načtené informace. Toto se provede ve funkci *loadInputsFromJSON()* přidáním příslušné části do načítacího for cyklu (obrázek B.2). Tento kód prochází data (třídy), která jsou ze souboru načtena v neznámém formátu *JsonValue* a pomocí proměnné obsažené uvnitř mateřské třídy AppInput (*„type“*), uložené v konfiguračním souboru na indexu 1 (tedy druhý parametr, např. „type: answerInput“) určí správný druh třídy a jako takovou ji uloží do dynamického pole inputData. Veškeré operace nad těmito třídami se poté volají z tohoto pole.

B.2 Vnitřní proměnné

Třída obsahuje dva typy dat. První jsou proměnné uchovávající informace o nastavení. Tato data jsou podstatná a přejeme si je uložit ukládat do konfiguračního souboru.
Druhý typ dat jsou pracovní proměnné jako jsou třídy obsahující ovládací prvky (Slider, Button) a reference na další prvky či data. Ukládání těchto údajů je nežádoucí a je třeba je před uložením v metodě destroyUI nastavit na null, poté budou ignorována. Všechny tyto proměnné jsou po uložení opět obnoveny znovuzavoláním metody initUI.
Proměnné použité v tomto příkladu jsou následující

```
Slider timeElapsed;
Slider correctness;
Slider fastAnswerSlider;
SelectBox options;
SelectBox intervalSelect;

List animationListReference;
BitmapFont fontReference;
ArrayList<animationData> animationDataReference;

int fastAnswerTime;
int fastAnswerAnimationId;
int interval1AnimationId;
int interval2AnimationId;
int interval3AnimationId;
int interval4AnimationId;
int interval5AnimationId;
int interval6AnimationId;
int interval7AnimationId;
ArrayList<String> intervals;        // 1 - 7
```

Jedná se (od shora) o ovládací prvky, pracovní reference a data třídy.
Položky Slider (posuvník) a SelectBox (rozkládací seznam) jsou prvky uživatelského rozhraní poskytnuté součástí frameworku, Scene2d.ui. V případě potřeby lze volit z množství dalších možností.
Reference odkazují na připravenou třídu fontu, tabulku dostupných animací (pravá tabulka z obrazovky settings) a pole obsahující jméno a id dostupných animací.
Data třídy nakonec obsahují informace o času pro animaci rychlé odpovědi a její id a pole intervalů načtených z konfiguračního souboru a animace k ním přiřazené. Třída uložení informací až pro 7 intervalů, tento počet však může být i nižší. Podmínkou jejich výběru přímá návaznost intervalů po sobě a zakončení na 100% (celkový rozsah 0-100%). Vzhledem k tomuto formátu byl zvolen způsob zápisu pomocí jedné číselné hodnoty obsahující vrchní konec intervalu (interval vždy začíná tam kde předchozí skončil + 1).

Zápis v konfiguračním souboru pak může vypadat například takto

```
intervals: [ 9, 50, 80, 100 ]
```

pro čtyři intervaly o hodnotách

```
0 – 9
10 – 50
51 – 80
81 – 100
```

Těmto intervalům je poté možné přiřadit animaci v závislosti na jejich pořadí (1 - 7).

```
for(JsonValue d : rawData){    // Determine type and save as
    if(d.getString(1).equals("mainInput")){
        inputData.add(json.readValue(AppInput.class, d));
        continue;
    }
    if(d.getString(1).equals("idleInput")){
        inputData.add(json.readValue(IdleInput.class, d));
        continue;
    }
    if(d.getString(1).equals("answerInput")){
        inputData.add(json.readValue(AnswerInput.class, d));
        continue;
    }
    if(d.getString(1).equals("profileChangeInput")){
        inputData.add(json.readValue(ProfileChangeInput.class, d));
        continue;
    }
    if(d.getString(1).equals("partialAnswerInput")){
        inputData.add(json.readValue(PartialAnswerInput.class, d));
        continue;
    }

}
```

Obrázek B.2: Cyklus zodpovědný za přiřazení správné třídy

B.3 Implementace metod

initUI
Metoda zodpovědná za inicializaci a nastavení proměnných. Nejprve jsou nastaveny předané reference jako

```
fontReference = fontRef;
animationListReference = ref;
animationDataReference = ad;
```

a poté vytvořeny a nakonfigurovány prvky rozhraní. Vzhledem k rozsáhlosti a podobnosti kódu ukáže tento příklad pouze nastavení jednoho z posuvníků.

Každý grafický prvek rozhraní nejdříve vyžaduje nastavení grafického vzhledu elementu. Tato práce k tomuto účelu využívá načtení předpřipravených textur. Tyto textury jsou načtené do posuvníku příbuzné třídy zvané SliderStyle pomocí v parametru předané třídy Skin uchovávající grafická data elementů.

```
Slider.SliderStyle sliderStyle = new Slider.SliderStyle();
sliderStyle.background = skin.getDrawable("sliderBg2");
sliderStyle.knob = skin.getDrawable("sliderKnob2");
sliderStyle.knobBefore = skin.getDrawable("knobBefore2");
```

Tento skin je následně použit k vytvoření samotného posuvníku (parametry: rozsah, krok, vertikální, styl).

```
fastAnswerSlider = new Slider (0.0f, 60.0f, 1.0f, false,
    sliderStyle);
fastAnswerSlider.setSize(270, 20);
fastAnswerSlider.setPosition(worldWidth / 2 -
    fastAnswerSlider.getWidth() / 2, 450);
fastAnswerSlider.setValue((float) fastAnswerTime);
```

Nakonec je k elementu přiřazen listener, který při změně posuvníku provede požadovaný kód

```
fastAnswerSlider.addListener(new ChangeListener() {
    @Override
    public void changed(ChangeEvent event, Actor actor) {
            fastAnswerTime = (int) fastAnswerSlider.getValue();
    }
});
```

Takovýchto listenerů existuje více druhů a lze si vybrat dle požadovaných vlastností (např. Click, Change, Drag). Více informací viz. dokumentace [10].

Ostatní elementy jsou v metodě nastaveny stejným způsobem.

destroyUI
Jak již bylo zmíněno, metoda se volá před uložením objektu a nastavuje nepotřebné hodnoty na null.

```
public void destroyUI(){

  timeElapsed = null;
  correctness = null;
  fastAnswerSlider = null;
  options = null;
  intervalSelect = null;

  animationListReference = null;
  fontReference = null;
  animationDataReference = null;

}
```

setupMainUI / setupSettingsUI
V této části se do scény přidají v daný okamžik požadované prvky uživatelského rozhraní. Ty jsou pro přehlednost přiřazeny do jedné skupiny, což urychluje jejich odstranění ze scény. Obě metody se liší pouze v názvech prvků.

```
public void setupMainUI(Stage stage){
   Group uiElements = new Group();
   uiElements.addActor(timeElapsed);
   uiElements.addActor(correctness);
   uiElements.setName("uiElements");
   stage.addActor(uiElements);
}
```

clearUI
Metoda vymaže v současné chvíli používané prvky rozhraní ze scény (skupinu uiElements). Lze zavolat jak pro hlavní, tak pro nastavovací obrazovku.

```
public void clearUI(Stage stage){
   Array<Actor> stageActors = stage.getActors();
   for(int i=0; i < stageActors.size; i++){
      Actor a = stageActors.get(i);
      if(a.getName() != null && a.getName().equals("
          uiElements")){
         a.remove();
         break;
      }
    }
}
```

drawUIText / drawUISettingsText
Vykreslí text pro požadovanou obrazovku aplikace. V tomto případě se vykreslují současné hodnoty posuvníků na zadaných souřadnicích.

```
public void drawUIText(Batch batch){
fontReference.draw(batch, "Time since last answer: " + String
    .valueOf((int) timeElapsed.getValue()) + " s", 15, 267);

fontReference.draw(batch, "Correct at: " + String.valueOf((
    int) correctness.getValue()) + " %", 15, 207);

}
```

selectCurrentAnimation
V tabulce animací na obrazovce nastavení změní vybranou animaci v závislosti na současných nastaveních a stavu rozhraní. Metoda se volá vždy, když je provedena změna v rozhraní na obrazovce nastavení. Rozhodnutí o vybrané animaci je provedeno pomocí sekvence if příkazů (v příkazu zkrácené). Tato animace je nakonec pomocí zvoleného id vyhledána a nastavena.

```
public void selectCurrentAnimation(){
    int aniToSelect = -1;
    if(options.getSelectedIndex() == 0) {
        aniToSelect = fastAnswerAnimationId;
    }
    if(options.getSelectedIndex() == 1) {
    .
    .
    .
    }

    for(int i = 0; i < animationDataReference.size(); i++) {
        if(animationDataReference.get(i).getId() == aniToSelect
            ) {
            animationListReference.setSelected(
                animationDataReference.get(i).getName());
            break;
        }
    }
}
```

setCurrentAnimation
Metoda zavolaná ve chvíli změně v tabulce animací v položce nastavení. Změní animaci (vyhodnocenou podle současného stavu rozhraní) na nové id. Příklad je opět zkrácený.

```
public void setCurrentAnimation(int id){
    if(options.getSelectedIndex() == 0) {
        fastAnswerAnimationId = id;
    }
    if(options.getSelectedIndex() == 1) {
        if(intervalSelect.getSelectedIndex() == 0){
            interval1AnimationId = id;
        }
      .
      .
      .
     }
}
```

requestAnimationId
Podobná předchozím dvěma, metoda se zavolá při stisknutí tlačítka přehrát, vyhodnotí stav ovládacích prvků na hlavní obrazovce a vrátí id výsledné animace pro přehrání. Ukázka kódu zkrácená. V případě neúspěšného vyhodnocení vrátí hodnotu -1 (žádná animace).

```
public int requestAnimationId(){
    if(((int) timeElapsed.getValue()) <= fastAnswerTime){
        return fastAnswerAnimationId;
    }

    int corectness = (int) correctness.getValue();
    if(corectness >= 0 && corectness <= Integer.valueOf(
        intervals.get(0))){
        return interval1AnimationId;
    }
    .
    .
    .
    return -1;
}
```

B.4 Záznam do konfiguračního souboru

Nový typ vstupu se do nakonec může do aplikace přidat pomocí jeho zapsání do konfiguračního souboru. Jeho formát vypadá následovně a závisí na pořadí proměnných (položka class je pomocná informace serializéru, další tři parametry obsahují informace pro nadtřídu AppInput - typ, id, text pro výpis názvu). Zápis je ve formátu JSON, frameworkem implementovaný serializér však nedodržuje některé normy (proměnné a hodnoty v uvozovkách a čárky na konci řádku). Takovýto zápis se v pořádku načte a zpracuje.

```
{
    class: com.preview.app.PartialAnswerInput
    type: partialAnswerInput
    inputId: 6
    btnText: Partial Answer
    fastAnswerTime: 45
    fastAnswerAnimationId: 3
    interval1AnimationId: 4
    interval2AnimationId: -1
    interval3AnimationId: 3
    interval4AnimationId: 1
    interval5AnimationId: 1
    interval6AnimationId: 2
    interval7AnimationId: 3
    intervals: [ 9, 50, 80, 100 ]
}
```

Příloha C

Obsah přiloženého CD

- `readme.txt` ..... Stručný popis obsahu CD
- `Runnable` ..... Adresář se spustitelnou formou implementace
  - `Android` ..... Soubor aplikace pro systém Android
  - `Desktop` ..... Java verze aplikace pro desktopové systémy
- `Source` ..... Zdrojové soubory
  - `readme.txt` ..... Instrukce ke kompilaci aplikace
  - `Application` ..... Zdrojové kódy implementace
  - `Thesis` ..... Zdrojová forma práce ve formátu LaTeX
- `Text` ..... Text práce
  - `BP_Podaný_Pavel_2016.pdf` ..... Text práce ve formátu PDF