České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Diplomová práca

Soubor úloh pro detekci a léčbu dyslexie, dysgrafie, dyskalkulie

Bc. Ján Adamček

Vedúci práce: Ing. Petr Novák, Ph.D.

Študijný program: Informatika pro magisterské studenty

Obor: Webové a softwarové inženýrství (magisterský)

Poďakovanie

Týmto by som sa chcel veľmi poďakovať vedúcemu práce Ing. Petrovi Novákovi, Ph.D., za poskytnuté rady, pomoc a vynikajúce odborné vedenie pri tvorbe tejto práce. Ďalej by som sa chcel poďakovať vedúcemu katedry softvérového inžinierstva Ing. Michalovi Valentovi, Ph.D., za pomoc pri problémoch v procese schvaľovania práce a pri schválení a pomoci pri registrácii oponenta tejto práce. V neposlednej rade by som chcel poďakovať mojim rodičom a bratovi za veľkú podporu a pomoc pri štúdiu.

Prehlásenie

Prehlasujem, že som prácu vypracoval samostatne a použil som iba podklady uvedené v priloženom zozname.

Nemám závažný dôvod proti použitiu tohto školského diela v zmysle §60 Zákona č. 121/2000 Sb., o autorskom práve, o právach súvisiacich s autorským právom a o zmene niektorých zákonov (autorský zákon).

V Prahe dňa 25.6.2014 .....

Abstract

The present thesis focuses on detection and treatment of three selected specific learning difficulties, namely dyslexia, dysgraphia and dyscalculia. The goals of this thesis were creating solution and implementation of application that covers 5 selected representative tasks for diagnosis and treatment of these specific learning difficulties, creating extensions for used framework, which will properly show processed data from tasks and test the application in environment for which it is designed.

Abstrakt

Táto práca sa zaoberá otázkou detekcie a liečby formou cvičení troch vybraných špecifických porúch učenia a to dyslexie, dysgrafie a dyskalkulie. Cieľom práce bolo zoznámenie sa s týmito poruchami a následný návrh a implementácia aplikácie, ktorá by zastrešovala 5 vybraných reprezentatívnych úloh na detekciu a liečbu práve týchto špecifických porúch učenia, rozšírenie použitého frameworku o vhodné zobrazenie už spracovaných dát z úloh a otestovanie aplikácie v prostredí, pre ktoré je aplikácie určená.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam programového kódu

1 Úvod

V súčasnej dobe rozvoja informačných technológií sa veľa bežných činností, ktoré boli vykonávané ručne, alebo v papierovej podobe prípadne v slovnej podobe, snažíme previesť do elektronickej verzie, ktorá ponúka účinnejšie a hlavne rýchlejšie riešenie a spracovanie, či už zo strany úspory miesta, prenášania z jedného miesta na druhé, univerzálnosti v rámci konfigurácii, časovej úspory, finančnej úspory, flexibility práce a ďalších výhod, ktoré sú so svetom informačných technológií neoddeliteľne spojené. Táto práca sa ale nezaoberá všetkými výhodami, ktoré takáto inovácia ponúka. Zameriava sa hlavne na univerzálnosť a úsporu z hľadiska času, flexibility a prenositeľnosti, teda použiteľnosti na ľubovoľnom mieste.

Vyššie spomenutý trend určite neobchádza ani zdravotnícke zariadenia a to hlavne pri detekcii alebo liečbe (rehabilitácii) širokého spektra rôznych chorôb alebo porúch, ktoré je nutné včas identifikovať a následne vhodne liečiť. Medzi spomínané poruchy patria pohybové poruchy, zrakové poruchy, rôzne typy kognitívnych porúch vzniknuté či už z dôvodu vyššieho veku, alebo ako následok ťažkého úrazu a rovnako tak špecifické poruchy učenia (SPU), kam patrí napríklad dyslexia, dysgrafia, dyskalkulia a iné. Všetky spomenuté poruchy majú jednu spoločnú vec a to je ich včasná a presná detekcia (odhalenie) a následná optimálna liečba alebo zmierňovanie ich účinkov (rehabilitácia). Táto práca sa ale nebude zaoberať všetkými vymenovanými problémami alebo poruchami, ale zameria sa konkrétne na tri vybrané špecifické poruchy učenia a to dyslexiu, dysgrafiu a dyskalkuliu.

Detekcia a liečba porúch pacienta pomocou počítačových aplikácií má nesporné výhody. Lekári museli v minulosti vytvárať úlohy a testy na detekciu alebo rehabilitáciu na papier, čo u niektorých úloh, hlavne tých, pri ktorých mal pacient za úlohu niečo dokresliť, alebo dopísať, značne komplikovalo rýchlu a opakovanú použiteľnosť týchto úloh. Úlohy museli byť, buď vytvorené znova alebo boli vytvorené dopredu, bohužiaľ vo väčšine prípadov rovnaké kópie. Takýto spôsob tvorby úloh je značne nepraktický, či už z hľadiska času, ktorý trvá tvorba takejto úlohy ale aj z hľadiska množstva papierov, ktoré je nutne spotrebovať a následne aj uschovať v prípade budúcich potrieb lekára. Na druhú stranu, pokiaľ pacient opakuje takmer rovnaký test niekoľkokrát po sebe, tak jeho účinnosť sa pomaly vytráca, pretože ho vlastne rieši skôr po pamäti, čo nie je žiaduce.
Ďalšou podstatnou výhodou elektronických úloh je rozmanitosť a možnosť okamžitej konfigurácie podľa potreby. Pri papierovej podobe úloh je konfigurovateľnosť skoro nemožná

alebo veľmi pracná. To nedovoľuje lekárovi prispôsobiť úlohu počas samotného vyšetrenia alebo rehabilitácie bez vytvárania celkom nových úloh na separátne listy papierov. Naopak pri elektronickej verzii, je adaptácia úlohy len otázkou pár nastavení, pričom tieto nastavenia, môže lekár meniť neustále i počas priebehu testovania a tak prispôsobovať napríklad obtiažnosť alebo vzhľad úlohy podľa aktuálnej potreby a hlavne vekových schopností pacienta.

Ďalšou nespornou výhodou elektronickej verzie úloh, je samotné vyhodnotenie výsledkov. Pri papierovej verzii musí lekár vyhodnocovať zistené poznatky ručne a v prípade, že si výsledky z úloh nezaznamená, nemôže sa k nim vrátiť po dlhšom čase, poprípade ich musí uchovávať, čo určite zaberá nezanedbateľné miesto. Navyše spracovanie takýchto výsledkov môže trvať nemalý čas, čo lekárovi neumožňuje rýchly návrh diagnózy, poprípade ohodnotiť stav pacienta počas rehabilitácie. Elektronická forma testovania umožňuje zbierať rôzne dáta podľa potreby lekára a následne ich odosielať na server, kde prejdú často i takmer automatickým spracovaním. Takto spracované dáta si môže lekár zobraziť vhodnou formou v elektronickej podobe (jednoducho na obrazovke monitoru), čo mu môže značne pomôcť pri určovaní diagnózy (pri detekcii poruchy) alebo zistení súčasného stavu (pri rehabilitácii) pacienta v reálnom čase (kedykoľvek). Takto môže lekár okamžite reagovať i na nepredpokladanú zmenu stavu pacienta. Lekárovi takéto riešenie dovoľuje vidieť aj staršie namerané výsledky z úloh, ktoré pacient riešil v minulosť a podľa nich patrične upraviť aktuálny postup pri rehabilitácii.

Veľa pacientov musí v dnešnej dobe cestovať za odborným lekárskym vyšetrením do vzdialenejších miest, čo môže pre značné množstvo pacientov ovplyvňovať ich psychický stav, ktorý je vždy kľúčovým faktorom pri liečbe, alebo samotné finančné hľadisko z nutnosti častého dochádzania do vzdialenejších miest, pričom musí pacient vynechávať pracovné povinnosti, najčastejšie práve z dôvodu rehabilitácií. Financie ale nie sú predmetom skúmania tejto práce. Dôležitejší faktor je psychika pacienta, ktorá môže zohrať veľkú rolu pri samotnej liečbe. Už práve nutnosť časovo dlhého cestovania môže vplývať na psychiku pacienta a ak sa k tomu ešte pridá stres z návštevy lekára tzv. syndróm bieleho plášťa, ktorým môže v dnešnej dobe trpieť dosť ľudí, je takáto nutnosť návštevy skôr stresujúca ako príjemná a účinná, alebo dokonca prospešná. (Syndróm bieleho plášťa je nepriaznivý psychický stav pacienta, v ktorom sa pacient môže nachádzať počas návštevy lekára. Prejavuje sa úzkosťou, zvýšením krvného tlaku, zvýšením nervozity, potením, neistotou a obavami. Takýto psychický stav pacient nedokáže vedome ovplyvniť [1].) Už spomenutý samotný stres nemusí dobre vplývať na výsledok vyšetrení a liečby. Preto je dosť podstatným faktorom možnosť znížiť také návštevy u lekára, ktoré nie sú nevyhnutne významné pri samotnej liečbe pacienta. Medzi takéto návštevy určite môže patriť

nutnosť rehabilitačných cvičení bez potreby dozoru samotného lekára. Z tohto dôvodu má elektronická podoba úloh veľkú výhodu v možnosti prístupu k úlohám, aj z iného miesta než je ordinácia lekára. Toto umožňuje pacientom vykonávať úlohy z domova, čo určite prispieva k zníženiu stresového faktora z cesty, alebo z množstva nutných návštev u lekára. Pacient vykonáva rehabilitačné úlohy, ktoré mu lekár určil, v pohodlí svojho domova a nie v preňho cudzom prostredí, čo môže zlepšovať psychický stav pacienta. Zlepšenie psychického stavu pacienta môže mať, ale nie nutne, významný vplyv na liečebný proces a na jeho dĺžku, hlavne u detí, ktoré zrejme najčastejšie pociťujú stres z návštevy lekára. Je ale potrebné zdôrazniť, že takéto domáce cvičenia nenahrádzajú plnohodnotnú rehabilitáciu pod dozorom lekára, ale ide len o doplnkovú a aktívnu rehabilitáciu k hĺbkovým kontrolám u samotného lekára a možnosti zníženia menej dôležitých lekárskych návštev. Lekár nemá pri takomto spôsobe domácej rehabilitácie okamžití prístup k dátam, ktoré sa musia najskôr namerať v domácom prostredí, potom odoslať na spracovanie a nakoniec vyhodnotiť. Veľkou výhodou však môže byť, že tieto dáta nemusia byť skreslené z dôvodu stresov pacienta z návštevy u lekára a tak môžu lekárovi vytvoriť iný pohľad na priebeh samotnej liečby. Dáta je možné následne porovnávať s nameranými dátami v ordinácii alebo so staršími dátami, ktoré boli namerané u pacienta v minulosti. Tieto poznatky môžu lekárovi značne pomôcť v úprave pribiehajúceho liečebného procesu pacienta.

Aby mohli byť všetky vymenované výhody splnené a dodržané, je potrebné vytvoriť spoločné aplikačné prostredie, ktoré by umožňovalo zoskupiť takéto úlohy tak, aby boli pokiaľ možno rovnako a súčasne jednoducho ovládateľné pre pacientov, ľahko konfigurovateľné pre obslužný personál a takisto jednoduché na vyhodnotenie.

1.1 Ciele práce

Ciele tejto práce môžeme rozdeliť na dva väčšie celky, a to na teoretickú (takmer výskumnú) časť a na praktickú (iba implementačnú) časť.

V teoretickej časti bolo potrebné zoznámiť sa s definíciou špecifických porúch učenia so zameraním na tri konkrétne špecifické poruchy a to dyslexiu, dsygrafiu a dyskalkuliu. Pri každej špecifickej poruche, bolo potrebné zoznámiť sa nie len s jej slovnou definíciou, ale súčasne s možnosťami diagnostiky (samozrejme iba z pohľadu tejto práce). Tieto informácie boli potrebné z dôvodu bližšieho zoznámenia sa z týmito poruchami, pre použitie pri neskoršom výbere a

implementácii vhodných úloh a testov. Po zoznámení sa s týmito konkrétnymi špecifickými poruchami, bolo potrebné vytvoriť prehľad úloh a testov vhodných na testovanie alebo rehabilitáciu týchto porúch. Zoznam týchto úloh je možné nájsť v prílohe B.

Implementačná časť sa zaoberá návrhom a tvorbou samotných úloh (testov), ktoré reprezentujú vybrané tri špecifické poruchy učenia.
Ako prvý bod implementačnej časti bolo využitie a prípadné doplnenie už exitujúceho frameworku pre tvorbu všeobecných testov (poskytnutým od vedúceho práce), tak aby bolo možné jeho pomocou jednoducho vytvoriť reprezentatívnu vzorku úloh (testov) určených na testovanie alebo rehabilitáciu vyššie spomenutých troch špecifických porúch učenia.
V ďalšom bode bolo potrebné vytvoriť konkrétnu sadu vybraných testov. Pri výbere bola braná do úvahy hlavne vhodnosť úloh pre vybrané konkrétne špecifické poruchy učenia a cieľovú skupinu užívateľov (hlavne detí). Následne bol vytvorený návrh na tvorbu týchto vybraných úloh. Pri návrhu bolo potrebné brať do úvahy nasledujúce faktory:

- **Možnosť ovládania** - tu bolo potrebné myslieť nie len primárne na ovládanie myšou, ale taktiež aby bolo možné využitie dotykových monitorov a prípadne tabletov.
- **Konfigurovateľnosť** - aby mohla byť úlohy jednoducho konfigurovateľné alebo rozšíriteľné o nové zadanie aj neodborným užívateľom.
- **Adaptatívnosť** - aby bolo možné úlohu prispôsobiť potrebnej obtiažnosti.
- **Cieľová skupina** – aby boli úlohy vhodne, jednoducho, zrozumiteľne a zábavne spracované podľa špecifickej cieľovej skupiny. Cieľové osoby sú najčastejšie malé deti a to hlavne v predškolskom veku.

Ďalej bolo potrebné zorganizovať vytvorené testy do vhodných skupín, ktoré budú slúžiť na pilotné otestovanie funkčnosti, ovládateľnosti, konfigurovateľnosti a hlavne správnej činnosti.
Ako posledný bod implementačnej časti bolo potrebné zabezpečiť, aby sa z implementovaných úloh dali zbierať a zhromažďovať dáta, ktoré sú potrebné pre hodnotenie vypracovaných úloh lekárom. Po zozbieraní nameraných dát počas chodu úlohy, sa dáta odosielajú na server, kde sú uložené a pripravené pre ďalšie podrobné hodnotenie. Samotné spracovanie a hodnotenie uložených dát nie je témou tejto práce. Takto namerané a hlavne už spracované dáta, je následne potrebné vhodným spôsobom zobraziť, tak aby boli zrozumiteľné a rýchlo pochopiteľné lekárom, ktorý na základe týchto dát môže upraviť postup rehabilitácie, alebo mu môžu dopomôcť k stanoveniu diagnózy pacienta. Z tohto požiadavku vyplýva posledná časť tohto

bodu, a to rozšíriť už existujúci framework o vhodné možnosti zobrazenia hodnotenia z úloh, aby boli jednoduché a hlavne zrozumiteľné. Väčšinou ide prevažne o grafy alebo jednoduché tabuľky, ktoré predstavujú vhodnú a zrozumiteľnú formu zobrazenia dát, pre ich porovnávanie so staršími nameranými výsledkami. V tomto prípade je snaha sa vyhnúť presným číselným hodnotám, ktoré pre bežnú / častú kontrolu stavu pacienta nie sú nijako dôležité.

1.2 Štruktúra práce

Práca je rozdelená na jednotlivé časti, ktoré sa venujú jej vytýčeným cieľom. Tieto časti sa dajú zoskupiť na dva veľké celky, tak ako sme rozdelili ciele práce v predošlej kapitole, a to na teoretickú (výskumnú) časť a praktickú (implementačnú) časť.

V prvej časti sa práca zameriava na popísanie teoretických definícií špecifických porúch učenia so zameraním na tri konkrétne vybrané poruchy: dyslexia, dysgrafia, dyskalkulia. Hlavná kapitola definuje všeobecný pojem „Špecifické poruchy učenia“ za pomoci definícií odborníkov, ktorý sa tejto téme venujú, vplyv špecifických porúch na vývoj dieťaťa a jeho zaradenie do budúceho života a na diagnostiku špecifických porúch s popisom testovacích metodík určených na odhalenie týchto porúch.
Po všeobecnom zoznámení sa so špecifickými poruchami učenia, sa práca bližšie zameriava na vybrané tri konkrétne špecifické poruchy, ktoré už boli spomenuté vyššie. Práca je tu rozčlenená do troch podkapitol, kde každá popisuje jednu špecifickú poruchu. Prvá kapitola bližšie približuje pojem „Dyslexia“ pomocou jej definície, ktorá opisuje problémy postihnutého jedinca touto poruchou, a popísaniu faktorov, ktoré sú potrebné, aby bolo dieťa schopné prijímať informácie z čítaného textu. Ďalšia podkapitola sa bližšie venuje pojmu „Dysgrafie“, definuje ho a popisuje problémy, ktoré táto porucha spôsobuje. Posledná kapitola popisuje pomocou definície pojem „Dyskalkulie“, uvádza faktory, ktoré sú touto poruchou zasiahnuté a približuje rozdelenie dyskalkulie na 6 hlavných typov na základe toho čo daný typ dyskalkulie predstavuje. Je potrebné zdôrazniť, že táto práca sa nezaoberá s teóriou okolo špecifických porúch učenia do hĺbky. Ide len o priblíženie a definovanie týchto porúch na teoretickej a nie výskumnej báze.

Zvyšok práce sa venuje samotnej softvérovej implementácii a rozšíreniu použitého frameworku. V prvej časti tohto celku sa práca venuje popisu súčasného stavu testovania a rehabilitácie špecifických porúch učenia so zameraním na vybrané tri špecifické poruchy v elektronickej forme. Ide o popis súčasných možností v testovaní alebo rehabilitácie týchto porúch učenia

pomocou informačných technológií. Ich kladov a záporov, vhodnosti, konfigurovateľnosti a iných vlastností.

Nasleduje časť venovaná analýze a návrhu implementácie so zameraním na návrh riešenia vybranej reprezentatívnej vzorky úloh a na výber implementačného prostredia. Tu sa v prvej časti práca zameriava na popis reprezentačného výberu úloh, ktoré sú vhodné na testovanie alebo rehabilitáciu už vyššie spomenutých špecifických porúch učenia. Bude popísaná ich vhodnosť z hľadiska, lekárskeho ale aj z hľadiska implementačného (konfigurovateľnosť, adaptatívnosť atď.). Z dôvodu použitia frameworku určeného na tvorbu takýchto typov úloh, bude stručne popísaný v analýze aj samotný už vytvorený framework. Návrh aplikácie sa bude sústrediť na jednotlivé úlohy a rozšírenia tohto frameworku, ako budú vytvorené z hľadiska grafického rozhrania, ich vnútorných elementov a zberu dát.

V implementačnej kapitole sa práca bude zaoberať konkrétnejším popisom niektorých vybraných špecifických neštandardných častí implementovaných úloh respektíve rozšírení frameworku.

Ďalšou dôležitou súčasťou práce bude dokumentácia k jednotlivým implementovaným úlohám a rozšíreniam, kde bude popísaný postup ovládania úloh, tvorby vlastného zadania a možnosti konfigurácie jednotlivých úloh.

Záver práce bude venovaný výsledkom pilotného testovania jednotlivých úloh z pohľadu užívateľov a lekárov, popisu nájdených problémov a návrhu ich riešenia do budúcnosti a zhrnutiu celej práce.

1.3 Na čo sa práca nezameriava

Pred samotnou analýzou a návrhom aplikácie je potrebne zdôrazniť také časti, ktoré nie sú súčasťou tejto práce, ale môžu byť jedným zo základných požiadaviek pri implementácii takéhoto projektu. Tieto časti sú spomenuté len z dôvodu informácie o tom, čo táto práca neobsahuje, ale pritom môžu vyvolať otázky, ktoré sú logickými následkami takejto implementácie.

Pri tvorbe aplikácií, ktoré majú za úlohu testovať alebo precvičovať určité funkcie užívateľa a popritom zbierať dáta z tohto testovania, je potrebné tieto dáta odosielať na nejaké vzdialené zariadenie (najčastejšie server), či už z dôvodu spracovania alebo uloženia pre prípad neskoršieho vyhodnotenia. Z tohto dôvodu sa vynára otázka zabezpečenia (osobných) dát, ktoré putujú sieťou na dané vzdialené úložisko. Hlavne v dnešnej dobe je zabezpečenie dát dôležitou súčasťou takéhoto presunu dát na vzdialenejšie servery. Z cieľov tejto práce však vyplýva, že táto práca sa zameriava len na návrh a implementáciu vhodných úloh na testovanie špecifických

porúch učenia a rieši len zber a odoslanie týchto dát na server pomocou vopred definovaného frameworku. Tým pádom otázka zabezpečenia dát ostáva na samotnom frameworku a do tejto práce nie je zahrnutá.

Ďalšou z otázok, ktoré môžu vychádzať z tejto práce je zabezpečenie autorizácie a autentifikácie užívateľov. V bežnej praxi sa pri tvorbe takejto aplikácie, ktorú používa väčšie množstvo užívateľov berie na vedomie identifikácia jednotlivých užívateľov, aby sa vhodným spôsobom separovali a označili namerané dáta, ktoré musia byť uložené a neskôr spracované pod daným identifikačným číslom. Toto sa väčšinou rieši pomocou prihlasovacích údajov užívateľa (meno a heslo). Táto časť sa v tejto práci rieši len z časti identifikácie pacienta pomocou číselného kódu, ktorý sprístupňuje len príslušné úlohy pre daného užívateľa a identifikuje užívateľa a pracovisko lekára. Aj toto je súčasťou použitého frameworku.

Poslednou z vlastností, na ktorú sa táto práca nezameriava je multifunkčnosť v rámci výberu operačného systému, na ktorom daná aplikácia beží. Je to spôsobené tým, že samotná implementácia úloh prebieha pomocou už existujúceho frameworku, ktorý je vytvorený v konkrétnom implementačnom prostredí a využíva jeho knižnice (Microsoft .NET Framework). Ďalším dôvodom je aj to, že pre tvorbu týchto úloh a teda aj samotného frameworku, bol kladený dôraz na výber jazyka, ktorý by plne spĺňal možnosti tvorby daného frameworku a jednotlivých úloh, bez akýchkoľvek zložitých programovacích metód a ktorý by spĺňal jednoduchú možnosť použitia frameworku ako webovej, desktopovej a v dnešnej dobe aj dotykovej aplikácie (C#). Preto nebol kladený dôraz na výber jazyka, ktorý by podporoval multiplatformovú použiteľnosť.

2 Špecifické poruchy učenia (SPU)

Špecifické poruchy učenia (SPU) je pojem, ktorý zoskupuje rôzne poruchy učenia, ktoré zabraňujú v integrácii mentálnych funkcií do jedného učebného cieľa. Ide o skupinu porúch, ktoré majú jeden spoločný element a to, že sa im pripisuje dysfunkcia centrálneho nervového systému (CNS). Takéto poruchy sa najčastejšie vyskytujú u detí, ale môžu poznamenať aj dospelosť jedinca trpiaceho týmito poruchami v detstve [2]. Pokorná vo svojej publikácii *„Teorie a náprava vývojových poruch učení a chování“* píše o nezvratnosti existencie fenoménu špecifických porúch učenia, ktorý môže v určitých prípadoch významne ovplyvniť vzdelávací a osobnostný rozvoj dieťaťa a teda aj jeho budúci život. Rovnako tak môže negatívne ovplyvniť aj rozvoj kognitívnych a intelektuálnych funkcií jedinca [3]. Toto tvrdenie podporuje aj Zelinková, ktorá označuje takéto poruchy učenia ako heterogénnu skupinu porúch, ktoré sa prejavujú pri osvojovaní a používaní reči.
Podľa Národného ústavu zdravia vo Washingtone a expertov Ortnovej spoločnosti znie definícia SPU nasledovne:

„Poruchy učenia sú súhrnným označením rôznorodej skupiny porúch, ktoré sa prejavujú zreteľnými ťažkosťami pri nadobúdaní a používaní takých vedomostí, ako je rozprávanie, porozumenie hovorenej reči, čítanie, písanie, matematické usudzovanie alebo počítanie. Tieto poruchy sú vlastné postihnutému jedincovi a predpokladajú dysfunkciu centrálneho nervového systému. I keď sa porucha učenia môže vyskytovať súbežne s inými formami postihnutia (zmyslové vady, mentálna retardácia, sociálne a emocionálne poruchy) alebo súbežne s inými vplyvmi prostredia (kultúrne zvláštnosti, nedostatočná alebo nevhodná výučba, psychogénne činitele), nie je priamym následkom takýchto postihnutí alebo nepriaznivých vplyvov.“ [4]

Z tejto definície je možné jednoznačne usúdiť, že na tieto poruchy nemajú vplyv žiadne vonkajšie činitele. Podľa definície ide o poruchy, ktorých príčinou je dysfunkcia v centrálnom nervovom systéme. Primárnou príčinou takýchto porúch môže byť tzv. ľahká mozgová dysfunkcia (LMD). Ide o mierne výchylky vo fungovaní centrálneho nervového systému. Je potrebné zdôrazniť, že tieto výchylky nemajú žiadny vplyv na inteligenciu u postihnutého jedinca, ktorý trpí poruchami SPU. Inteligencia je prevažne normálna až nadpriemerná. K LMD väčšinou dochádza v dôsledku malých poškodení na mozgu počas raného vývinového štádia dieťaťa, teda pred pôrodom, počas pôrodu alebo tesne po ňom [4].

Diagnostika SPU sa vykonáva až vo vyššom veku a to v čase okolo 8 rokov. V predškolskom veku je takáto diagnostika náročnejšia. Ako prvý dôvod je ten, že deti sa vo svojich vedomostiach môžu výrazne odlišovať. Každé dieťa môže mať v predškolskom veku rozdielnu inteligenciu. Ďalším dôvodom je, že na dieťa v predškolskom veku nie sú ešte kladené nároky na učenie a školskú prácu. Samotná diagnostika špecifickej poruchy nie je nikdy úplná. Diagnostika sa opiera o poznatky o vývoji a poruchách percepčných a kognitívnych funkcií :

1. Vnímanie, jeho význam pri orientácii v prostredí, zmysluplnosť vnímania [4].
2. Psychomotorický vývoj – vývoj pohybových a motorických aktivít človeka, ktoré sú prejavom jeho psychických funkcií a psychického stavu (úsmev, mávnutie ruky, ochablý postoj atď.) [5].
3. Poruchy vo vnímaní telesnej schémy – napr. viera v seba samého, v partnera, v rodinu atď. [6].
4. Proces pamäti a poruchy zapamätávania si informácií [4].

Pre určenie špecifickej poruchy je najskôr potrebné vylúčiť ostatné známe príčiny zhoršenia školského prospechu u jedinca. Sem môžeme zaradiť zmyslové poruchy (poruchy zraku, sluchu), motorické poruchy, nevhodnosť prostredia (odlišné kultúrne faktory), emocionálne poruchy. Následne je potrebné stanoviť, o akú špecifickú poruchu sa jedná. Diagnostika sa zameriava na 7 typov testovacích metodík [4].

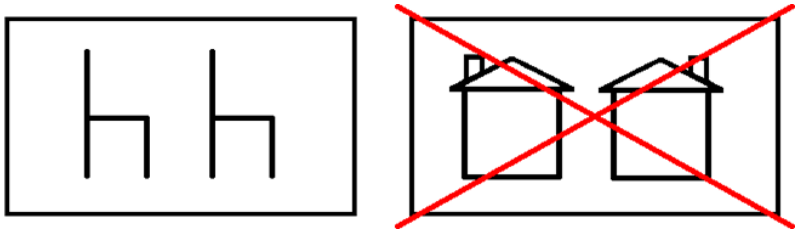
1. **Vyšetrenie čitateľských schopností**. Pri tomto vyšetrení sa hodnotí rýchlosť čítania a porozumenie čítaného textu. Analyzujú sa chyby a správanie, dieťa pri čítaní. Pri testovaní sa používajú normované texty. Rýchlosť čítania sa vyjadruje pomocou tzv. čitateľského kvocientu (ČQ), pomocou ktorého sa stanovuje rýchlosť čítania na základe počtu správne prečítaných slov za prvú minútu čítania. Ak je medzi IQ a ČQ rozdiel vyšší ako 20 bodov, ide o varovný signál, že dieťa môže trpieť špecifickou poruchou [4].
2. **Hodnotenie písomného prejavu**. Tu sa hodnotí hlavne tvar písaného písma, dodržiavanie linearity, správanie pri písaní. Písomný prejav dieťaťa je vhodné zisťovať nie len zo špecificky vypracovaných úloh, diktátov, opisu a prepisu pri vyšetrení, ale aj z jeho školského zošitu. Pri analýze chýb sa kladie dôraz hlavne na komolenie slov, prehadzovanie písmen, zamieňanie si tvarovo alebo zvukovo podobných písmen, vynechávanie alebo nesprávne umiestenie diakritiky, nerozlišovanie mäkkých a tvrdých

slabík, zrkadlové písanie písmen alebo číslic, zamieňanie číslic, nedodržiavanie stĺpcov pri písomnom sčítavaní, odčítavaní atď. [4].

bubon bnbou
budon bubou
dubon bnbon

Obrázok 2.1 - Príklad zámeny tvarovo podobných písmen.

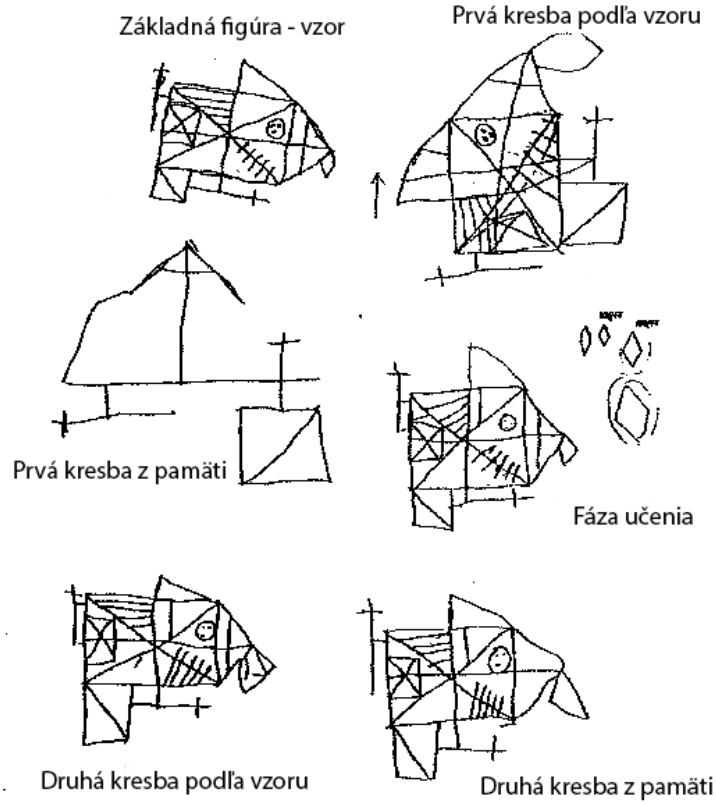
3. **Sluchová analýza a syntéza**. Pri tomto teste musí dieťa určovať počet slov vo vete, vymýšľa vety s daným počtom slov, určuje poradie určeného slova vo vete, rozkladá slová na slabiky, určuje prvú a poslednú slabiku v slove, vymýšľa slová, ktoré začínajú na určenú slabiku, rozkladá slová na hlásky a naopak ich môže z hlások skladať a ďalšie podobné úlohy [7].
4. **Vyšetrenie zrakovej percepcie tvarov**. Ide o skúšku, pri ktorej sa zisťujú príčiny zámeny písmen a číslic. Ide predovšetkým o vodorovné preklopenie. Na toto vyšetrenie slúži Eldfeldtova reverzná skúška. Táto skúška je založená na princípe porovnávania dvoch obrazcov a určenia, či sú sledované obrazce totožné [8].



Obrázok 2.2 - Príklad Eldfeldtovej reverznej skúšky. Áno - obrázky sú totožné (obrázok vľavo), Nie - obrázky nie sú totožné (obrázok vpravo) [8].

5. **Vnímanie priestorovej orientácie**. Ide o test, pri ktorom sa skúma priestorová orientácia v závislosti na zrakovej, sluchovej a kinestetickej percepcii. Vykonáva sa pomocou Žlabového testu orientácie vpravo – vľavo, pri ktorom dieťa dvíha pravú alebo ľavú ruku vo vzťahu k vlastnému telu, alebo u starších detí (od 3 rokov) vo vzťahu k inej osobe [4].
6. **Vyšetrenie predstavy priestorovej orientácie**. Testuje sa pomocou už spomínanej Žlabovej skúšky, alebo u starších detí je možne použiť Reyove komplexné figúry. Princíp

spočíva v tom, že dieťa najskôr obkreslí figúru na čistý papier a následne sa musí pokúsiť z hlavy, bez predlohy, nakresliť tú istú figúru na nový čistý papier. Následne sa dieťaťu vysvetlia vzťahy medzi jednotlivými čiarami kresby a dieťa nakreslí znova danú figúru na nový čistý papier. Nakoniec sa porovná prvá a posledná kresba [4].



Obrázok 2.3 – Príklad testu pomocou Reyovej komplexnej figúry [9].

7. **Vnímanie časovej postupnosti**. Toto vyšetrenie sa prevádza v oblasti vizuálnej a sluchovej. Medzi takéto testy patrí vymenovanie alebo zoradenie dní týždňa, mesiacov v roku, ročné obdobia podľa časovej postupnosti, vymenovanie časových údajov bez nadväzujúceho poradia, nastavovanie času na hodinách podľa pokynu, prevod jednotiek (dni, hodiny, minúty, sekundy), vedieť povedať dnešný dátum, dátum narodenia seba, svojich rodičov a mnoho ďalších [10].

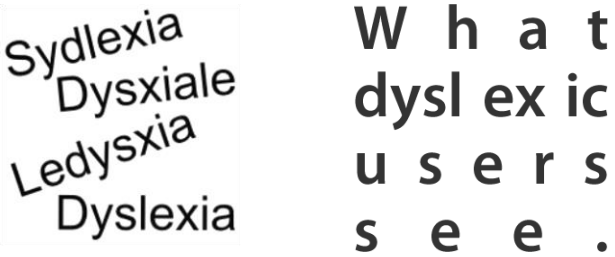
Na základe zamerania sa tejto práce na tri konkrétne špecifické poruchy učenia bude nasledujúca časť venovaná práve týmto trom špecifickým poruchám. Ide o dyslexiu, dysgrafiu a dyskalkuliu.

2.1 Dyslexia

Ide o špecifickú poruchu čítania. Postihnutý jedinec má ťažkosti s porozumením čítaného textu a tým pádom problémy s pochopením výukovej látky preberanej v škole. Nedokáže čítať text plynulo, buď číta príliš pomaly, číta slová po slabikách alebo písmenkách, alebo číta až príliš rýchlo a tým vynecháva písmená v slove, ktorých význam si následne musí domýšľať. Je potrebné zdôrazniť, že dieťa vo väčšine prípadov oplýva priemernou až nadpriemernou inteligenciou [4].
V Českej republike sa pre praktickú diagnostiku a nápravu tejto špecifickej poruchy využíva definícia, ktorú zaviedli Matejček s Langmeierom v roku 1960 a ktorá znie:

„Vývojová dyslexia je špeciálny defekt čítania, podmienený nedostatkom alebo poruchou niektorých primárnych schopností, ktoré skladajú komplexnú schopnosť pre učenie čítania za danej výukovej metódy. Objavuje sa u detí obvykle od samých počiatkov výučby a spôsobuje, že úroveň čítania je trvalo v nápadnom rozpore so zistenou úrovňou intelektuálnych schopností dieťaťa.“ [11]

Z tejto definície je možné povedať, že ide teda o špecifickú poruchu, kedy je konkrétne postihnutá schopnosť dieťaťa naučiť sa čítať, aj napriek snahe naučiť sa to od dieťaťa samotného alebo jeho rodičov. Je potrebné zdôrazniť, že bez ohľadu na túto poruchu je inteligencia dieťaťa na normálnej až nadpriemernej úrovni [12].



Obrázok 2.4 - Príklad videnia dyslektika [13] [14].

Pri čítaní sú sledované nasledujúce faktory, ktoré by malo dieťa zvládať pre správnosť porozumenia čítaného textu [4]:

1. **Rýchlosť**. Samotná rýchlosť nie je základnou charakteristikou správneho čítania, ale môže veľmi dobre napovedať, že má dieťa problémy s čítaním textu. Totižto významne súvisí s ďalšími znakmi, ktoré môžu odhaliť problém dieťaťa. Ak dieťa číta príliš pomaly,

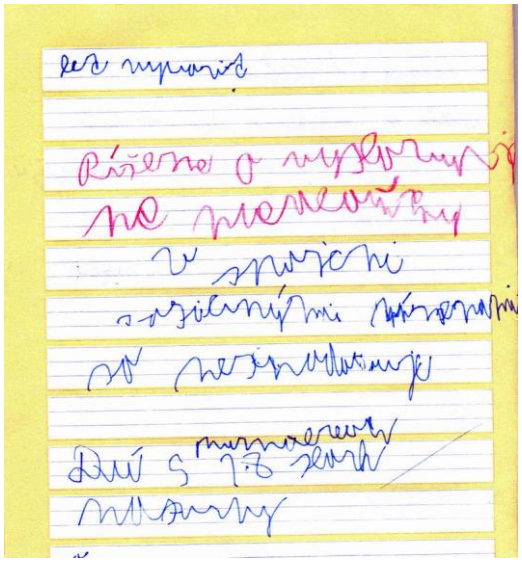
po písmenkách alebo slabikách, tak má evidentne problém s prečítaním a samozrejme rovnako tak s pochopením textu. V tomto prípade ide skôr o snahu dekódovať jednotlivé slová textu, tak aby im porozumelo. Na druhej strane aj prílišná rýchlosť a zbrklosť pri čítaní môže tiež poukazovať na poruchu čítania. Dieťa sa snaží prečítať text čo najrýchlejšie, sústreďuje sa na rýchlosť dekódovania slov a nevníma čítaný text. Obidva prístupy majú za následok to, že dieťa nie je schopné využiť čítanie na získavanie nových informácií, ktoré mu pomáhajú pri výučbe.

2. **Správnosť**. Ide o jeden z najdôležitejších faktorov čítania. Deti trpiace poruchou čítania, často zamieňajú podobne vyzerajúce (b/d/p, u/n, m/n) alebo znejúce (v/f, b/m) písmená, pridávajú si vlastné písmená, môže dôjsť k prešmyčke písmen alebo slabík, domýšľajú si vlastný význam slov, čo má za následok nesprávne prečítanie textu a tým pádom aj jeho nesprávne porozumenie.
3. **Technika čítania**. Samotná technika čítania poukazuje na úroveň dieťaťa v čítaní. Porucha čítania sa prejavuje u dieťaťa aj na jeho technike. Ide zväčša o prekrývanie písmen a slabík a neplynulé čítanie s veľkým množstvom prerušovania. Takéto prejavy sú väčšinou dôsledkom nezvládnutej prvej etapy čítania, kam patrí identifikácia písmen a plynulé čítanie slov a slabík.
4. **Porozumenie čítanému textu**. Ide o asi najpodstatnejšiu časť vedomosti čítania. Samotné čítanie nie je len o prednese daného textu, ale hlavne o jeho porozumení a pochopení. Dieťa by malo byť schopné po prečítaní textu povedať, o čom daný text bol. Pri poruche čítania, je táto schopnosť potlačená. Dieťa nie je schopné porozumieť textu aj napriek bezchybnému čítaniu a intonácii. Je to spôsobené už vyššie spomenutým rýchlym čítaním, kedy sa dieťa sústreďuje len na rýchlosť dekódovania čítaného textu a neuvedomuje si jeho význam.

2.2 Dysgrafia

Dysgrafia je poruchou grafickej stránky písomného prejavu jedinca. Pre dieťa s touto poruchou je učenie písať náročnou úlohou. Matejček píše o dysgrafikovi ako o jedincovi, ktorý sa nenaučí písať, aj keď netrpí žiadnou pohybovou poruchou a nedostatkami inteligencie [15]. Text u takto postihnutého dieťaťa je zväčša kŕčovitý, kostrbatý. Dieťa nesprávne spája písmená, veľa škrtá, zamieňa si tvarovo alebo zvukovo podobné písmená tak ako pri dyslexii, zrkadlovo ich otáča, nedodržuje línie textu – niektoré písmená sú väčšie a iné menšie, sklon písmen je odlišný. Dysgrafia je často sprevádzaná dyslexiou. Je to spôsobné tým, že už pri samotnej dyslexii má

dieťa problém pri rozoznávaní písmen pri čítaní, čo sa následne môže prejaviť aj v písomnom prejave. Najčastejšími dôvodmi môže byť deficit vo vývoji grafomotoriky, koordinácie ruky a oka, impulzivita, kedy dieťa začne písať text ďalšieho slova pričom nedokončilo slovo predošlé [4].



Obrázok 2.5 - Príklad písma dysgrafika [16].

2.3 Dyskalkulia

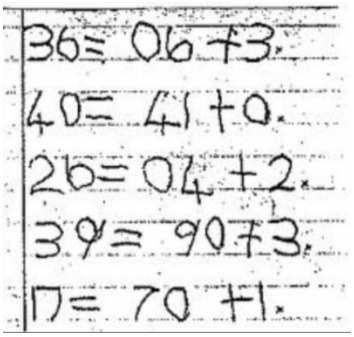
Dyskalkulia je špecifická porucha, ktorej jedinec trpí dysfunkciou matematických schopností. Pri tejto poruche dieťa podáva pri riešení matematických problémov podstatne horšie výkony, než by sa dali predpokladať v porovnaní s jeho výškou inteligencie. Toto tvrdenie vychádza aj z definície, ktorú formuloval Ladislav Košč v roku 1985:

„Vývojová dyskalkulia je štrukturálna porucha matematických schopností, ktorá má svoj pôvod v génovo alebo prenatálnymi vplyvmi podmienenom narušení tých častí mozgu, ktoré sú priamym anatomicko-fyziologickým substrátom veku primeraného dozrievania matematických funkcií, ktoré však zároveň nemajú za následok zníženie všeobecných rozumových schopností.“

To jednoznačne definuje dyskalkuliu ako následok poškodenia tých častí mozgu, ktoré sú zodpovedné za schopnosť riešenia matematických funkcií, či už vplyvom génov alebo prenatálnym vývinom dieťaťa [17].

Nejde však o žiadnu všeobecnú matematickú schopnosť. Vstupuje do nej viacero faktorov, ktoré majú na túto poruchu vplyv. Do počítania sa zapája verbálny faktor (hovorená a písaná reč), priestorový faktor (geometria), usudzovanie (matematická logika), numerický faktor a iné. Dyskalkulia preto vzniká spojením závažností postihnutia jednotlivých faktorov [4].

Dyskalkulia sa nemusí vyskytovať len u detí, ale môže sa vyvinúť aj vo vyššom veku, dôsledkom úrazu hlavy a tým zavineného poškodenia mozgu [4], čo v podstate podporuje aj vyššie spomenutá definícia L. Košču, v ktorej definuje dôvod dyskalkulie ako porušenie určitých častí mozgu.



Obrázok 2.6 - Príklad písania čísel a matematických príkladov dyskalkulika. [18]

Ako už bolo spomenuté vyššie, samotnú dyskalkuliu ovplyvňujú viaceré faktory, ktoré môžu mať rôzne príznaky. Na základe príznakov sa dyskalkulia delí na ďalšie skupiny, ktoré popisujú konkrétnejší efekt dyskalkulie. Nasledujúce typy dyskalkulie sú odstupňované podľa Košča, ktorý ich delí do týchto skupín [19]:

1. **Praktognostická dyskalulia.** Ide o poruchu manipulácie s konkrétnymi predmetmi alebo ich symbolmi. Sem patria číslice, operačné znamienka a podobne. Dieťa má problém s poskladaním predmetov, číslic a znamienok do celkov o danom počte prvkov z čoho vyplýva samotný problém s pochopením prirodzených čísel. Na základe tohto ich nedokáže medzi sebou porovnávať, zoradiť ich podľa veľkosti, v geometrii nedokáže určiť, ktorý z geometrických tvarov je väčší, nedokáže jednotlivé geometrické tvary dostatočne odlišovať, predstaviť si ich rozmiestnené v priestore atď.
2. **Verbálna dyskalkulia.** Ide o poruchu, kedy dieťa nedokáže slovne označiť počet predmetov, vymenovať jednotlivé číslice, operačné znaky a matematické úkony. Dieťa nevie zoradiť čísla vzostupne alebo zostupne, pri vymenovávaní zoradených čísel skáče z čísla na číslo, nedokáže vymenovať násobky čísel, správne si predstaviť a chápať

vyslovené číslo, povedať aký počet predmetov, ktoré sa mu ukazujú, vidí atď. Verbálna dyskalkulia má aj vplyv na pochopenie a rozlišovanie medzi určitými termínmi spojenými s matematikou, ako sú „o 4 viac“ a „4-krát viac“.

3. **Lexická dyskalkulia.** Pri tomto type dyskalkulie má dieťa problém s čítaním matematických symbolov (číslic, čísel, operačných znamienok). Dieťa nie je schopné čítať viacmiestne čísla, zlomky, odmocniny, desatinné čísla, čísla s nulami uprostred, zamieňa si podobne vyzerajúce číslice alebo čísla (6/9, 3/8, 32/23), rímske čísla (IV/VI). Nedokáže prečítať číslo ako celok, ale len ako jednotlivé číslice. Nechápe pozičný význam číslic v čísle (zamieňanie desiatok, jednotiek, stoviek atď.).
4. **Grafická dyskalkulia známa tiež ako Numerická dyskalkulia.** Pri tomto type dyskalkulie má dieťa problém s písomným prejavom v matematike. Dieťa má problém so zápisom čísla, nedokáže správne zapísať diktované číslo (píše ho slovne), nevie písať čísla v primeranej a rovnakej veľkosti, pri násobení alebo sčítavaní pod sebou nedokáže zapísať čísla pod seba tak, aby boli jednotlivé číselné sústavy správne v stĺpcovom zákryte. V geometrii sa prejavuje táto porucha tak, že dieťa nedokáže narysovať jednoduché geometrické tvary a obrazce.
5. **Operačná dyskalkulia.** Pri tomto type dyskalkulie je narušená schopnosť riešenia matematických operácií. Dieťa si často zamieňa operácie, ako napríklad sčítanie za násobenie a odčítanie za delenie. Pri zložitejších matematických operáciách, ako napríklad pri sčítavaní dlhších číselných rád si dieťa zamieňa desiatky za jednotky, pri delení si zamieňa čitateľa s menovateľom. Vo vyššom veku, kedy by malo mať dieťa zvládnuté a zafixované bežné matematické operácie, je znakom operačnej dyskalkulie aj počítanie na prstoch, s čím je spojená zvýšená chybovosť so sčítavaním a odčítavaním do 20, pri násobení, delení a riešení kombinovaných matematických úloh.
6. **Ideognostická dyskalkulia.** Podľa Košča ide o posledný typ dyskalkulie. Pri tomto type poruchy dieťa nevie pochopiť matematické pojmy a vzťahy medzi nimi. Ako príklad je možné uviesť to, keď dieťa vie, že číslo 9 sa číta ako „deväť“ a vie, že keď prečíta slovo „deväť“, tak ide o číslicu 9 ale nevie pochopiť to, že 9 je o jedno menej ako 10, alebo to, že 9 je rovné 3x3 a iné súvislosti, ktoré s daným číslo súvisia. Pre dieťa s touto poruchou, je napríklad problémom vyriešiť matematickú úlohu, ak sa pozmení jej šablónovitý postup. Medzi najťažšie poruchy patrí to, keď dieťa nedokáže pokračovať vo vymenovávaní číslic po jednom od ľubovoľne zadaného čísla. Ideognostická dyskalkulia patrí medzi najrozšírenejšie poruchy postihujúce matematické vedomosti.

3 Súčasný stav testovania SPU

Trh v dnešnej dobe ponúka rôzne psychologické testy, úlohy na precvičovanie špecifických porúch učenia a ďalšie, ktoré sú určené na tzv. „domáce cvičenie“ alebo vzdialené testovanie, alebo iba ako jednoduché precvičovanie pri daných poruchách. Lenže tieto programy sú mnohokrát len akýmsi „jednoduchým balíčkom“, ktorý neponúka možnosti ako sú

- Odosielanie dát po dokončení úlohy na pracovisko lekára, dozoru
- On-line spracovanie predbežných výsledkov, testov
- Prispôsobenie na stav riešiteľa
- Konfigurovateľnosť testov
- Jednotné rozhranie pre tvorbu a ovládanie rôznych typov úloh

Väčšina týchto hier je orientovaná len na desktop, bez možnosti použitia na iných zariadenia, ako napríklad tablety, hlavne z dôvodu ich ovládania pomocou klávesnice (stisk klávesy ako odpoveď). Čo neumožňuje prenášanie, aplikácie alebo ovládanie špecifických typov úloh iba na dotykovej obrazovke. Ďalšou nevýhodou takýchto aplikácií je nejednotnosť rozhrania, teda vzhľadu pre užívateľov. Väčšina testov je dodávaná ako celkom samostatné úlohy (moduly). Ak chce užívateľ používať rôzne úlohy, musí si ich samostatne nainštalovať na svoj počítač a spúšťať ich jednotlivo. Ďalšou veľkou nevýhodou je neschopnosť odosielania dát na spracovanie na určené pracovisko odbornej osoby, čo môže mať určité následky na kontroly priebežného stavu rehabilitačného procesu.

Webové rozhranie taktiež ponúka rozličné aplikácie, ktoré sú vhodné na testovanie takýchto porúch. Avšak aj tieto verzie trpia určitými nedostatkami. Väčšinou sa jedná o úlohy typu „hra“, čo znamená, že užívateľ po dokončení daného testu alebo úlohy, dostane nejaký druh lokálneho vyhodnotenia bez možnosti odoslania dát na spracovanie odbornou osobou. Ako to bolo u desktopových aplikácií, tak aj u webových chýba jednotné rozhranie, ktoré by zastrešovalo úlohy s rovnakým, alebo podobným zameraním a umožňovalo by jednoduchý spôsob ich tvorenia a ovládania.

Ďalšou nevýhodou súčasných testov je, že ich rozhranie neposkytuje možnosť konfigurácie a adaptativity, alebo len vo veľmi obmedzenom rozsahu. Ide o dosť veľký nedostatok, ktorý robí z týchto úloh jednoliate riešenia bez možnosti vytvorenia nového zadania úlohy, alebo prispôsobenia úlohy v rámci obtiažnosti.

Posledným typom testov, ktoré je možné v dnešnej dobe dostať, sú brožúrované / papierové vydania. Hlavnou úlohou týchto testov je precvičovanie daných špecifických porúch učenia. Sú určené hlavne na domácu rehabilitáciu alebo akýchsi foriem precvičovania už spomenutých špecifických porúch. Aj v tomto prípade je najväčší problém jednotvárnosť úloh (stále sa opakujúce zadanie), nemožnosť konfigurovateľnosti a adaptativity. Ale najväčším problémom je nemožnosť okamžitého odosielania dát na spracovanie odbornou osobou, keďže ide o papierové verzie. Výsledky z týchto testov bývajú málokedy alebo nebývajú vôbec spracované odbornou osobou, pretože proces doručenia výsledkov lekárovi väčšinou chýba.

4 Analýza implementovanej aplikácie

Nasledujúca časť práce sa bude venovať analýze implementovaných úloh. Táto analýza v sebe zahŕňa predstavenie reprezentačnej skupiny úloh pre testovanie alebo rehabilitáciu špecifických porúch učenia. Krátky popis princípu úloh, na akú poruchu sú vhodné a aké ponúkajú možnosti konfigurácie. Ďalej sa bude táto časť venovať analýze vybranej implementačnej platformy – prečo bola vybraná, aké ponúka výhody oproti ostatným atď. Na základe vybranej platformy sa bude venovať aj popisu a funkčnosti tohto frameworku. Čo daný framework ponúka, aké typy úloh sa v ňom dajú vytvárať, ako funguje zber a spracovanie dát a popis analýzy rozšírenia pre zobrazenie nameraných a spracovaných dát, ktoré bolo vytvorené ako súčasť tejto práce.

4.1 Výber vzorových úloh pre implementáciu (v tejto práci)

Pred samotným návrhom a implementáciou úloh bolo potrebné vyhľadať a vytvoriť zoznam úloh, ktoré by boli vhodné na testovanie alebo precvičovanie dyslexie, dysgrafie a dyskalkulie. Prehľad týchto úloh je možné vidieť v prílohe B. Pre návrhovú a implementačnú časť práce bolo vybraných, ako reprezentatívna vzorka, 5 typov úloh, pričom dve z nich sú pozmenené tak, že každá vytvára ďalšie samostatné úlohy. Jednotlivé úlohy boli rozdelené do troch špecifických skupín, ktoré reprezentujú ich zameranie:

- **Zrakové vnímanie** – úlohy určené na vnímanie rôznych geometricky alebo graficky odlišných objektov
- **Číselné vnímanie** – úlohy určené na rozoznávanie tvaru čísel a ich súvislostí
- **Textové vnímanie** – úlohy určené na vnímanie textu, rozoznávanie jednotlivých písmen, slabík a slov.

Do týchto skupín boli rozdelené z dôvodu ich vhodnosti nie len na jednu špecifickú poruchu, ale hneď na niekoľko.

Prvým príkladom je úloha s názvom **„Rozdiely“**. Zameriava sa na precvičovanie zrakového vnímania. Zrakové vnímanie je dôležitou súčasťou vývoja dieťaťa a prispieva k správnemu porozumeniu sveta okolo neho, tým pádom aj samotného čítaného alebo písaného textu. Z tohto dôvodu je táto úloha vhodnou formou pre diagnostiku alebo rehabilitáciu špecifických porúch učenia. Pri dyslexii a dysgrafii je potrebné, aby dieťa dôkladne vnímalo čítaný respektíve

písaný text svojím zrakom, aby dokázalo vnímať jednotlivé slová správne, v spletitosti celého textu. Preto táto úloha ponúka precvičovanie zrakového vnímania, ktoré je dôležité aj pri samotnom vnímaní písmen a ich tvarov, slabík a slov pri čítaní alebo písanom prejave dieťaťa. Úlohou užívateľa je nájsť na dvoch identických obrázkoch predmety, v ktorých sa tieto dva obrázky odlišujú. Takýmto spôsobom je dieťa nútené všímať si odlišnosti a tak precvičovať svoje vnímanie v širšom kontexte.

Druhá úloha s názvom **„Rôzne Obrázky“** sa taktiež zameriava na precvičovanie zrakového vnímania. Ako už bolo spomenuté vyššie, zrakové vnímanie je dôležitou súčasťou správneho porozumenia okolitého sveta a neobchádza to ani čítaný a písomný prejav dieťaťa. Úlohou užívateľa je nájsť spomedzi obrázkov umiestnených v riadku za sebou tie, ktoré sa odlišujú od väčšiny obrázkov v tomto riadku. Nemusia byť rozmiestnené len v riadku ale aj v matici. Preto je táto úloha vhodná na precvičovanie alebo diagnostiku všetkých troch skúmaných špecifických porúch učenia. Pri dyslexii a dysgrafii má veľkú výhodu, ako to bolo aj v predošlej úlohe, vnímanie rozdielov medzi obrázkami a nájdenie toho požadovaného. Rozlišovanie v tvaroch a natočení, ktoré je podstatnou zložkou aj pri rozlišovaní písmen, slabík a slov. Pri dyskalkulii pomáha hlavne pri precvičovaní geometrických obrazcov, keďže obrázky v tejto úlohe sú tvarovo rozličné a je potrebné vždy nájsť medzi nimi rozdiely.

Treťou úlohou je **„Spájanie“**. Ako sa dá vydedukovať zo samotného názvu ide o spájanie častí objektu tak, aby takto užívateľ vytvoril celý objekt. Presnejšie povedané, objekt je rozrezaný na menšie časti a užívateľovou úlohou je poskladanie celého objektu spájaním týchto častí. Zameranie tejto úlohy je opäť zrakové vnímanie. Pri tejto úlohe je oproti predošlým úlohám kladený dôraz hlavne na tvar objektu a jeho častí. Užívateľ musí dôkladne sledovať rezy jednotlivých častí, aby ich dokázal na seba naviazať. Z tohto dôvodu by sa dalo povedať, že primárne je úloha určená pre diagnózu alebo rehabilitáciu dyskalkulie, konkrétne grafickej dyskalkulie, pod ktorú patria aj poruchy rysovania alebo rozoznávania geometrických objektov. Samozrejme tak ako to bolo aj u predošlých úloh, aj táto úloha sa dá použiť na precvičovanie dyslexie a dysgrafie, keďže rozoznávanie tvarov napomáha pri precvičovaní zrakového vnímania písmen a ich tvarov, slabík a slov.

Ďalšou úlohou je **„Čísla“**. Táto úloha je rozdelená na dve verzie, ktoré vytvárajú dve rozdielne úlohy. V prvej verzii úlohy, musí užívateľ na základe zobrazeného čísla na pravej strane obrazovky a počtu zobrazených bodiek na ľavej strane obrazovky, odpovedať kliknutím na

príslušné tlačítko („Áno“, „Nie“), či je počet bodiek rovný zobrazenému číslu. Pri druhej verzii užívateľ neodpovedá pomocou kliknutia na tlačítko „Áno“ respektíve „Nie“, ale vyberá z troch možností čísel to, ktoré sa rovná počtu zobrazených bodiek. U predošlých úloh išlo o zrakové vnímanie, čo je vo všeobecnosti použiteľné pre rôzne typy špecifických porúch učenia. Pri tejto úlohe je zameranie konkrétnejšie nasmerované na dyskalkuliu. Z popisu úlohy je možne vidieť, že hlavnou myšlienkou je diagnostika alebo precvičovanie prepojenia zrakového vnímania čísla ako tvaru a toho čo predstavuje, určiť počet niečoho, v tomto prípade bodiek.

Poslednou vybranou úlohou je **„Správne slovo“.** Patrí do skupiny textového vnímania. Ide o úlohu, v ktorej užívateľ vyberá správne slovo z dostupného zoznamu. Ten je tvorený v podstate iba jedným slovom, ktoré sa odlišuje rôznym prevedením (zámena písmen u/n, b/d/p/q atď.) pričom len jedno z týchto zobrazených slov je napísané správne. Aj v tomto prípade je úloha rozdelená na dve verzie. Prvá verzia je presne tá, ktorá bola popísaná pred chvíľkou, pričom druhá verzia sa líši v tom, ako sa tvoria nevyhovujúce slová. Tu je z týchto slov odstránené nejaké písmenko alebo slabika, pričom sa testuje či si užívateľ vybrané slovo dôkladne prečítal, alebo ho len rýchlo preletel. Pri oboch verziách je zameranie úlohy nasmerované na diagnostiku alebo rehabilitáciu dyslexie a dysgrafie, keďže si užívateľ musí všímať rozdiely v správnom a chybných slovách. Tie sú charakteru chýb, ktorých sa dopúšťajú dyslektici a dysgrafici (zámena písmen, vynechávanie písmen atď.).

4.2 Použitý Framework

Ako bolo popísané v cieľoch práce, zameriava sa na implementáciu úloh, ktoré majú slúžiť na diagnostiku respektíve rehabilitáciu troch špecifických porúch učenia a to dyslexiu, dysgrafiu a dyskalkuliu. Aby boli tieto úlohy jednoducho a hlavne rovnako ovládateľné a rýchlo prístupné, bolo potrebné tieto úlohy zoskupiť pod jednotné aplikačné prostredie, ktoré by umožňovalo rozdeliť jednotlivé úlohy do rôznych kategórií a zabezpečilo by jednoduchý a rýchly prístup k týmto úlohám. Vzhľadom k tomu, že v roku 2011 bol vytvorený framework, ktorý slúži na tvorbu takýchto typov úloh, bol tento framework využitý aj pri tvorbe úloh v tejto práci. Framework bol pôvodne vytvorený vedúcim tejto práce, pre úlohy na liečbu rôznych foriem očných chýb (strabizmu). V dnešnej dobe má širokú použiteľnosť od domácej rehabilitácie spomínaného strabizmu, až po cvičenia určené pre ľudí s poruchami pohybového aparátu. Preto by som sa v nasledujúcej časti práce rád stručne zmienil o tomto frameworku a priblížil jeho funkciu pri tvorbe týchto úloh.

4.2.1 Organizácia (správa) systému

Pod týmto názvom kapitoly rozumieme to, ako framework identifikuje svojich užívateľov, ako zamedzuje úniku osobných údajov atď.
Identifikácia užívateľa je pri tvorbe úloh, ktoré používa väčšie množstvo užívateľov, podstatnou požiadavkou. Je potrebné zabezpečiť to, aby sa akékoľvek osobné údaje užívateľa alebo jeho dáta nezverejnili. Okrem ochrany osobných údajov je ďalšou požiadavkou jednoznačná identifikácia pacienta (užívateľa), jeho lekára a pracoviska, na ktorom lekár ordinuje. Pri prvom bode by bolo najjednoduchším riešením použitie tzv. „hash“ hodnoty zo zadaného vstupu, napríklad z mena a rodného čísla pacienta. Takéto riešenie však nevyhovuje druhej podmienke, ktorou nie je len identifikácia pacienta, ale aj rovnako jeho lekára alebo dokonca jeho pracoviska. Z tohto dôvodu si použitý framework identifikuje svojich užívateľov pomocou unikátneho číselného kódu, ktorý sa skladá z troch častí:

- Miesto pôsobenia lekára / kontrolnej osoby (geografické miesto – dve číslice)
- Identifikácia lekára / kontrolnej osoby na tomto mieste (pracovisko - dve číslice)
- Poradové číslo pacienta u tohto lekára (šesť číslic)

Týmto číslom sa dá jednoznačne identifikovať konkrétny pacient, jeho lekár a pracovisko lekára. Framework tento identifikačný kód nedovoľuje vložiť bez vstupného hesla, ktoré má každý užívateľ. Takto je zabezpečené neautorizované vloženie tohto unikátneho čísla cudzou osobou. Týmito dvoma údajmi sú dáta identifikované po dobu prenosu a spracovania. Až po vyhodnotení môžu byť dáta vložené do databázy pod menom pacienta. Tieto unikátne čísla sa dajú dekódovať pomocou prevodnej tabuľky.

4.2.2 Technické riešenie – výber použitej technológie

Pod pojmom technického riešenia sa rozumejú použité informačné technológie (použitý programovací jazyk, webové technológie, operačný systém,...atď.).
Z pohľadu technického riešenia tejto práce, pri ktorom ide vždy o kompromis medzi dostupnými technológiami a ich zložitosťou, ponúka použitý framework riešenie na rozsiahlu základňu požiadavkou na výber technologickej platformy:

- Možnosť tvorby, v podstate ľubovoľne interaktívnych úloh s takmer ľubovoľnou

multimediálnou podporou

- Využitie vyššieho programovacieho jazyka pre tvorbu komplexných aplikácií, namiesto použitia rôznych skriptovacích jazykov s obmedzenými schopnosťami
- Využívať technológie s veľkou prenositeľnosťou kódu dostupné na tvorbu nie len desktopových aplikácií, ale aj webových respektíve tabletových aplikácií
- Umožniť včlenenie rôznych výpočtových algoritmov priamo do úlohy užívateľa. Ako sú napríklad algoritmy na adaptatívnosť úlohy, priebežné vyhodnotenie úlohy...atď.
- Pri webových aplikáciách celkové ovládanie úlohy presunúť na klientsku stranu a spojenie so serverom obmedziť len na spúšťanie úlohy a následné odosielanie a spracovanie dát od užívateľa
- Možnosť použitia ľubovoľných externých zariadení od Webových kamier až po špecializované ovládače komunikujúce prevažne pomocou USB portu

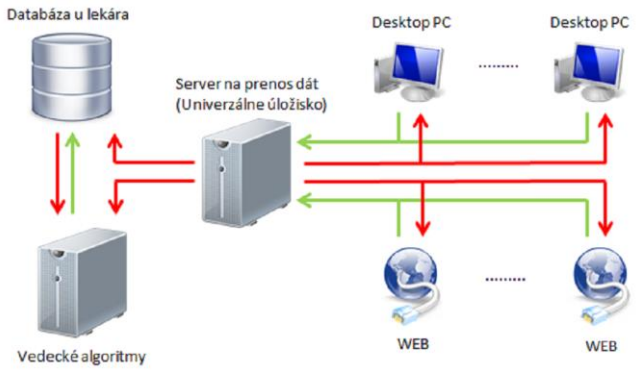
Ak zoberieme do úvahy požiadavku na webové aplikácie, tak pre riešenie takéhoto projektu v podstate vypadáva použitie akejkoľvek statickej webovej stránky (HTML), alebo serverom generovanej stránky (PHP, ASP.NET). Pre možnosť použitia rôznych externých zariadení pripojiteľných pomocou USB portu, alebo iných, je tiež nevhodná možnosť použitia niektorých zo skriptovacích jazykov ako JavaScript alebo Flash. Ďalšou požiadavkou je, aby použitá technológia bola schopná začleniť niektoré zo špeciálnych výpočtov (funkcionálne jazyky, náročné matematické algoritmy) priamo do aplikácie a zároveň aby pokryla možnosť vytvorenia skoro jednotnej aplikácie, ktorá by bola schopná bežať ako dekstopová, webová alebo dotyková aplikácia. Z toho vyplýva, že jediným a ľahko dostupným platformovým riešením, ktoré vyhovuje všetkým požiadavkám, je Microsoft .NET Framework a jeho odnože, ako sú MONO (Linux) a Xamarin (Android). Keďže je vybraný framework vytvorený pomocou technológie .NET, poskytuje tak riešenie pre všetky vyššie spomenuté požiadavky a tým pádom je vhodným riešením pre tvorbu a implementáciu aplikácie v rámci tejto práce. Samozrejme technológia .NET má aj svoje negatívum a to je obmedzená platformová univerzálnosť. Platforma .NET je primárne viazaná na operačný systém od Microsoftu, čo ju (jednoducho) neumožňuje prenášať napríklad na operačný systém založený na Linuxe. Avšak pomocou platformových rozšírení, ako je MONO alebo Xamarin, je do istej miery možné vytvárať aplikácie aj na OS Linux alebo Android, čo vytvára z .NET vo všeobecnosti vysoko hodnotné riešenie pre tvorbu aplikácie v rámci tejto práce.

Z dôvodu použitia vybraného frameworku a teda aj platformy .NET a jeho súčastí (WPF, Silverlight) je možné vytvárať aplikácie nie len ako desktopové, ale aj webové a vďaka rozvoju a rozširovaniu tabletov používajúcich OS Windows 8 alebo RT aj ako tabletové, bez nutnosti veľkých zmien kódu vytváranej aplikácie. Táto výhoda je spôsobená tým, že .NET používa formu programovania, kedy je možné veľmi vhodne separovať kód GUI od vývoja samotnej logiky aplikácie. Ďalšou výhodou použitia tejto technológie je možnosť priamej integrácie funkčných jazykov ako sú F#.NET a Prolog.NET. .NET navyše umožňuje použitie viacerých vývojových jazykov ako sú C# alebo VisualBasic. V prípade implementácie aplikácie v rámci tejto práce bol vybraný vývojový jazyk C# z dôvodu, že samotný framework je pomocou neho vytvorený a z dôvodu mojej znalosti tohto jazyka.

4.2.3 Oblasť využiteľnosti

Vybraný framework využíva ako platformovú technológiu .NET, ktorá disponuje rozsiahlou škálou podporných knižníc, od základných určených na prácu s premennými, zoznamami atď. až po sofistikované grafické knižnice pre 2D ale aj 3D objekty. Z tohto dôvodu nie je oblasť použiteľnosti limitovaná vybranou technológiou tak, ako by to mohlo byť napríklad pri použití iných programovacích jazykov (Java, PHP, C, C++ atď.). V tomto prípade ide hlavne o snahu plne využiť schopnosti technológie Microsoft .NET. Preto je možné povedať, že oblasť využiteľnosti frameworku je relatívne neobmedzená.

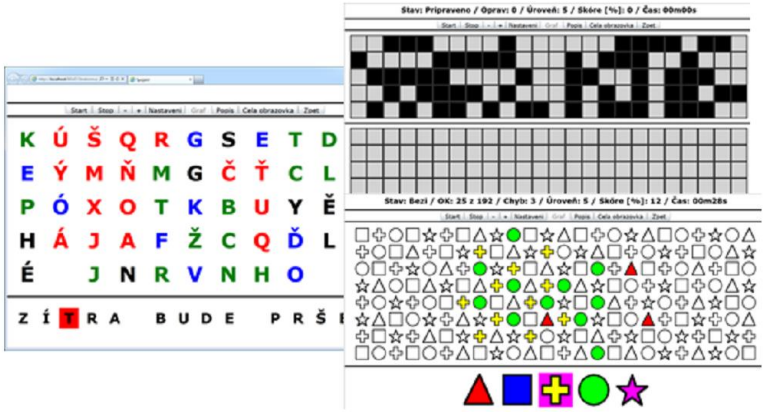
Využiteľnosť frameworku z pohľadu úloh, ktoré sú v ňom vytvárané, je možné rozdeliť na dve časti, na samotnú tvorbu úlohy a na prenos dát z úlohy. Zber nameraných dát a prenos dát na server je jednou zo základných požiadavkou, ktoré vybraný framework spĺňa. Dáta sú pomocou frameworku posielané na server pomocou rôznych prenosových kanálov, kde sú neskôr spracované. Z tohto dôvodu sú prenášané dáta uložené vo forme XML. Ide o jazyk, ktorý je určený hlavne na serializáciu dát a ich prenos medzi aplikáciami. Jeho úlohou je popis vecného obsahu dát a nezaoberá sa sám o sebe vzhľadom dokumentu alebo jeho časťou. Následne je možné tieto dáta vyzdvihnúť zo servera inou aplikáciou, ktorá môže mať na starosti spracovávanie dát. Dáta je po spracovaní možné znova odoslať na počítač dozornej osoby (lekára), ktorý si ich môže vhodnou formou zobraziť (grafy, tabuľky atď.). Z dôvodu ochrany osobných údajov framework neodosiela na server žiadne osobné údaje o užívateľovi (pacientovi), tie ostávajú uložené len v počítači lekára.



Obrázok 4.1 - Princíp spolupráce jednotlivých častí systému a cesty so smermi prenosu dát [20].

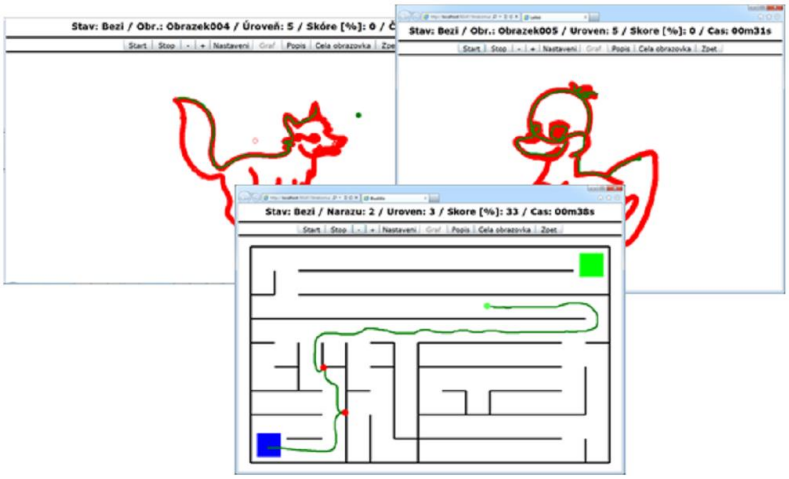
Pri samotnej tvorbe úloh, poskytuje vybraný framework podporu pre tvorbu rozličných úloh a testov, od jednoduchších doplňovacích úloh, až po zložitejšie, ktoré využívajú rôzne animácie (2D, 3D), pohyb, transformácie atď. Pomocou neho už bolo vytvorených značné množstvo úloh, ktorých obrázky je možné vidieť v nasledujúcej časti, ako príklad rozmanitosti úloh vytvorených pomocou tohto frameworku. Je ich možné podľa ich funkcionality rozdeliť na 3 hlavné skupiny:

1. **Doplňovacie úlohy** (predstavivosť, stratégia, logické myslenie, pamäť rozoznávanie tvarov, stratégia) – Ide o úlohy, v ktorých je primárnym cieľom užívateľa niečo doplniť, či už ide o písaný text, písmená alebo políčka mozaiky.



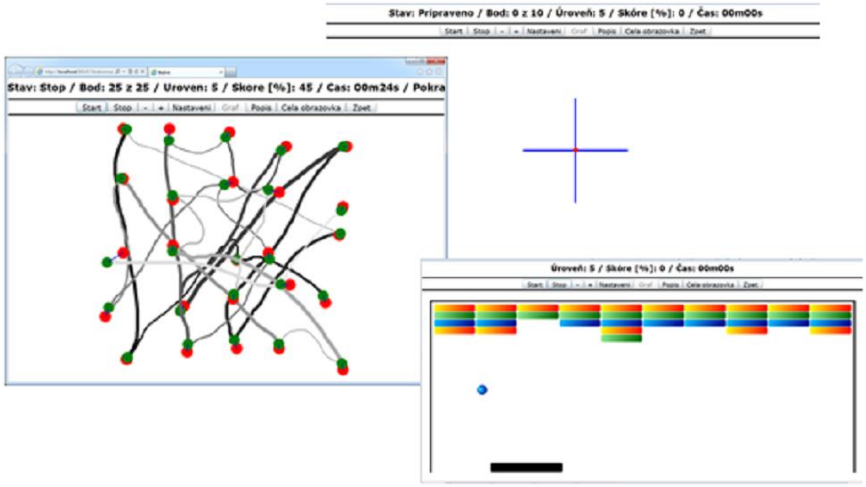
Obrázok 4.2 - Príklady úloh založených na maticovom hracom poli (hľadanie písmen, vypĺňovanie podľa vzoru, doplňovanie obrázkov) [20].

2. **Úlohy na cvičenie jemnej motoriky** (logické myslenie, koncentrácia, vizuálne schopnosti, rozoznávanie tvarov atď.) – Pri tomto type úloh je ich hlavnou charakteristikou bohatšia vnútorná logika.



Obrázok 4.3 - Príklady úloh na jemnú motoriku (obkresľovanie, bludisko) [20].

3. **Úlohy na hodnotenie postrehu** (reakcie, pozornosť, koncentrácia, sluch, farebný vnem) – Hlavným zameraním sú úlohy s pohyblivými časťami.



Obrázok 4.4 - Príklady postrehových testov (reakcie na zobrazené body, streľba bodov na kríž, pinball) [20].

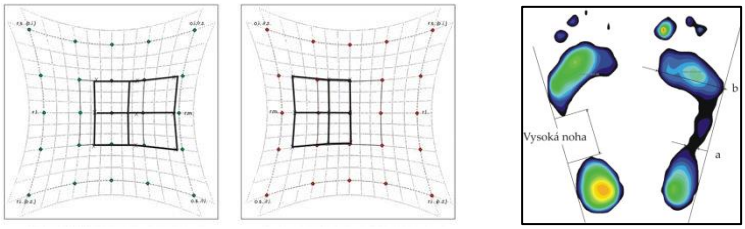
4.2.4 Analýza rozšírenia frameworku

Okrem samotnej implementácie jednotlivých úloh na diagnostiku alebo rehabilitáciu špecifických porúch učenia, bolo jedným z cieľov tejto práce, rozšírenie použitého frameworku o možnosť zobrazenia spracovaných dát, ktoré sa namerajú vždy pri riešení jednotlivých úloh užívateľom. Ide o dôležitú súčasť, ktorú samotný framework zatiaľ neobsahoval.

Pre typ aplikácie, ktorej hlavným cieľom je zastrešenie úloh určených na diagnostiku alebo rehabilitáciu, je jednou z hlavných požiadaviek, ako to už bolo spomenuté, zber dát z priebehu riešenia úlohy. Tie sú potrebné pri zistení stavu pacienta v súvislosti so zameraním danej úlohy na jednu alebo skupinu porúch, pre ktorú je daná úloha určená. Ako už bolo spomenuté, tieto dáta sa následne odosielajú na server, kde sú podrobené špecializovanému spracovaniu. Po takomto spracovaní by malo byť možné dané dáta vhodne zobraziť. Pre lekára, ako dozornej osoby, je veľmi dôležité, aby si mohol namerané a spracované dáta z týchto úloh prehliadnuť a poprípade porovnať so staršími výsledkami. Preto bolo potrebné rozšíriť využitý framework o zobrazovacie aplikácie, ktoré by dané dáta vhodne prezentovali. Počas dlhoročnej praxe, ktorú má vedúci tejto práce z tvorby obdobných úloh a spolupráce s lekármi v diagnostike a rehabilitácií rôznych porúch zrakového a pohybového ústrojenstva, bol za jeho pomoci vytvorený návrh pre zobrazovanie nameraných a spracovaných dát tak, aby spĺňal základné požiadavky:

- Jednoduchosť – iba dôležité údaje a nie preplnenosť údajmi
- Zrozumiteľnosť – vhodné pozičné a farebné odlíšenie
- Prehľadnosť – dostatočne veľké texty, obrázky a informácie

Obzvlášť v medicíne sú tieto požiadavky potrebné. Pri diagnostike porúch rôznych funkcií, ako napríklad motoriky, vnímania alebo zraku, je potrebné jednoducho, zrozumiteľne a prehľadne zobraziť namerané dáta z diagnostických testov. Je to z toho dôvodu, že samotní lekári potrebujú okamžite z nameraných výsledkov vidieť problém, na ktorý dáta poukazujú. Pri použití bežných číselných metód je zrozumiteľnosť a prehľadnosť potlačená do úzadia, pretože číselné údaje nie sú nijak prehľadné. Preto je potrebné, aby forma zobrazenia dát spĺňala vyššie spomenuté požiadavky. Ako príklad takéhoto zobrazenia môže slúžiť výstup z diagnostickej metódy tzv. Hassove plátno (Hass Screen), alebo výstup z tlakovej podložky (Obrázok 4.5).



Obrázok 4.5 - Príklady zobrazenia nameraných dát z testovania. V ľavo Hassove plátno, v pravo tlaková podložka [21] [22].

Je možné vidieť, že zobrazenia na oboch obrázkoch neobsahujú žiadne čísla, ale ide len o grafické a hlavne prehľadné zobrazenie meraných dát pri diagnostike problémov pacienta. Ide o veľmi efektívne, prehľadné a zrozumiteľné zobrazenie nameraných dát. Skúsený lekár iba pohľadom na takýto typ vhodného zobrazenia získa veľké množstvo informácií, s ktorými ďalej dokáže veľmi efektívne pracovať. Týmto sa však táto práca nezaoberá, obrázky sú uvedené len ako ilustračné príklady zobrazenia nameraných a spracovaných dát v obrazovej forme, pričom spĺňajú vyššie spomenuté základné požiadavky. Je potrebné ešte poznamenať, že v medicíne sa používajú aj presné číselné hodnoty v prípade potreby exaktných chirurgických zákrokov. Avšak pri samotnej diagnostike problému v širšom kontexte, je postačujúce a vítané, aby dané formy zobrazenia boli čo najjednoduchšie a zároveň presne poukazovali na konkrétny problém pacienta.

Ako rozšírenie frameworku, boli do tejto práce vybrané 3 formy zobrazenia dát a to graf (pre formu číselného zobrazenia), tabuľka (pre formu textového hodnotenia) a kalendár (pre prehľadové informácie).

Prvou formou zobrazenia dát je stĺpcový **graf**. Ponúka vhodné spojenie vizuálnej stránky s hodnotami, ktoré reprezentujú konkrétne dáta, avšak bolo potrebné dodržať vyššie spomenuté požiadavky. Z tohto dôvodu nebolo možné použiť bežné čiarové grafy. Tie sú v tomto prípade neprehľadné, pretože väčšina dát zobrazovaných touto formou je zameraná na prezentáciu konkrétnej hodnoty za konkrétny deň, mesiac, rok alebo variantu úlohy. Čiarový graf by zbytočne prepájal jednotlivé vrcholky zobrazených dát, čo by bolo príliš mätúce. Na tento účel sa nehodí ani koláčový graf. Jeho nevýhodou je nemožnosť porovnania vzrastu alebo poklesu hodnôt za dlhší časový úsek. Preto najvhodnejšie riešenie poskytujú stĺpcové grafy, ktoré zobrazujú relatívnu hodnotu (percentá) pre konkrétnu meranú veličinu (za deň, mesiac, rok, podľa varianty úlohy atď.). Tie umožňujú aj samotné porovnanie už spomínaného vzrastu a poklesu. Navyše je možné farebným gradientom veľmi vhodne zvýrazniť percentuálnu hodnotu, ktorú príslušný stĺpec reprezentuje. Tento spôsob zobrazenia rešpektuje aj vyššie spomenuté požiadavky. Pri tomto type grafu nie je ani potrebné čítať zobrazené hodnoty. Je postačujúce zamerať sa buď len na posudzovanie veľkosti (výšky) jednotlivých stĺpcov, alebo len letmo pozorovať celkové zafarbenie grafu. Preto je vhodné pre porovnávanie dlhodobo trvajúcich záznamov.

Ako ďalší typ zobrazenia bola vybraná jednoduchá **tabuľka**. Ide o formu, ktorá jednoznačne pokrýva požiadavky jednoduchosti, zrozumiteľnosti a prehľadnosti. V mnohých prípadoch obsahujú namerané dáta údaje, ktoré je zložité zobraziť pomocou grafov, alebo dokonca

pomocou čísel. Ide o rôzne úlohy, v ktorých užívateľ vyberá nejakú správnu odpoveď spomedzi väčšieho množstva nesprávnych. Záujmovou položkou výstupu z takejto úlohy je hlavne užívateľom vybraná odpoveď a skutočne správna odpoveď pre danú variantu úlohy. Pri hodnotení ide väčšinou o porovnanie vybranej odpovede so správnou, preto je grafový výstup nevhodným zobrazením takéhoto porovnania. Najlepšia možnosť je už spomínaná tabuľka, v ktorej je možné zobraziť jednotlivé slová v riadku, ako jeden záznam, a tak ich jednoducho a hlavne prehľadne prezentovať. Lekár následne ich porovnaním môže vidieť presné slovo / slová, ktoré pacientovi (užívateľovi) robili problém.

Poslednou formou zobrazenia informácií, ktoré boli vybrané ako rozšírenie frameworku, je **kalendár**. Kalendár je aj v bežnom živote používaný hlavne na to, aby mali ľudia prehľad o určitej činnosti za nejaké časové obdobie. Zaznamenávajú sa doňho údaje, na základe ktorých je možné povedať napríklad koľko krát do mesiaca sme robili určitú činnosť. Využitie tejto metódy záznamu je veľmi praktické aj pri zobrazovaní nameraných dát a to hlavne z úloh určených na domácu rehabilitáciu. Tu lekár často potrebuje vidieť intenzitu riešenia úloh v domácom prostredí. Preto bola zvolená zobrazovacia forma dát založená na báze kalendára, pomocou ktorej môže lekár vidieť dni, v ktoré pacient úlohy riešil a v ktoré nie. Navyše pomocou využitia farby je možné zvýrazniť intenzitu riešenia úloh v daný deň. Je možné veľmi ľahko rozlíšiť, či užívateľ napríklad venoval riešeniu úloh 10 min alebo 1 hodinu. Pri každom dni je možné zadať stručný text, ktorý môže slúžiť ako určitý typ krátkej poznámky k riešeniu úlohy v daný deň. Jedná sa teda o veľmi vhodné prehľadové zobrazenie, ako pacient pristupuje k svojím cvičeniam. I v tomto prípade čísla skôr prekážajú a pomocou vhodného farebného podania je možné dosiahnuť omnoho väčšej vypovedajúcej hodnoty.

5 Návrh implementovanej aplikácie

Pri implementácii takéhoto špecializovaného softvéru je neoddeliteľnou súčasťou jeho vhodný návrh. Pri návrhu je potrebné definovať štruktúru aplikácie, z akých častí sa bude skladať, ako bude vyzerať jej grafické rozhranie, následne je potrebné vytvoriť návrh samotných častí a navrhnúť ich funkčnosť, ovládanie a zobrazovanie dát. V nasledujúcej kapitole bude preto popísaný návrh implementovanej aplikácie. Od vzhľadu celkovej aplikácie, cez jednotlivé úlohy ich ovládanie, zberu meraných dát, až po jednotlivé rozšírenia použitého frameworku, určené pre zobrazovanie dát. Pri návrhu jednotlivých častí boli brané v úvahu možnosti použitého frameworku, jeho knižníc a grafického rozhrania.

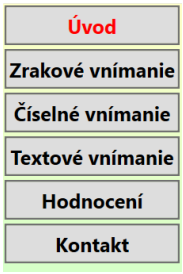
Vo zvyšnej časti tejto práce budú používané nasledujúce pojmy:

- **Informačná lišta** – panel pre zobrazovanie stavu a informáciách o úlohe.
- **Ovládacia lišta** – panel s ovládacími tlačítkami úlohy respektíve aplikácie
- **Hracia plocha** – základná plocha úlohy
- **Hracie plátno** – plátno, kde sa vykresľujú objekty úlohy
- **Obyčajné plátno (Canvas)** – plátno slúžiace na presné kreslenie po pixloch
- **Maticovo založené plátno (Grid)** – herný podklad, ktorý je možné rozdeliť na jednotlivé bunky predstavujúce maticu

5.1 Hlavné menu aplikácie

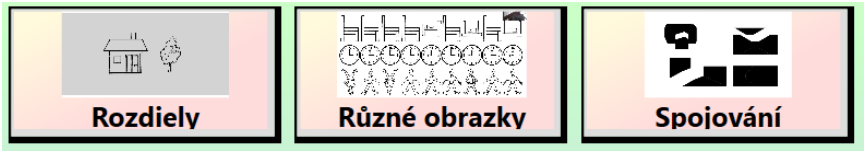
Ako už bolo povedané na začiatku tejto práce, jedným z hlavných dôvodov prečo je potrebné vytvoriť takúto aplikáciu je ten, že v dnešnej dobe je málo aplikácií alebo dokonca žiadne, ktoré by zastrešovali úlohy na liečbu špecifických porúch učenia pod jednu tzv. grafickú užívateľskú strechu. Z tohto dôvodu je potrebné navrhnúť spoločné rozhranie, ktoré by umožňovalo prehľadný a jednoduchý prístup k týmto úlohám, jednotnú formu ovládania a možnosť zobrazenia nameraných dát. Najvhodnejším riešením je použitie klasickej ponuky tzv. „Menu“. Vďaka použitému frameworku, ktorý poskytuje základné rozhranie GUI pre tvorbu takýchto aplikácií, nie je potrebné vytvárať návrh úplne od začiatku. Framework poskytuje pomocou externého xml súboru možnosť vytvorenia menu a rozdeliť jednotlivé úlohy do skupín na základe ich zamerania, alebo nejakej inej príslušnosti. Rovnako tak poskytuje aj základnú jazykovú lokalizáciu. Keďže sa táto práca zameriava na tri konkrétne špecifické poruchy, ktoré je možné na

základe vybraných úloh rozdeliť do troch skupín, bude vytvorené menu aplikácie, kde bude možné medzi týmito skupinami prepínať (Obrázok 5.1).



Obrázok 5.1 - Menu aplikácie rozdeľujúce jednotlivé úlohy podľa skupín (položky Úvod a Kontakt sú iba informačné, neobsahujú žiadne úlohy).

Ďalšou nutnosťou je separovanie jednotlivých úloh do samostatných modulov. To zabezpečí oddelenie úloh, ich logiky a ich ovládania. Ďalšou výhodou takéhoto rozdelenia do modulov je to, že v prípade keď bude potrebné vytvoriť úlohu založenú na už implementovanej a funkčnej úlohe, čiže ide len o nastavenie vnútorných parametrov existujúcej úlohy, je možne vytvoriť nový odkaz na už existujúci modul úlohy a predať mu pri štarte potrebné nastavenie parametrov. Tým sa v podstate vytvorí úplne nová úloha.



Obrázok 5.2 - Ukážka rozdelenia skupiny úloh do jednotlivých modulov (modul predstavuje jeden typ úlohy).

5.2 Implementované úlohy

Ako už bolo spomenuté v analýze, pre túto prácu bolo vybraných 5 úloh. Tie reprezentujú úlohy určené na diagnostiku alebo rehabilitáciu troch uvažovaných špecifických porúch učenia, a to dyslexiu, dysgrafiu a dyskalkuliu. Táto kapitola sa zameria na návrh implementácie týchto úloh, z akých častí budú vytvorené a ako budú pracovať. Každá úloha je v jednej triede, ktorá predstavuje jeden tzv. modul celej aplikácie. Ide o triedu, ktorá dedí súčasti zo základnej triedy úlohy „`TaskBaseClass`“. Táto trieda obsahuje preddefinované metódy, ktoré uľahčujú tvorenie úloh. Napríklad ide o metódy určené na zachytávanie udalostí klikov na ovládacie tlačítka, zachytávanie udalosti zmeny veľkosti okna, zachytávanie udalostí myši, metódu, ktorá obsahuje

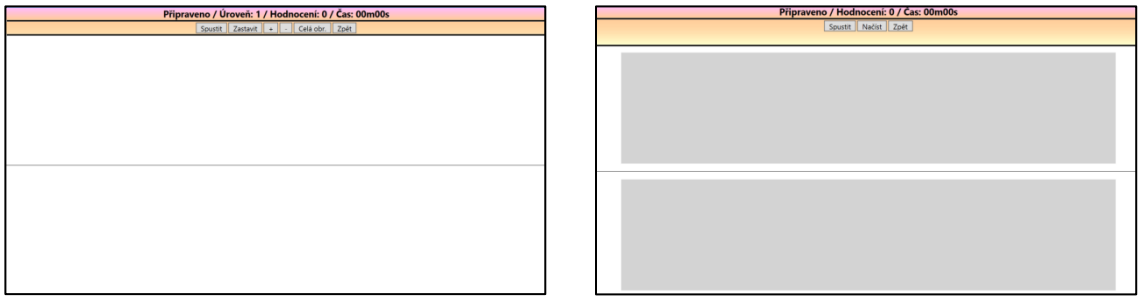
kód prepínajúci sa na základe stavu úlohy atď. Každý samostatný modul vytvára nové hracie plátno s informačným a ovládacím záhlavím.

5.2.1 Návrh úlohy „Rozdiely“

Pri analýze tejto úlohy bolo povedané, že daná úloha sa zameriava na precvičovanie alebo diagnostiku zrakového vnímania. Cieľom v tejto úlohe je hľadanie a vyznačovanie rozdielov medzi dvoma skoro identickými obrázkami. Od tohto cieľa sa rozvíja celý jej návrh.

Grafické rozhranie (GUI)

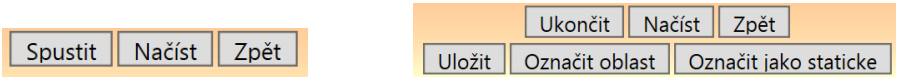
Keďže majú byť v úlohe zobrazené dva skoro rovnaké obrázky, je potrebné rozdeliť hlavnú hraciu plochu na dve časti (Obrázok 5.3).



Obrázok 5.3 - Ukážka rozdelenej hlavnej hracej plochy na dve časti, bez vložených hracích plátien (vľavo) a s pridanými hracími plátnami (vpravo).

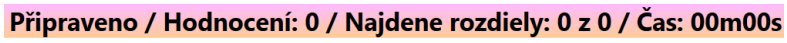
Použitý framework umožňuje jeho horizontálne rozdelenie na dve nezávislé časti, v ktorých sa môže nachádzať obyčajné plátno tzv. „Canvas“ alebo maticovo založene plátno tzv. „Grid“. Takéto rozdelenie umožní vytvoriť dva nezávislé obrázkové celky, ktoré budú zložené z menších, čiastočných obrázkov. Tieto obrázky je potrebné vkladať do týchto častí postupne a hlavne s možnosťou ľubovoľného výberu pozície v rámci hracej plochy konkrétnej časti (horná / spodná). Takéto možnosť ponúka len obyčajné plátno. To umožňuje položiť objekt na ľubovoľnú pozíciu v rámci jeho vnútornej ale aj vonkajšej plochy. Z tohto dôvodu obidve časti, horná aj spodná, budú obsahovať takéto plátno. Aby boli obidva obrazové celky jednoznačne rozoznateľné (kde začínajú a kde končia), bude potrebné farebne odlíšiť vnútorné hracie plátna od okolitého základu a zároveň vytvoriť jeho vonkajší okraj, ktorý ho definuje pre obe časti hlavnej hracej plochy (tzv. vlastnosť Margin) (Obrázok 5.3).

Ďalšou dôležitou súčasťou grafického rozhrania, je vytvorenie ovládania hry. Keďže majú byť tieto úlohy ovládateľné aj pomocou dotykových obrazoviek alebo tabletov, nie je možné použiť žiadne klávesy alebo klávesové skratky. Preto použitý framework ponúka vytvorenie ovládacej lišty, ktorá obsahuje tlačítka potrebné na ovládanie alebo konfiguráciu každej úlohy. Za týmto účelom bude potrebné vytvoriť sadu tlačítiek, pomocou ktorej bude užívateľ (pacient) spúšťať alebo zastavovať úlohu a zároveň lekár (kontrolná osoba) bude môcť vytvoriť novú verziu alebo prekonfigurovať už existujúcu úlohu. Pri takomto ovládaní, kedy môže k tlačítkam pristúpiť cez jednotné rozhranie lekár a zároveň aj pacient, bude potrebné vhodne sprístupňovať jednotlivé tlačítka na základe nejakých pravidiel, napríklad tým, že sa bude nastavovať tlačítkam viditeľnosť (Obrázok 5.4).



Obrázok 5.4 - Ukážka ovládacej lišty úlohy „Spájanie“. Tlačítka aktívne pred spustením úlohy (vľavo). Tlačítka aktívne pri tvorbe novej verzie úlohy (vpravo).

Poslednou súčasťou je informačná lišta, ktorá bude slúžiť na zobrazovanie priebehu úlohy, dosiahnutého skóre a času behu úlohu od jej štartu až po jej zastavenie (Obrázok 5.5).

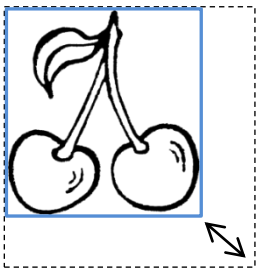


Obrázok 5.5 - Ukážka Informačnej lišty úlohy „Spájanie“

Vnútorné elementy

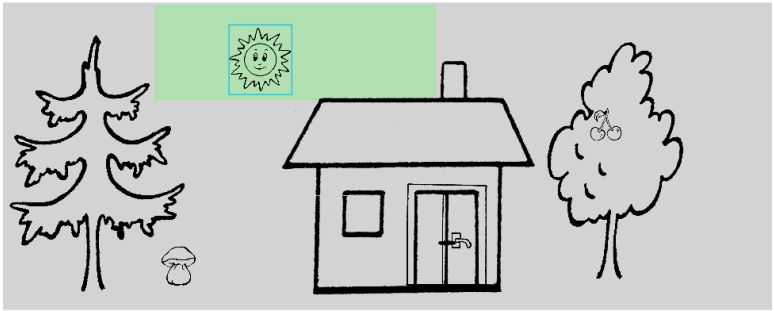
Keďže je potrebné aby bola úloha konfigurovateľná, je dôležité aby umožňovala vytvorenie rôznych variácií. V bežnej praxi by bolo možno postačujúce načítať dopredu vytvorené dva obrázky určené na hľadanie rozdielov a len im určiť oblasti kliknutia myšou, ktoré sú na obrázkoch odlišné. Ide asi o najjednoduchšie riešenie, ktoré však neumožňuje väčšiu voľnosť vo vytváraní a konfigurovaní nových verzií tejto úlohy. Ak chceme aby si užívateľ (lekár) mohol sám vyskladať vlastný obrázok a tak vytvoriť nový typ úlohy, je potrebné aby sa na hracie plátno vkladali obrázky jednotlivo. Vďaka tomuto bude možné ľubovoľné upravovať a tak vytvárať rôzne variácií úlohy s rovnakými obrázkami.

Keďže budú obrázky nahrávané zo súboru a ich veľkosť bude rovnaká bez ohľadu na pôvodné rozmery obrázka, je potrebné zabezpečiť aby sa obrázkom dala meniť veľkosť a tak ich prispôsobiť potrebám užívateľa. Najlepšie riešenie je zmena veľkosti ťahaním za roh obrázka, pričom je potrebné dodržať zachovanie pomeru strán, aby sa obrázok nedeformoval. Aby užívateľ vedel, ktorý obrázok upravuje, bude obrázok označený rámčekom, ktorý zároveň definuje jeho okraje a veľkosť, podľa ktorých bude možné obrázok rozťahovať (Obrázok 5.6).



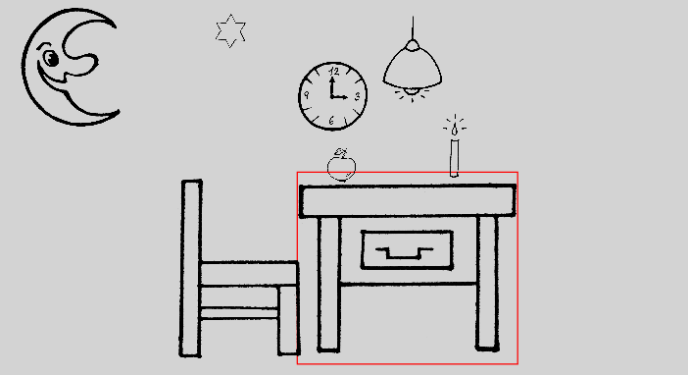
Obrázok 5.6 - Ukážka zmeny veľkosti obrázka a jeho označovacieho rámčeka (modrý rámček).

Aby sa umožnila variabilita pri vytváraní rôznych a hlavne náhodných obrázkov, ktoré bude užívateľ porovnávať, je výborným riešením umožniť náhodné generovanie jednotlivých čiastkových obrázkov v rozmedzí určitej vyznačenej oblasti. Z tohto dôvodu bude môcť tvorca úlohy (lekár) označiť vlastnoručne oblasť, v ktorej sa bude príslušný obrázok predstavujúci rozdiel generovať na náhodne vybranej pozícii z tejto oblasti. Princíp bude spočívať v tom, že si užívateľ označí objekt, ktorému chce takúto oblasť nastaviť a jednoducho mu túto oblasť vyznačí pomocou ťahania myši (ako klasické označovanie vo Windows, napríklad textu, alebo súborov). Samozrejme bude potrebné, aby bola daná oblasť zvýraznená inou farbou ako je okolie, aby bola dobre identifikovateľná (Obrázok 5.7).



Obrázok 5.7 - Ukážka vyznačenej oblasti, v ktorej sa bude daný obrázok náhodne generovať (zelená oblasť).

Posledná významná funkcionalita, ktorú si bude môcť tvorca úlohy nastaviť, je určenie čiastkového obrázka, ktorý má byť statický (Obrázok 5.8). To znamená, že tento obrázok pri generovaní hry nebude nikdy vymazaný. Je to podstatná vlastnosť, pretože pri štarte hry sa vygenerujú obidva porovnávané obrázky, pričom sa z oboch vymaže náhodný počet čiastkových obrázkov. Aby sa zabránilo vymazaniu niektorých kľúčových obrázkov (ako napríklad stôl z popod jablka, ktorý na ňom leží), je potrebné aby bol stôl definovaný ako statický, čo umožňuje určiť táto funkcionalita.



Obrázok 5.8 - Ukážka definície obrázku stola ako statického. Obrázok je vyznačený červeným rámčekom, aby užívateľ vedel, že ide o statický objekt.

Keďže cieľom tejto úlohy je porovnávanie dvoch rovnakých obrázkov, je potrebné zabezpečiť aby sa tieto obrázky líšili v určitom počte čiastkových obrázkov, z ktorých sú zložené. Preto sa pri generovaní bude musieť zabezpečiť náhodné mazanie niektorých čiastkových obrázkov tak, aby tie ktoré sa vymažú na vrchnom hracom plátne, neboli vymazané na spodnom. Toto zabezpečí rozdielnosť v oboch obrázkoch a zároveň to, že úloha bude vyzerať pri každom jej vygenerovaní vždy trochu inak, aj keď bude obsahovať tie isté čiastkové obrázky. Rovnako tak samotné rozdiely budú zobrazené na trochu inej pozícii v oblasti pre nich vždy určenej. Toto všetko významne zvyšuj variabilitu úlohy pri opakovanom použití.

Merané dáta

Z každej úlohy je potrebné zbierať určité dáta, ktoré sú potrebné pri samotnej diagnostike stavu pacienta. Pri riešení tejto úlohy, užívateľ označuje tie objekty, v ktorých sa dané obrázky líšia. Preto je potrebné sledovať a zaznamenávať jeho výber, či už samotný správny výber alebo nesprávne určený výber. Pri označení nesprávneho rozdielu sa navyše zaznamenáva, koľkokrát užívateľ takúto nesprávnu voľbu uskutočnil. Obidva sledované parametre sú dôležité pri

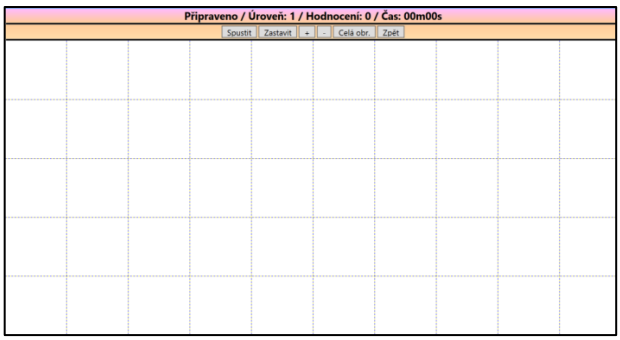
posudzovaní diagnózy, aby lekár mohol presne vidieť, ktoré tvary pacientovi (užívateľovi) robili problém pri ich rozoznávaní. Ďalšou významnou položkou, ktorá môže mať určitú výpovednú hodnotu, je čas za aký pacient danú úlohu dokončil, preto sa taktiež zaznamenáva a posiela ako jeden z meraných parametrov úlohy.

5.2.2 Návrh úlohy „Rôzne obrázky“

Ako už bolo spomenuté pri analýze tejto úlohy, jej zameraním je zrakové vnímanie. Úlohou užívateľa je spomedzi obrázkov, ktoré sú generované v riadku, alebo v matici, vybrať a označiť tie, ktoré sa od väčšiny obrázkov líšia napríklad tvarom alebo natočením.

Grafické rozhranie (GUI)

Zo samotnej špecifikácie úlohy je jednoduché vydedukovať, že táto úloha nebude potrebovať rozdelenie hracej plochy na viac častí. Ide o zobrazenie primitívnych obrázkov buď do riadku alebo do matice. Riadok sám o sebe je v podstate druh matice, ktorá je tvorená 1 riadkom a nejakým počtom n stĺpcov. Preto najvýhodnejším riešením bude použitie maticovo založeného hracieho plátna tzv. „Grid“, ktoré bude tvoriť hraciu plochu grafického rozhrania úlohy, pričom každé okienko bude obsahovať jeden obrázok (Obrázok 5.9). Ak bude potrebné zobraziť v každom riadku iný typ obrázkov, alebo zaplniť celú maticu rovnakým obrázkom, vždy bude princíp založení na vypĺňaní matice. Počet jednotlivých buniek matice bude možné nastaviť podľa potreby.



Obrázok 5.9 - Ukážka hlavnej hracej plochy obsahujúcej maticovo založené plátno (zobrazené čiary matice sú len pre ilustráciu).

Aj pri tejto úlohe je potrebné zabezpečiť jej ovládanie bez pomoci klávesnice, preto obsahuje ovládaciu lištu s tlačítkami. Tak ako to bolo aj v predošlej úlohe, aj tu budú jednotlivé tlačítka sprístupňované na základe stavu úlohy pomocou nastavenie ich viditeľnosti (Obrázok 5.4). Nad ovládacou lištou je umiestnený informačný text, ktorý informuje užívateľa o stave úlohy prípadne dosiahnutom skóre (Obrázok 5.5).

Vnútorné elementy

Pre splnenie jednotlivých požiadavkou na tvorbu úloh, bolo potrebné vytvoriť návrh, ktorý by ich pokrýval. Aby bola umožnená užívateľovi tvarovateľnosť úlohy podľa svojich predstáv a potrieb, bude potrebné zabezpečiť možnosť jednoduchej konfigurovateľnosti. Hlavný element tvoriaci túto úlohu je obrázok, preto bude celková možnosť nastavenia úlohy závislá práve na týchto elementoch.

Pri takomto type úlohy, kde je podstatou jej riešenia testovanie alebo precvičovanie užívateľovho zrakového vnímania, je potrebné poskytnúť možnosť narábania so samotným obrázkom tak, aby si sám návrhár zadania úlohy mohol určiť, aký obrázok chce použiť, ako ho chce natočiť alebo prevrátiť. Z tohto dôvodu bude možné definovať súbor obrázkov, ktoré sa majú v hre zobrazovať, jednoduchým nahraním týchto obrázkov do príslušného adresára úlohy. Sem bude užívateľ vkladať všetky obrázky, ktoré bude chcieť aby úloha zobrazovala. Ako už bolo povedané, v úlohe sa zobrazujú obrázky v riadku alebo v matici a hľadajú sa medzi nimi také, ktoré sa odlišujú od väčšiny. Odlišné obrázky sa dajú generovať dvoma spôsobmi:

1. Použitie dvoch fyzicky odlišných obrázkov. Čiže úloha bude generovať obrázky z dvoch odlišných súborov.
2. Použitie toho istého obrázka. Odlišnosť medzi obrázkami, ktoré hra vygeneruje, bude definovaná pomocou stanovenej rotácie obrázka, alebo zrkadlového otočenia.

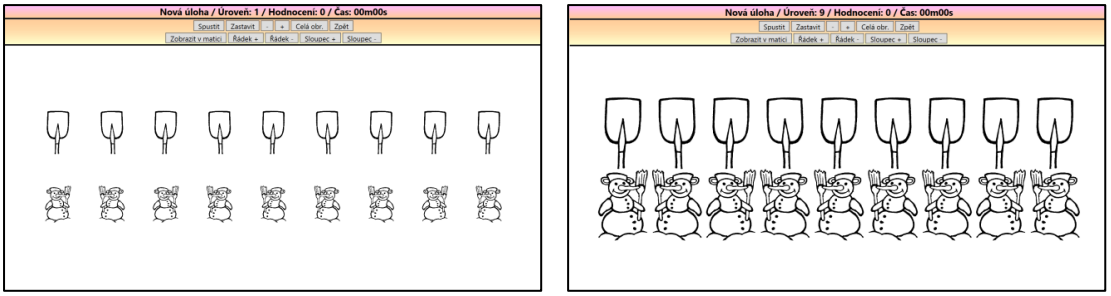
Keďže sú jednotlivé obrázky uložené v jednom konfiguračnom adresáre, bude potrebné zabezpečiť možnosť konfigurácie jednotlivých obrázkov užívateľom pomocou špecializovaného a presného názvu jednotlivých súborov s obrázkami. Z tohto dôvodu bol vytvorený presný názov súboru zložený z dvoch častí, ktoré sú oddelené pomlčkou:

1. Prvú časť tvorí číselný identifikátor obrázka. Ten slúži na identifikáciu obrázka alebo dvoch obrázkov, ktoré patria k sebe a tvoria jeden riadok po vygenerovaní v úlohe. Ide o 4 miestne číslo. Užívateľ si bude môcť zadefinovať tento identifikátor podľa seba tak, aby v prípade ak k danému obrázku patrí iný obrázok boli oba obrázky označené rovnakým identifikátorom.
2. Druhá časť je konfiguračná. Tá označuje typ obrázka a to, či má byť rotovaný alebo zrkadlovo otočený. Typ obrázka sa označuje dvoma písmenami, buď A alebo B. Obrázok typu A je ten, ktorý bude v úlohe vygenerovaný viackrát a typ B označuje obrázok, ktorý bude mať menšie početné zastúpenie a bude sa od druhého obrázka nejako odlišovať. Ak bude chcieť užívateľ nastaviť obrázku rotáciu, zrkadlové otočenie alebo obe naraz, zadefinuje to pomocou presne určených skratiek (napríklad Mir, Rot, Rot90, MirV atď.). Presnejší popis jednotlivých skratiek bude uvedený v časti dokumentácie.

Obrázok 5.10 - Príklad správne definovaného názvu súborov s obrázkami, ktoré patria k sebe. 0012-A predstavuje základný obrázok a 0012-B,Rot,Mir predstavuje druhý obrázok s možnosťou rotácie a zrkadlenia.

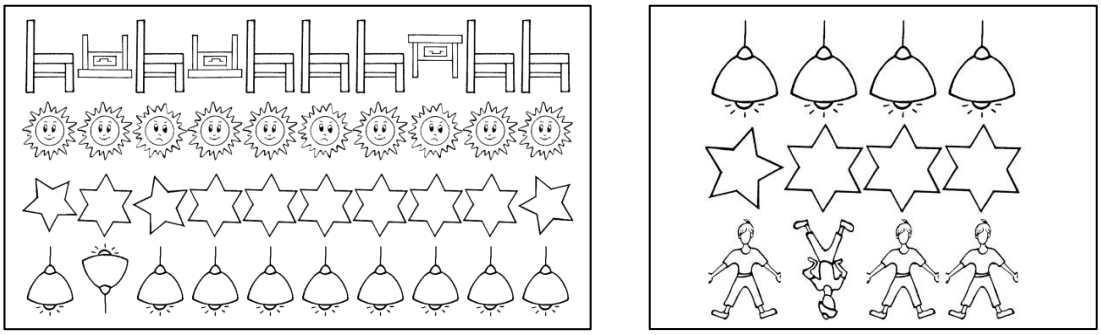
Užívateľ si takýmto spôsobom môže presne definovať nastavenie daného obrázka a to, či je obrázok sám alebo združený v dvojici s iným. Pri generovaní úlohy sa tento názov dekóduje a na základe vyššie spomenutých značiek sa vytvorí obrázkový objekt s príslušným nastavením. V prípade, že obrázok nemá definovaný presný uhol rotácie (90, 180, 270), alebo presný smer zrkadlového otočenia (vertikálne, horizontálne), generujú sa tieto hodnoty automaticky, čím sa zvyšuje rozmanitosť pri generovaní úlohy.

Pri takomto type úlohy, kde užívateľ hľadá rozdiely medzi vygenerovanými obrázkami, zohráva dôležitý faktor to, aké vzdialenosti majú medzi sebou jednotlivé obrázky. Čím sú obrázky tesnejšie na seba naviazané, tým to zmenšuje prehľadnosť, čo spôsobuje, že sa hľadajú rozdielne obrázky zložitejšie (Obrázok 5.11). Preto bude možné túto vzdialenosť nastaviť pomocou tlačítok na ovládacej lište úlohy.



Obrázok 5.11. - Príklad zobrazenia obrázkov s väčšími rozstupmi (vľavo) a s minimálnymi rozstupmi (vpravo).

Rovnaký efekt na zvyšovanie alebo znižovanie prehľadnosti má aj nastavenie počtu zobrazených riadkov a stĺpcov na hracej ploche, kde sa jednotlivé obrázky generujú (Obrázok 5.12). Preto bude umožnené tieto dva parametre meniť pomocou príslušných tlačítok.



Obrázok 5.12 - Príklad zmeny počtu obrázkov v riadku a v stĺpci.

Merané dáta

Keďže je táto úloha založená na podobnom princípe ako úloha „Rozdiely“, čiže na rozlišovaní medzi obrázkami, aj v tomto prípade sú hlavnými meraný veličinami výbery, ktoré robí užívateľ. To znamená, že je potrebné zaznamenávať každý výber užívateľa a zároveň počet nesprávnych klikov na príslušný obrázok. Aj pri tejto úlohe je významným parametrom samotný čas behu úlohy, preto je ho potrebné taktiež zahrnúť medzi merané dáta.

5.2.3 Návrh úlohy „Spájanie“

Tak ako predošle dve úlohy, aj táto sa zameriava na zrakové vnímanie. Avšak sa od nich značne odlišuje. Pri predošlých dvoch úlohách išlo v podstate o rozoznávanie rozdielnych obrázkov. Tu je

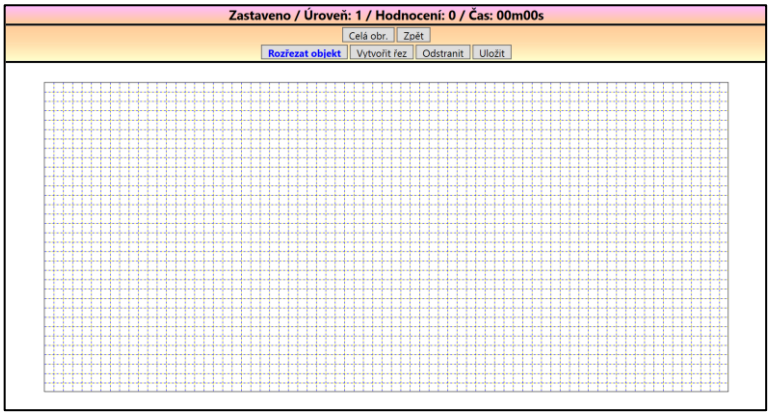
úlohou spájanie častí geometrických tvarov tak, aby po ich spojení vznikol celistvý geometrický objekt.

Grafické rozhranie

Samotná úloha je zložená z dvoch odlišných grafických rozhraní. Je to spôsobné charakterom tejto úlohy. Keďže ide o spájanie častí geometrických tvarov a je nutné brať do úvahy možnosti konfigurácie, bolo potrebné zabezpečiť aj možnosť vytvorenia takto rozrezaných objektov. Z tohto dôvodu bude potrebné rozdeliť grafické rozhranie na dve nezávislé časti a to herné (riešenie úlohy) a dizajnové (tvorba úlohy).

Dizajnové grafické rozhranie

Pri tvorbe rezov geometrického objektu je vhodné, aby bol čo najjednoduchšie vytvorený a hlavne aby dával užívateľovi, ktorý bude takýto rez vytvárať, voľnosť v jeho prevedení. Najjednoduchším riešením, ktoré by splnilo takéto riešenie, je vytváranie rezov na mriežkovom podklade. Ten pomocou možnosti označovania jednotlivých bodov rezu na priesečníkoch mriežky, umožňuje vytvoriť dostatočne ľubovoľný tvar rezu a tým pádom rozrezať objekt na menšie časti podľa potreby užívateľa. Preto bude na hlavnej hracej ploche vytvorené maticovo založené plátno (Grid), ktoré bude rozdelené na potrebný počet políčok (Obrázok 5.13).

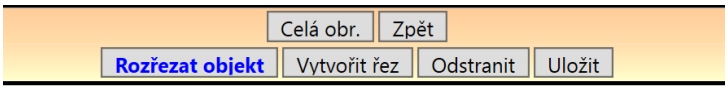


Obrázok 5.13 - Príklad dizajnového grafického rozhrania s mriežkou určenou na rozrezanie objektu.

Nevýhodou takéhoto plátna je komplikovanejšie vykresľovanie bodov na priesečníkoch mriežky a zároveň presné vykresľovanie čiar rezu. Grid totižto neumožňuje jednoducho položiť objekt na presné súradnice, ale len na číslo bunky (riadok x stĺpec). Aby sa spresnilo a zjednodušilo vykresľovanie bodov a rezov, bude Grid prekrytý obyčajným plátnom (Canvas), ktoré bude tvoriť

priesvitnú vrstvu, na ktorej sa budú tieto rezy vykresľovať. Grid bude slúžiť len pre vykreslenie mriežky. Takéto riešenie dvoch prekrytých plátien je najlepším riešením z hľadiska rýchlosti vykresľovania. Ak by sa aj mriežka generovala na Canvas-e pomocou samostatných čiar, značne by to spomalilo generovanie celého grafického rozhrania, pričom by sa musela zabezpečiť zmena veľkosti mriežky pri rozťahovaní okna aplikácie. Pri použití Grid-u sa mriežka vykreslí len na začiatku a samotná zmena veľkosti je zabezpečená automaticky, čo zrýchľuje generovanie a ovládanie celého dizajnového grafického rozhrania. Je potrebné zdôrazniť, že celá zobrazená mriežka reprezentuje jeden rezaný objekt, ktorým je obdĺžnik.

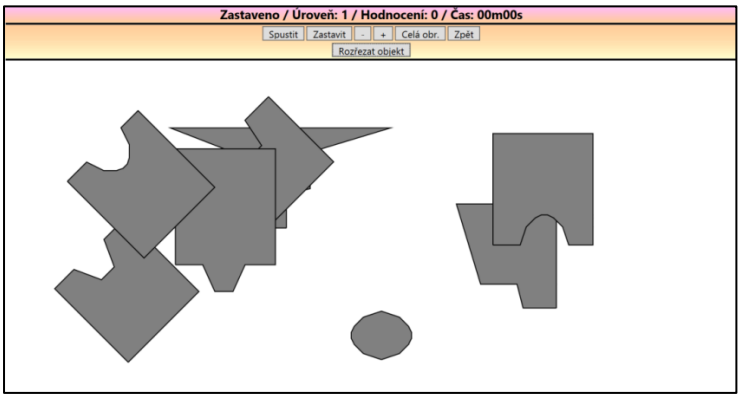
Ako každá úloha aj táto obsahuje ovládaciu lištu. V tomto grafickom rozhraní je tvorená tlačítkami určenými na vytváranie rezov a ich uloženie do konfiguračného súboru (Obrázok 5.14).



Obrázok 5.14 - Ovládacia lišta dizajnovacieho grafického rozhrania úlohy „Spájanie“.

Herné grafické rozhranie

Aby bola úloha aj hrateľná, je potrebné vytvoriť hraciu plochu, ktorá by dokázala zobraziť jeden alebo viacero rozrezaných objektov. Keďže ide o úlohou, v ktorej sa budú spájať rôzne objektu pomocou ich presúvania po hracej ploche, bude najvhodnejším riešením použiť obyčajné plátno (Canvas). Na jeho ploche sa dajú objekty vykresľovať po presných grafických bodoch tzv. pixloch, čo umožní vytvoriť dojem Drag and Drop, čiže presúvanie objektov z jedného miesta na druhé.

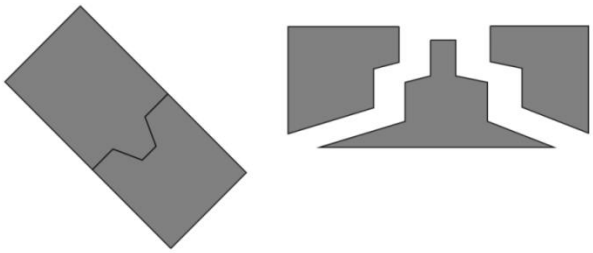


Obrázok 5.15 - Ukážka hracieho grafického rozhrania.

Ovládacia lišta bude tvorená klasickými tlačítkami na spustenie a zastavenie hry, roztiahnutím na celú obrazovku a tlačítkom na spustenie dizajnovacieho rozhrania. Nad ovládacou lištou sa bude nachádza informačná časť, ktorá bude ukazovať aktuálny stav úlohy, spolu s meračom času behu úlohy a úrovňou obtiažnosti.

Vnútorné elementy

Základným geometrickým tvarom, ktorý je v tejto úlohe rozrezávaný a teda skladaný je obdĺžnik. Obdĺžnik je tvarovo dobre prispôsobiteľný. Je možné ho rozrezať aj na iné výsledné geometrické objekty, ako napríklad štvorec, trojuholník, ale vďaka rezaniu na základe celkom jemnej mriežky aj na približný kruh. Preto bude obdĺžnik tvoriť základný geometrický tvar objektu úlohy. Aby sa zabezpečila variabilnosť úlohy a tým pádom náhodnosť pri generovaní jednotlivých častí objektov, budú tieto časti vykresľované na náhodnú pozíciu s nastavenou náhodnou rotáciou. Natočenie časti a tým pádom aj samotného objektu po jeho poskladaní je z praktických dôvodov pri ich spájaní len v násobkoch uhla 45°. Jednotlivé časti bude možné spájať v ľubovoľnom natočení, čiže nie je potrebné, aby si užívateľ jednotlivé objekty horizontálne vyrovnával pred ich samotným spájaním (Obrázok 5.16).



Obrázok 5.16 - Ukážka jednotlivých častí pri ich spájaní a možnosť ich natočenia.

Po poskladaní celého objektu musí mať užívateľ možnosť si takto poskladaný objekt presunúť na iné miesto ako celok. Preto sa časti správne poskladaného objektu zablokujú a objekt bude možné presúvať ako celok. Rozloženie na jednotlivé časti už správne poskladaného objektu nebude možné, pretože sa predpokladá, že užívateľ nebude takúto činnosť po jeho správnom poskladaní zvažovať. Pri takomto skladaní objektov je potrebné, aby jednotlivé časti do seba presne zapadli. Keďže je možné tieto časti posúvať po základných grafických bodoch (pixloch), je nutné aby sa tieto časti dostatočne automaticky spojili už pri ich priblížení na určitú minimálnu

vzdialenosť. Preto bude pri ich spájaní kontrolná vzdialenosť pre tieto objekty nastavená na väčší počet pixlov. Presná hodnota záleží na neskoršom otestovaní alebo nastavení.

Ďalšou funkcionalitou, ktorá zvyšuje variabilitu úlohy, je možnosť zafarbenia vykreslených objektov. Úloha bude obsahovať dva typy zafarbenia:

1. **Zafarbenie objektu** – Zafarbí sa celý objekt rovnakou farbou (všetky jeho časti majú rovnakú farbu) (Obrázok 5.17).
2. **Zafarbenie rezu** – Zafarbí sa každý rez objektu inou farbou (časti objektu, z ktorých je zložený majú rôznu farbu) (Obrázok 5.17).

Obrázok 5.17 - Ukážka zafarbenia objektu (vľavo) a rezu (vpravo).

Výber farby bude náhodný, pričom sa farba nikdy nebude opakovať u jednotlivých objektoch / častiach objektov v prípade, že bude definovaný dostatočný počet farieb na zafarbenie všetkých objektov / častí na hracom plátne. Užívateľ dostane možnosť vytvorenia vlastného zoznamu farieb pomocou úpravy konfiguračného súboru pre túto úlohu.

Merané dáta

Úloha sa bude zameriavať na zber dát z pohybu jednotlivých elementov pri ich spájaní, ktoré sú potrebné pre vizuálnu identifikáciu problémov pri riešení tejto úlohy. Rovnako tak bude uchovávať informácie aké dve časti sa užívateľ pokúšal, i neúspešne, spojiť. Okrem nich bude úloha, tak ako predošlé, zaznamenávať čas potrebný na jej dokončenie.

5.2.4 Návrh úlohy „Čísla“

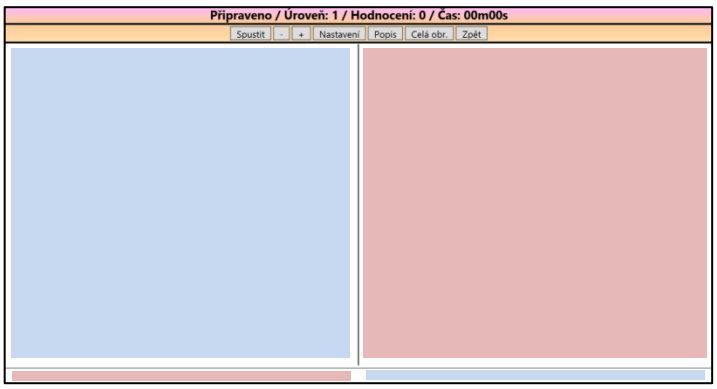
Zameranie tejto úlohy patrí do odlišnej kategórie ako doteraz popisované úlohy. Zameriava sa na číselné vnímanie (stanovenie počtu predmetov). Pri predošlých úlohách išlo zväčša

o univerzálnejšie použitie pre dyslexiu, dysgrafiu a dyskalkuliu, avšak v tomto prípade je využitie špecifikované hlavne na dyskalkuliu. Táto úloha bola rozdelená na dva moduly. Ide teda v podstate o dve samostatné úlohy, ktoré však zdieľajú rovnaký programový kód, pričom používajú len rozdielne nastavenie parametrov pre úlohu. V prvom module úlohy s názvom **„YesNo“** je cieľom označiť, či sa zobrazený počet bodiek rovná zobrazenému číslu. Pri druhom moduly s názvom **„MultipleChoice“** užívateľ vyberá zo zoznamu troch čísel to, ktoré sa rovná počtu zobrazených bodiek.

Grafické rozhranie (GUI)

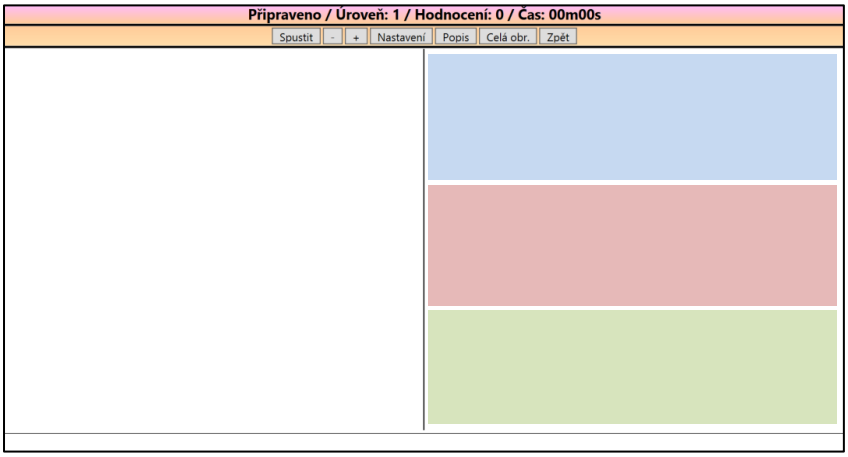
Grafické rozhranie u oboch modulov je v základnej štruktúre rozdelenia rovnaké, avšak tým, že moduly majú odlišný výber správnej odpovede respektíve čísla, sa v niektorých častiach GUI odlišuje.

Keďže je potrebné zabezpečiť, aby bolo možné zobraziť pre modul úlohy „YesNo“ tlačítka na výber správnej odpovede, je hlavná hracia plocha úlohy rozdelená na dve horizontálne časti, pričom vrchná tvorí približne 90% celkovej výšky hlavnej hracej plochy. Zvyšných približne 10% tvorí spodnú časť, ktorá bude slúžiť na zobrazenie tlačítiek „Áno“ a „Nie“ v prípade modulu “YesNo“ (Obrázok 5.18). Obidve časti (vrchná aj spodná) budú obsahovať maticovo založené hracie plátno „Grid“, z dôvodu sprístupnenia ďalšieho delenia týchto častí. Aby sa oddelila časť, v ktorej sa bude zobrazovať číselná hodnota od časti s vykreslenými bodkami, je potrebné vrchnú hraciu plochu rozdeliť vertikálne na dve nezávisle okná (Obrázok 5.18). Čo nebude problém, pretože vrchná časť je tvorená Grid-om. Toto umožní aby boli jednotlivé časti obsluhované samostatne.



Obrázok 5.18 - Rozdelenie hracej plochy na vrchnú a spodnú časť a následne vertikálne rozdelenie týchto častí na pravú a ľavú stranu.

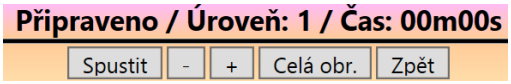
Keďže v oboch prípadoch ide v podstate o rovnaký programový kód, grafické rozhranie musí byť prispôsobené tak, aby dokázalo zobraziť obidva moduly len na základe odlišných konfiguračných parametrov. Preto bude potrebné aby pravá strana vrchného hracieho plátna obsahovala Grid, ktorý bude možné rozdeliť na taký počet riadkov alebo stĺpcov, koľko bude úloha aktuálne potrebovať (Obrázok 5.19). Takéto nastavenie bude napríklad potrebné pre zobrazenie možností čísel pre modul úlohy „MiltipleChoice“, ale taktiež umožňuje použitie pri tvorbe iných modulov tejto úlohy v budúcnosti. Ľavá strana vrchného hracieho plátna bude vykresľovať len bodky, preto je najlepším riešením použitie obyčajného plátna „Canvas“, ktoré umožní náhodné rozmiestnenie týchto bodov.



Obrázok 5.19 - Rozdelenie pravej strany vrchného hracieho plátna na tri horizontálne časti.

Ako bolo spomenuté vyššie, spodná časť hlavnej hracej plochy, bude určená na zobrazenie tlačítiek potrebných na riešenie modulu „YesNo“, kde sa musí užívateľ rozhodovať medzi odpoveďami Áno a Nie. Preto bude spodná časť taktiež rozdelená na ľavú a prvú stranu, v ktorých sa jednotlivé tlačítka zobrazia (Obrázok 5.18). Samozrejme túto časť hracej plochy bude možné v budúcnosti využiť aj na zobrazenie iných objektov než sú tlačítka. Modul úlohy „MultipleChoice“ v tejto časti hracej plochy nezobrazuje nič.

Samozrejmosťou bude aj ovládacia a informačná lišta zobrazená vo vrchnej časti okna. Ovládacia lišta bude obsahovať základné tlačítka na spustenie a zastavenie úlohy, ukončenie hry, rozšírenie okna na celú obrazovku, ale aj tlačítka na zvyšovanie respektíve znižovanie úrovne obtiažnosti („+“, „-“) . Na informačnej lište bude zobrazený aktuálny stav úlohy, úroveň obtiažnosti a čas behu úlohy (Obrázok 5.20).

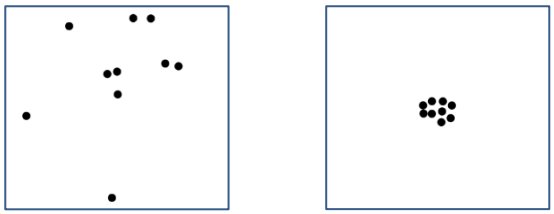


Obrázok 5.20 - Ovládacia a informačná lišta úlohy „Čísla“.

Vnútorné elementy

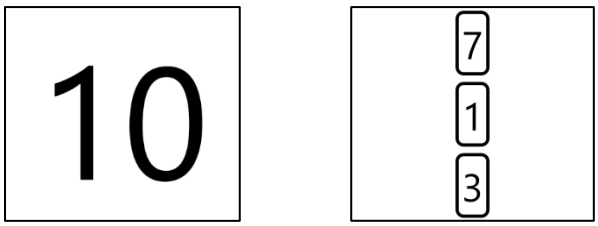
Táto úloha je z hľadiska objektov, ktoré sa vykresľujú na jej hracej ploche jednou z jednoduchších v porovnaní s predošlými úlohami. Obsah hracej plochy je tvorený len zobrazenými číslami na ľavej strane hracej plochy a bodkami na pravej strane hracej plochy.

Bodky budú tvorené obyčajnou elipsou zafarbenou čiernou farbou. Ako už bolo popísané v časti grafického rozhrania, budú zobrazené na obyčajnom plátne, čo umožní ich umiestenie na náhodne vygenerovanú pozíciu. Takéto riešenie je potrebné, pretože užívatelia (v tomto prípade pacienti) by nemali mať možnosť vidieť vždy rovnaké rozloženie bodiek. Samozrejme bude potrebné zabezpečiť, aby sa jednotlivé bodky neprekrývali a tak nezabraňovali užívateľovi v možnosti ich spočítanie. Z dôvodu potreby umožnenia zvyšovania alebo znižovania obtiažnosti úlohy, bude možné zmenšiť alebo zväčšiť oblasť, v ktorej sa bodky budú môcť generovať. Čím bude úroveň obtiažnosti vyššia tým budú bodky na seba viac zahustené a naopak, čím bude obtiažnosť menšia tým budú body od seba generované vo väčšej vzdialenosti (Obrázok 5.21).



Obrázok 5.21 - Ukážka generovania bodiek na hracom plátne.

Vrchná pravá strana hracej plochy má zobrazovať čísla, s ktorými bude užívateľ porovnávať vykreslené bodky z ľavej strany. Aby sa zabezpečila väčšia flexibilita a možnosť konfigurácie hry a vytvárania nových modulov úlohy, bude možné tieto čísla zobraziť v ľubovoľnom počte riadkov respektíve stĺpcov. Bude potrebné dohliadnuť, aby zobrazené čísla boli vždy vycentrované a aby sa vždy ich veľkosť prispôsobila veľkosti pravej hracej plochy a počte týchto čísel na ploche (Obrázok 5.22).



Obrázok 5.22 - Ukážka zobrazenia čísel na pravej strane vrchného hracieho plátna z modulov "YesNo" (vľavo) a "MultipleChoice" (vpravo).

Merané dáta

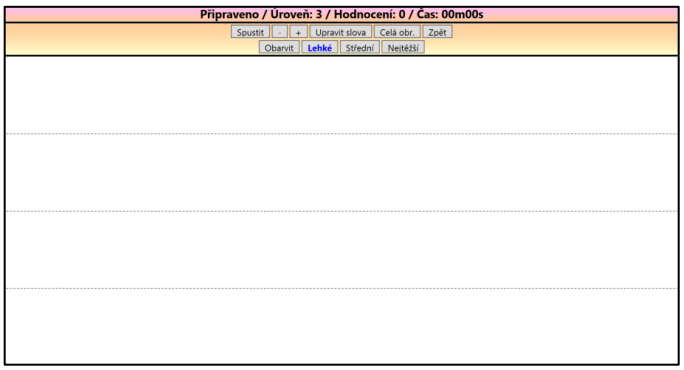
Najdôležitejšie parametre tejto úlohy, ktoré sú potrebné pre samotného lekára, aby dokázal identifikovať aktuálny stav pacienta je porovnanie výberu, ktorý spravil pacient a skutočne správnej odpovede. V module „YesNo“ sú týmito dátami odpovede „Áno“, „Nie“ a zároveň číslo, ktorého sa táto odpoveď týkala. Na druhej strane pri module úlohy „MultipleChoice“ sú takýmto meranými parametrami len samotné čísla a to vybrané číslo a číslo, ktoré predstavuje naozaj skutočne správnu odpoveď. Ako to bolo aj pri predošlých úlohách aj tu je jedným z meraných parametrov čas za aký užívateľ úlohu dokončil.

5.2.5 Návrh úlohy „Správne slovo“

Aj posedná vybraná úloha sa zameriava konkrétnejšie na typ špecifických porúch, kde je potrebné diagnostikovať čítaný prejav pacienta, poprípade aj písaný. Preto jeho zameranie bolo pridané do kategórie vnímania textu. Úlohou užívateľa je označiť spomedzi zoznamu slov to jediné, ktoré je napísané správne. Aj táto úloha je rozdelená na dva moduly. Ich rozdiel spočíva v načítaní odlišných konfiguračných súborov. Preto bude návrh úlohy popisovaný len z pohľadu samotnej úlohy a nie jednotlivých modulov.

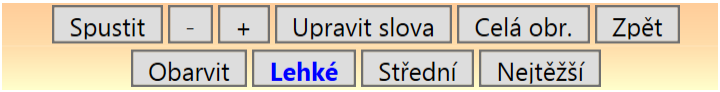
Grafické rozhranie (GUI)

Pri tejto úlohe nebude potrebné zložité rozdeľovanie grafické rozhrania na niekoľko častí. Keďže je potrebné zobraziť len zoznam slov, z ktorých má užívateľ vyberať, bude postačujúce využiť hlavnú hraciu plochu obsahujúcu maticovo založené plátno „Grid“, ktoré bude obsahovať toľko riadkov, koľko slov bude úloha generovať (Obrázok 5.23). Každý riadok bude následne zobrazovať jedno slovo.



Obrázok 5.23 - Príklad rozdelenia hracieho plátna na 4 horizontálne časti pre zobrazenie 4 slov.

Ovládacia lišta pri tejto úlohe bude obsahovať základné tlačítka na spustenie a zastavenie úlohy, avšak z dôvodu možnosti konfigurácie úlohy, bude obsahovať tlačítka, pomocou ktorých je možné ovládať úroveň obtiažnosti a tak nastavovať rôzne verzie úlohy. Ide o špecifickejšie tlačítka, pretože pri tejto úlohe nie je nastavenie úrovne závislé len na jednom parametre. Ide o skupinu parametrov, ako počet zobrazených slov, úroveň slova a zafarbenie slov. Preto bude potrebné vytvoriť jednotlivé tlačítka na ich ovládanie. „+“ a „-“ bude slúžiť na určenie počtu zobrazených slov, „Obarvit“ bude zapínať alebo vypínať zafarbenie slov a tlačítka „Lehké“, „Střední“ a „Nejtěžší“ budú slúžiť na výber obtiažnosti slova (Obrázok 5.24). Lišta bude obsahovať aj tlačítko „Upravit slova“ na otvorenie konfiguračného súboru z dôvodu možnosti jeho priameho a jednoduchého nastavenia z rozhrania úlohy.



Obrázok 5.24 - Ovládacia lišta úlohy "Správne slovo" s konfiguračnými tlačítkami.

Samozrejmosťou bude aj informačná lišta, ktorá bude informovať užívateľa o aktuálnom stave úlohy, obsahujúca informácie ako sú aktuálna úroveň obtiažnosti, aktuálny stav a čas za aký bola úloha dokončená (Obrázok 5.20).

Vnútorné elementy

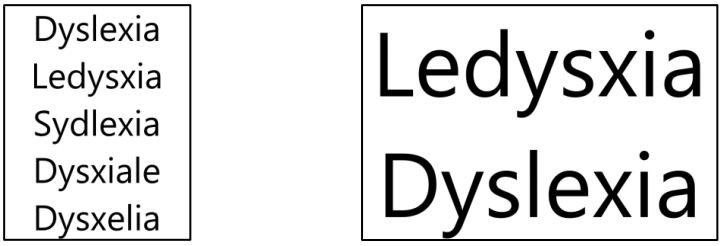
Ako už bolo spomenuté vyššie, základným elementom úlohy je textové pole. Pri tejto úlohe je podstatnou podmienkou umožniť osobe, ktorá bude jednotlivé úlohy nastavovať, aby mohla vytvoriť vlastný zoznam slov, z ktorého sa jednotlivé slová budú načítavať do úlohy. Preto je

najlepším riešením vytvorenie externého konfiguračného súboru, ktorý bude obsahovať definície jednotlivých slov. Užívateľ (lekár) si bude môcť slová zadefinovať podľa potreby na základe pacientovho stavu respektíve vhodnosti slov na testovanie poruchy. Keďže je možné, že tento súbor bude upravovaný osobou, ktorá nemá IT vzdelanie, nie je vhodné použiť napríklad bežne používané xml súbory, ale nejakú jednoduchšiu variantu, najlepšie celkom obyčajný textový súbor.

Aby sa zvýšila flexibilita a možnosť lepšieho upravovania úlohy, bude k dispozícii možnosť zafarbenia vygenerovaných slov v úlohe. Tie sa budú farbiť podľa zoznamu farieb, ktorý bude definovaný taktiež v konfiguračnom súbore, čo umožní jeho tvorbu alebo úpravu podľa požiadavkou užívateľa. Slová sa budú farbiť tak, aby sa žiadna z farieb neopakovala viackrát, ak to nebude potrebné z dôvodu väčšieho počtu zobrazených slov ako počtu definovaných farieb (Obrázok 5.25).

Obrázok 5.25 - Ukážka zafarbenia slov v úlohe.

Keďže je potrebné aby bolo úlohu vhodne modifikovať, bude sa dať v úlohe meniť jej obtiažnosť na základe niektorých parametrov. Ako už bolo spomenuté pri popise grafického rozhrania, ide o parametre počtu zobrazených slov, už spomínaného sfarbenia a obtiažnosť samotného slova. Počet zobrazených slov znamená, že vygenerované slová v úlohe budú obsahovať vždy iba jedno správne napísané slovo a zvyšok budú tvoriť slová, ktoré sú definované v konfiguračnom súbore ako nesprávne (Obrázok 5.26).



Obrázok 5.26 - Ukážka zobrazenia väčšieho (vľavo) a menšieho (vpravo) počtu slov v úlohe.

Ak bude nesprávnych slov menej ako nastavená úroveň počtu generovaných slov v úlohe, tak sa niektoré z nesprávnych slov vygeneruje znova, bez ohľadu na to, či už bolo takéto slovo vygenerované (slová sa generujú cyklicky). Ako príklad môže slúžiť Obrázok 5.25, kde je vidieť dvakrát vygenerované slovo **„jadlko“**, pretože zoznam definovaných slov v konfiguračnom súbore obsahuje len 3 nesprávne slová.

Obtiažnosť slova je definovaná v konfiguračnom súbore. Na základe vybranej obtiažnosti slova, sa generujú zo súboru len tie zoznamy slov, ktoré sú označené touto úrovňou.

Merané dáta

Aj pri tejto úlohe je dôležité zaznamenávať rozdiely medzi skutočne správnou odpoveďou a užívateľom (pacientom) vybraným slovom, aby mohol lekár vidieť presné rozdiely medzi týmito dvoma slovami. Preto bude zaznamenávané presné znenie slova, ktoré vybral pacient a slova, ktoré je skutočne správnou odpoveďou. Aj pri tejto úlohe sa zaznamená čas za aký bola úloha dokončená.

5.3 Implementované rozšírenia frameworku

V analýze bolo spomenuté, že použitý framework je potrebné rozšíriť o možnosti zobrazenia nameraných a hlavne následne spracovaných dát. Preto boli navrhnuté tri typy zobrazení a to grafové, jednoduchá tabuľka a kalendár. Vytvorené typy zobrazení vychádzajú hlavne z vyššie popisovaných typov úloh. Všetky tieto rozšírenia sú vytvorené rovnakou formou ako jednotlivé implementované úlohy, čiže využívajú prázdnu triedu modulu úlohy. Preto je aj samotné spúšťanie a ovládanie týchto rozšírení založené na rovnakom princípe, ako u samotných úloh. Nasledujúca časť sa bude venovať návrhu týchto rozšírení z pohľadu grafického rozhrania, základných vnútorných elementov a typu dát, pre ktoré sú určené.

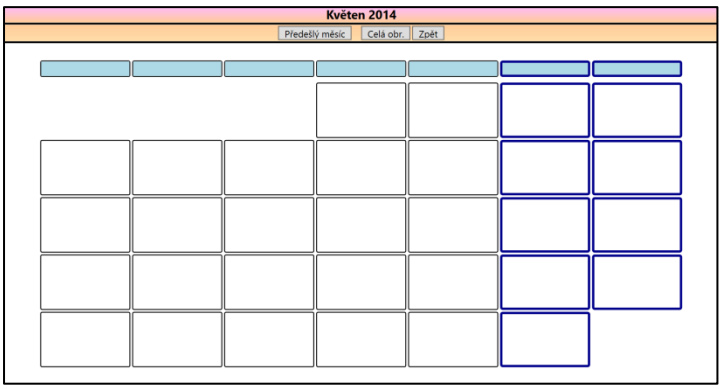
5.3.1 Návrh rozšírenia „Používanie“

Ide o zobrazenie typu kalendár, ktoré poskytuje určité pridané schopnosti. Už názov tohto rozšírenia, hovorí o tom, že pôjde o zobrazenie dát popisujúcich používanie jednotlivých úloh

užívateľom (pacientom) za určité obdobie. Z tohto dôvodu bol vybraný návrh formy kalendára, na ktorom sa dá vhodne a presne zobraziť obdobie a stavy používania danej úlohy.

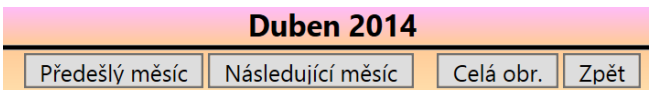
Grafické rozhranie (GUI)

Základnou štruktúrou kalendára je mesiac obsahujúci jednotlivé dni. Z tohto bude vychádzať aj návrh tohto rozšírenia. Ako základná plocha pre zobrazenie jedného mesiaca bude použité maticovo založené plátno, ktoré sa rozdelí na jednotlivé políčka reprezentujúce dni mesiaca. Matica bude mať fixný počet siedmych stĺpcov a variabilný počet riadkov, ktorý sa bude meniť v rozmedzí 5 až 7 riadkov. Je to spôsobené posunom začiatočného dňa mesiaca v rámci jedného týždňa. Najvrchnejší riadok matice je venovaný hlavičke kalendára, v ktorej budú zobrazené dni týždňa (Obrázok 5.27).



Obrázok 5.27 - Ukážka rozdelenia plátna na jednotlivé políčka dní a hlavičky kalendára.

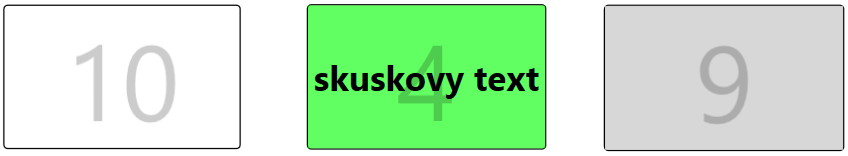
Ovládacia lišta kalendára bude značne jednoduchá. Bude obsahovať len tlačítka na prepínanie jednotlivých mesiacov, ktoré budú sprístupňované na základe toho, či je možné prepnutie na ďalší respektíve predošlý mesiac. Okrem nich bude možné zapnutie celoobrazovkového zobrazenia a možnosť vrátenia sa do hlavného menu aplikácie. Informačná lišta zobrazuje len mesiac a rok aktuálne zobrazovaného kalendára (Obrázok 5.28).



Obrázok 5.28 – Ovládacia a informačná lišta rozšírenia frameworku „Kalendár“.

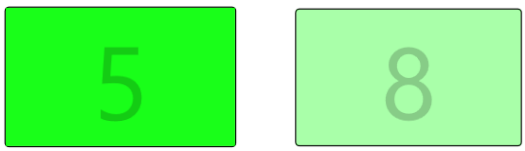
Vnútorné elementy

Základným elementom kalendára je jeden deň, ktorý bude reprezentovaný pomocou dvoch textových blokov. Prvý je potrebný z dôvodu zobrazenia číselného údaju o dni pričom druhý bude slúžiť na zobrazenie prípadných textových poznámok, ktoré môžu špecifikovať zobrazené informácie pre daný deň. Aby bola poznámka dňa čitateľná, bude potrebné, aby bolo číslo dňa svetlejšie ako text poznámky (Obrázok 5.29).



Obrázok 5.29 - Príklad zobrazenia dňa, kedy užívateľ nemal úlohu prístupnú (vľavo), kedy sa úlohe venoval (v strede) a kedy sa jej nevenoval (vpravo) + príklad dňa bez poznámky (vľavo) a s poznámkou (v strede).

Keďže je úlohou tohto rozšírenia zobrazenie napríklad frekvencie používania úlohy užívateľom (za jednotlivé dni), je potrebné rozlíšiť dni kedy bola úloha použitá, kedy sa jej užívateľ nevenoval a dni kedy užívateľ ešte nemal k úlohe prístup. Najlepšie riešenie pre takéto rozlišovanie údajov je ich farebné rozlíšenie. Preto bude každý deň obsahovať možnosť zafarbenia pozadia, ktoré bude definovať stav použitia úlohy (Obrázok 5.29). Aby bolo možné zobraziť intenzitu používania úlohy za príslušný deň, bude pri dňoch, kedy bola aplikácia použitá, nastavená sýtosť farby podľa toho, akú intenzitu bude reprezentovať (čím vyššia intenzita tým bude farba sýtejšia) (Obrázok 5.30).



Obrázok 5.30 - Príklad intenzívnejšieho (vľavo) a menej intenzívnejšieho (vpravo) riešenia úloh.

Dobrým zvykom pri tvorbe kalendára je odlíšenie pracovných dní od víkendových, preto budú rámčeky víkendových dní zafarbené na modro (Obrázok 5.31).

Obrázok 5.31 - Označenie víkendového dňa.

Hlavička kalendára, ktorá zobrazuje názvy dní, bude tvorená rovnakým elementom ako jednotlivé dni, pričom sa využije možnosť zobrazenia poznámky v tomto elemente. Namiesto ľubovoľnej poznámky bude text obsahovať vždy názov dňa (Obrázok 5.32).

Obrázok 5.32 - Hlavička kalendára.

Zobrazované dáta

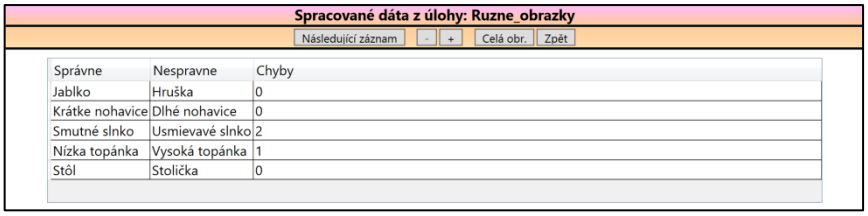
Ako už bolo spomenuté vyššie, cieľom tohto rozšírenia je zobrazenie používania konkrétnej úlohy za určité časové obdobie. Preto je toto rozšírenie určené na zobrazovanie hlavne spracovaných dátumových alebo časových údajov pre jednotlivé úlohy.

5.3.2 Návrh rozšírenia „Tabuľka“

Nasledujúce rozšírenie je založené na použití jednoduchej tabuľky. Ako už bolo spomenuté v analýze tohto rozšírenia, vzniklo z dôvodu nevhodnosti použitia grafových alebo iných zobrazení pre určitý špecifický typ dát. Ide o veľmi jednoduchý typ tabuľkového zobrazenia, preto bude nasledujúci návrh stručný.

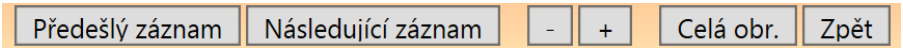
Grafické rozhranie (GUI)

Grafické rozhranie tohto rozšírenia bude vytvorené najjednoduchším možným spôsobom. Keďže ide len o jednoduchú tabuľku, nie je potrené vytvárať zložité rozdelenie grafickej plochy. Tá bude obsahovať len maticovo založené plátno „Grid“, ktoré nebude nijak delené na menšie časti. Jeho využitie má len zarovnávaciu funkciu, pretože každý objekt pridaný do bunky Grid-u je automaticky zarovnaný na stred. Keďže bude implementácia tvorená pomocou vybranej platformy Microsoft .NET, bude použitá jeho súčasť a to tzv. „DataGird“. Ide o reprezentáciu dátovej tabuľky, ktorú je možné upraviť na základe ľubovoľných požiadaviek (Obrázok 5.33).



Obrázok 5.33 - Ukážka dátovej tabuľky.

Prepínanie medzi jednotlivými tabuľkami zobrazujúcimi rôzne dáta z rôznych úloh bude možné pomocou tlačítiek „Předešlý záznam“ a „Nasledující záznam“, ktoré sa budú sprístupňovať na základe dostupnosti tabuliek. Budú umiestené na ovládacej lište. Okrem nich bude možné zmenšiť alebo zväčšiť písmo v tabuľke pomocou tlačítiek „+“ a „-“. Táto funkcionalita má čisto praktický dôvod, aby si mohol užívateľ (lekár) prispôsobiť veľkosť písma v prípade potreby.



Obrázok 5.34 - Ovládacia lišta rozšírenia frameworku "Tabulka".

Vnútorné elementy

Jediným elementom použitým v tomto rozšírení je dátová tabuľka. Jej hlavnou výhodou je možnosť načítania externého xml súboru s dátami priamo do tejto tabuľky. Je postačujúce, aby sa spracované dáta, ktoré sa budú zobrazovať v tejto tabuľke, uložili do externého xml súboru s definovanou správnou štruktúrou obsahu a následne je ho možné bez akýchkoľvek problémov načítať do dátovej tabuľky (Obrázok 5.33).

Zobrazované dáta

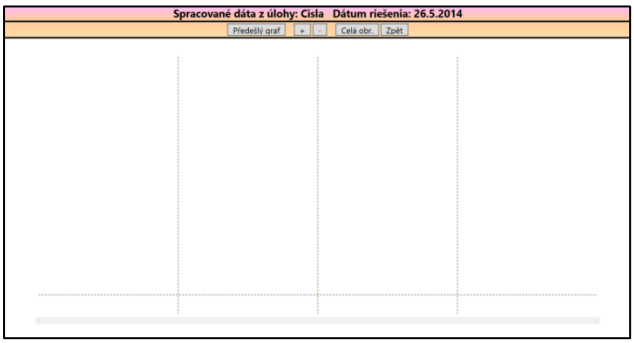
Hlavným dôvodom použitia tohto typu zobrazenia je ten, že niektoré spracované výsledky nie je možné vhodne zobraziť pomocou grafov. Je potrebné ich porovnávať z hľadiska ich významu a obsahu, ako napríklad výber správneho slova zo skupiny slov. Preto sa toto rozšírenie bude používať na dáta takéhoto typu. Napríklad na porovnanie výberu užívateľa so skutočne správnou odpoveďou z úlohy „Správne slovo“.

5.3.3 Návrh rozšírenia „Graf“

Posledným rozšírením použitého frameworku „Graf“. Ide o typ zobrazenia dát pomocou jednoduchého stĺpcového grafu, ktorý je vhodný na porovnávanie úspešnosti riešenia úloh za určité časové obdobie.

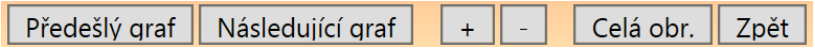
Grafické rozhranie (GUI)

Aby bol graf čo najjednoduchšie vytvorený, nebudú použité pri jeho tvorbe žiadne externé knižnice určené na tvorbu grafov. Najlepším riešením je použitie maticovo založeného plátna „Grid“, ktoré bude rozdelené na jednotlivé stĺpce, pričom bude obsahovať dva riadky. Prvý riadok bude zaberať väčšinovú časť celéj výšky Grid-u a bude slúžiť na zobrazenie jednotlivých stĺpcov grafu. Druhý riadok bude obsahovať hlavičku grafu s popisom jednotlivých stĺpcov. Toto rozdelenie bude reprezentovať štruktúru stĺpcového grafu (Obrázok 5.35).



Obrázok 5.35 - Príklad rozdelenia plátna, ktoré bude vykresľovať 4-stĺpcový graf s hlavičkami.

Tak ako to bolo aj pri predošlých formách zobrazovania dát, aj tu bude ovládacia lišta tvorená tlačítkami na prechod medzi nameranými dátami (grafmi) „Předešlý graf“ a „Nasledující graf“. Okrem nich bude obsahovať tlačítka „+“ a „-“, ktorými si bude môcť užívateľ zväčšiť alebo zmenšiť vykreslenie grafu v prípade, že bude graf obsahovať väčší počet stĺpcov, ktoré budú vykreslené z dôvodu počtu príliš na tenko (Obrázok 5.36).



Obrázok 5.36 - Ovládacia lišta rozšírenia framewroku „Graf“.

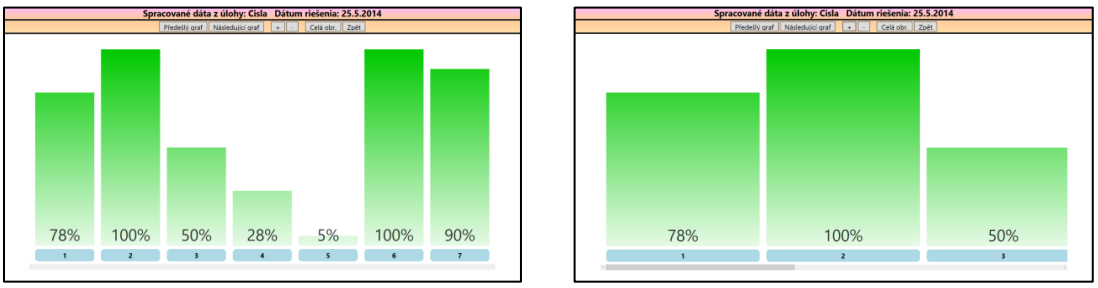
Informačný panel bude taktiež súčasťou tohto rozšírenia. Jeho úlohou bude zobrazenie názvu úlohy, ktorej sa tieto dáta týkajú a dátumu, kedy bola úloha riešená, čiže kedy boli dáta namerané (Obrázok 5.37).

Spracované dáta z úlohy: Cisla Dátum riešenia: 25.5.2014

Obrázok 5.37 - Ukážka informačnej lišty rozšírenia frameworku „Graf“.

Vnútorné elementy

Základným elementom, ktorý bude tvoriť stĺpec grafu je obyčajný obdĺžnik. Ide o najvhodnejšiu a najbežnejšiu formu pre zobrazenie stĺpca grafu. Jeho výhodou je možnosť zafarbenia poprípade zaoblenia hrán na koncoch grafu. Aby bol vytvorený čo najdôveryhodnejší, ale hlavne čo najlepšie čitateľný graf, bude jeho zafarbenie vytvorené pomocou gradientu farby. Jej sýtosť sa bude na základe percentuálnej hodnoty líšiť, aby bol docielený nie len výškový vizuálny efekt stĺpcov pri čítaní rozdielu hodnôt, ale aj ten farebný. Čím bude hodnota, ktorú daný stĺpec reprezentuje nižšia, tým bude sýtosť farby menšia a zas naopak, čím bude hodnota vyššia, tým bude sýtosť farby výraznejšia. Jednotlivým obdĺžnikom bude potrebné nastaviť vonkajší okraj, aby boli stĺpce grafu dobre rozlíšiteľné a nesplývali. (Obrázok 5.38).



Obrázok 5.38 - Ukážka zobrazenia dát v grafe v normálnej forme (vľavo) a v priblíženej forme (vpravo). Na obrázku je možné vidieť farebné rozdielny stĺpcov na základe ich hodnoty, ktorú reprezentujú.

Keďže bude možné graf zmenšovať alebo zväčšovať, bude potrebné zabezpečiť aby si užívateľ mohol graf prezrieť celý, aj keď sa mu jeho časť skryje za okraj plochy. Preto je celý graf vložený do tzv. „ScroolView“, ktorý zabezpečí automatickú možnosť posúvania grafu pomocou okrajových posuvníkov (Obrázok 5.38).

Hlavičky grafu budú tvorené textovými blokmi, ktoré tak umožnia vložiť ľubovoľný text do príslušnej hlavičky (Obrázok 5.39).

Obrázok 5.39 - Hlavičky grafu.

Zobrazované dáta

Hlavným zameraní tohto rozšírenia je porovnanie úspešnosti riešenia nejakej úlohy za určitý počet jej spustení, alebo za nejaké časové obdobie. Preto spracované dáta, ktoré zobrazuje musia pochádzať z dlhšieho časového obdobia a ich hodnoty musia byť v relatívnej percentuálnej hodnote.

6 Niektoré podrobnosti z implementácie aplikácie

Pri implementácii jednotlivých úloh a rozšírení frameworku, boli použité rôzne metódy a objekty, ktoré ponúka použitá technológia Microsoft WPF.NET a ktoré uľahčili tvorbu rôznych špecifických vlastností úloh. V nasledujúcej kapitole budú popísané niektoré neštandardné riešenia použité pri tvorbe úloh. Boli vybrané dve takéto riešenia a to konkrétne spájanie rozrezaných častí objektov v úlohe **„Spájanie“** a animácia stĺpcov grafu v rozšírení frameworku **„Graf“.** Ostatné vytvorené úlohy nie je potrebné v tejto kapitole nijak podrobne rozoberať, pretože využívajú väčšinou štandardné programovacie techniky a postupy.

6.1 Spájanie objektov v úlohe „Spájanie“

Spájanie objektov je hlavná funkcia úlohy „Spájanie“, kde užívateľ pomocou presúvania rozrezaných častí k sebe skladá dohromady jednotlivé objekty. Popis tejto funkcie bol vybraný z dôvodu, že boli použité špecifické možnosti technológie Microsoft WPF.NET, ktoré tak spravili z tejto súčasti úlohy neštandardne poňaté riešenie. Nejde však o využívanie zložitých algoritmov, ale skôr o to ako bola táto funkcia jednoducho a hlavne účelne implementovaná.

Funkcia spájania využíva možnosť vlastnosti objektov tzv. **Clip**, ktorá je určená pre vizuálne orezanie objektu na základe preddefinovaného vzoru. Tento vzor môže byť vytvorený buď pomocou tried definujúcich geometrické tvary (obdĺžnik, elipsa a iné.) alebo pomocou tzv. cesty, ktorá je definovaná kolekciou bodov, pomocou ktorej je možné vytvoriť ľubovoľný geometrický tvar. Metóda vytvárajúca tento rez sa nazýva **SetClip**(), ktorá vo svojom parametre prijíma kolekciu bodov, ktoré reprezentujú daný rez. Metóda je implementovaná v triede **Element2.cs** reprezentujúcej rez objektu. Keďže pri tvorbe rezu má užívateľ voľnosť v jeho prevedení, je využitá v implementácii práve zmienená cesta. Konkrétne ide o triedu tzv. geometrickej cesty `PathGeometry`, ktorá sa po jej definovaní priradí do clip-u daného objektu (Kód 1).

```
// vytvorenie objektu geometrickej cesty, ktorá bude definovať orezanie
   objektu
PathGeometry gPath = new PathGeometry();
// pridanie figúry cesty no zoznamu figúr geometrickej cesty
gPath.Figures.Add(new PathFigure(clipPoints.First(), psCollection, true));
// priradenie geometrickej cesty do 'Clip-u' objektu, ktorý ho oreže
basicShape.Clip = gPath;
```

Kód 1 - Tvorba inštancie geometrickej cesty a definovanie jej figúry.

Pred jej priradením je však potrebné definovať tzv. figúru cesty `PathFigure`, ktorá je definovaná pomocou začiatočného bodu a následne kolekciou zvyšných bodov rezu. V tomto prípade bol ako počiatočný bod rezu vybraný prvý bod z kolekcie bodov, definujúcej rez objektu. Zvyšné body boli priradené do tzv. `PathSegmentCollection` pomocou segmentu čiary `PolyLineSegment`, ktorý definuje cestu rezu (Kód 2). Do tohto segmentu sa priradí kolekcia bodov rezu, ktorá je predávaná metóde ako jej parameter. Inštancia kolekcie segmentov sa následne predáva ako parameter pri tvorbe inštancie figúry cesty (Kód 1).

```
// vytvorenie segmentu čiary z bodov rezu
PolyLineSegment plSegment = new PolyLineSegment(clipPoints, false);
// vytvorenie zoznamu segmentov
PathSegmentCollection psCollection = new PathSegmentCollection();
// priradenie segmentu do zoznamu segmentov
psCollection.Add(plSegment);
```

Kód 2 - Definovanie segmentu čiary a zoznamu segmentov.

Takýmto spôsobom sa oreže toľko objektov, koľko je vytvorených rezov daného rozrezaného objektu. Spájanie objektov je následne jednoduché. Keďže je ako rezaný objekt použitý obdĺžnik, tak sa využíva pri spájaní jednotlivých častí (orezané obdĺžniky) ľavý horný bod obdĺžnika, ktorý je určený ako tzv. kontrolný bod. Keď je tento bod pre dve spájané časti rovnaký sú spojené správne. Samozrejme kontrola rovnosti bodov má nastavenú voľnosť 10 grafických bodov (CHECK_SENSIBILITY), čo zapríčiní samovoľné doskočenie obidvoch spájaných častí k sebe, ak sú ich kontrolné body v tejto vzdialenosti (Kód 3). Celý objekt je správne poskladaný, ak sa všetky kontrolné body jednotlivých častí rovnajú.

```
/* ak je presúvaný element (elem) dostatočné blízko pri kontrolnom elemente,
tak
   sa spoja */
if (MathUtility.IsPointInRing(elem.ControlPoint, element.ControlPoint,
       MathUtility.ValueFromPercentage(CHECK_SENSIBILITY,
       Math.Min(element.Size.Width, element.Size.Height))))
{
       // nastavenie novej pozície elementu
       elem.Position = new Point(elem.Position.X + (element.ControlPoint.X -
       elem.ControlPoint.X), elem.Position.Y + (element.ControlPoint.Y -
       elem.ControlPoint.Y));
       // nastavenie nového kontrolného bodu pre presúvaný element
       elem.ControlPoint = element.ControlPoint;
}
```

Kód 3 - Kontrola spojenia dvoch častí z metódy CheckPosition v triede WholeObject.

Objekt je teda na hracej ploche úlohy zobrazený toľkokrát, z koľkých častí je zložený, pričom z každého zobrazenia je pomocou clip-u viditeľná len plocha odpovedajúca príslušnému zvyšku po aplikovaní rezu.

6.2 Animácia stĺpcov grafu v rozšírení frameworku „Graf“

Pri tvorbe tohto rozšírenia boli použité animácie, ktoré vytvárajú efekt rastúcich stĺpcov grafu a zvyšujúcej sa percentuálnej hodnoty grafu. Ide o funkciu, ktorá má vo svojej podstate len grafický efekt, ale využitou neštandardnou schopnosťou použitej technológie, vytvára zaujímavé neštandardné riešenie v rámci implementácie aplikácie.

Stĺpec grafu je tvorený obdĺžnikom, ktorý je vyplnený gradientnou farbou pomocou triedy `LinearGradientBrush`. Tá pomocou triedy `GradientStop`, definuje jednotlivé farby, ktoré sú v gradiente postupne zastúpené. Inštancia triedy `GradientStop` definuje jednu farbu gradientu pomocou vlastnosti tzv. **Offset**. Ide o percentuálnu hodnotu od 0 po 1, upresňujúcu zastúpenie farby v celej škále gradientu (Kód 4). Ak má farba definovaný offset s hodnotou 0,5 a druhá farba obsahuje rovnakú hodnotu, prechod medzi farbami bude ostro zakončený v polovici celkovej dĺžky gradientu. Tento efekt je spôsobený tým, že obidve farby začínajú v začiatku gradientu, ale prvá farba skončí v jeho polovici, pričom druhá vyplní zvyšok gradientu. Ak sa ich hodnoty budú líšiť, gradient bude mať medzi farbami plynulý prechod. Plynulosť závisí na veľkosti rozdielu medzi hodnotami offsetov jednotlivých farieb.

```
LinearGradientBrush brush = new LinearGradientBrush();
// startovaci bod gradientnej farby
brush.StartPoint = new Point(0.5, 1);
// koncový bod farby
brush.EndPoint = new Point(0.5, 0);
// vytvorenie minimálnej farby a maximálnej farby
var minColor = Color.FromArgb(minAlpha, color.R, color.G, color.B);
var maxColor = Color.FromArgb((byte)(minAlpha +
       MathUtility.ValueFromPercentage(value, maxAlpha - minAlpha)),
       color.R, color.G, color.B);
// vytvorenie farieb gradientu pomocou gradientných stopov
stop1 = new GradientStop(minColor, 0);
stop2 = new GradientStop(maxColor, 0);
stop3 = new GradientStop(Colors.Transparent, 0);
```

Kód 4 - Vytvorenie gradientnej farby pomocou definície jednotlivých farieb gradientu.

Pri tvorbe gradientu je potrebne definovať počiatočný (StartPoint) a koncový (EndPoint) bod jeho vykreslenia. Definuje sa pomocou percentuálnych hodnôt. Keďže gradient musí začínať v strede základne stĺpca grafu, je hodnota x-ovej súradnice štartovacieho bodu nastavená na 0,5. Y-ová súradnica tohto bodu je nastavená na hodnotu 1, pretože chceme aby sa gradient vykresľoval od spodnej časti obdĺžnika (súradnicová sústava plátna, na ktorom je graf vykreslený ide z ľava do prava pre x-ovú os a z vrchu na dol pre y-ovú os). Koncový bod gradientu má definovanú x-ovú súradnicu taktiež na pozícii 0,5 a y-ovú na 0, kdeže má končiť v hornej časti stĺpca grafu (Kód 4).

Gradient obsahuje 3 farby, pričom posledná farba je priesvitná. Tá slúži len z dôvodu doplnenia gradientu, aby sa farba grafu zastavila na definovanej hodnote. Konkrétne farby grafu sa vypočítavajú na základe percentuálnej hodnoty, pomocou ktorej sa vypočíta najsvetlejšia a najtmavšia farba gradientu (Kód 4). Keďže gradient vytvára postupný prechod medzi týmito farbami, nebolo potrebné medzi ne definovať doplňujúce prechodové farby. Nakoniec sa definované farby (`GradientStop`) priradia do inštancie štetca (Kód 5).

```
// pridanie farieb do gradientu
brush.GradientStops.Add(stop1);
brush.GradientStops.Add(stop2);
brush.GradientStops.Add(stop3);
```

Kód 5 - Priradenie farieb gradientu.

```
// registrácia farieb gradientu k menu
this.RegisterName("GradientStop1", stop1);
this.RegisterName("GradientStop2", stop2);
this.RegisterName("GradientStop3", stop3);
```

Kód 6 - Registrácia mien farieb gradientu.

Animácia jednotlivých stĺpcov je vytvorená pomocou triedy `DoubleAnimation`, ktorá zabezpečuje automatický nárast hodnôt typu `double`, po zadaní minimálnej a maximálnej hodnoty a typu priebehu. Pomocou tejto animácie sa mení hodnota offsetu jednotlivých farieb gradientu. Pri každej je minimálna hodnota nastavená na 0, pričom animácia maximálnej a transparentnej farby má maximálnu hodnotu animácie nastavenú na príslušnú percentuálnu hodnotu, akú má daný stĺpec grafu reprezentovať. Dobu po akú má animácia meniť premennú offsetu, je definovaná pomocou vlastnosti **Duration**. Následne je potrebné priradiť animáciu farbe gradientu, tá sa priraďuje pomocou metódy **SetTargetName**, do ktorej sa pridelí inštancia

animácie a zaregistrované meno inštancie farby gradientu (Kód 6). Posledným krokom je definovanie, ktorá premenná farby gradientu sa má meniť, pomocou metódy **SetTargetProperty**, v ktorej je priradená premenná vlastnosti Offset (Kód 7).

```
// definícia animácie pre farbu gradientu
DoubleAnimation offsetAnimation2 = new DoubleAnimation();
// nastavenie minimálnej hodnoty animácie
offsetAnimation2.From = 0.0;
// nastavenie maximálnej hodnoty animácie
offsetAnimation2.To = value;
// nastavenie dĺžky trvania animácie
offsetAnimation2.Duration = TimeSpan.FromSeconds(seconds);
// priradenie animácie k farbe gradientu podľa jej mena
Storyboard.SetTargetName(offsetAnimation2, "GradientStop2");
// nastavenie cieľovej premennej farby gradientu, ktorá sa má meniť
Storyboard.SetTargetProperty(offsetAnimation2,
    new PropertyPath(GradientStop.OffsetProperty));
```

Kód 7 - Definícia animácie farby.

Aby mohla byť animácia prevedená, je potrebné použiť triedu `Storyboard`, ktorá všetky v nej zaregistrované animácie dokáže spustiť alebo zastaviť (Kód 8).

```
// priradenie animácii do 'Storyboard-u'
ColorAnimation.Children.Add(offsetAnimation1);
ColorAnimation.Children.Add(offsetAnimation2);
ColorAnimation.Children.Add(offsetAnimation3);
```

Kód 8 - Pridanie animácií do inštancie triedy Storyboard.

Okrem animácie stĺpca grafu bolo vytvorené aj automatické zväčšovanie percentuálnej hodnoty grafu. Táto vlastnosť textovej hodnoty čísla, je taktiež vytvorená pomocou animácie. Na rozdiel od stĺpca grafu, bola použitá trieda `Int32Animation`, ktorá dokáže automaticky meniť hodnotu typu `int`. Nastavenie inštancie tejto triedy je rovnaké ako to bolo pri použití triedy `DoubleAnimation`.

```
// definícia animácie pre hodnotu kolónky grafu
Int32Animation valueAnimation = new Int32Animation();
valueAnimation.From = 0;
valueAnimation.To = value;
valueAnimation.Duration = TimeSpan.FromSeconds(seconds);
Storyboard.SetTargetName(valueAnimation, "Value");
Storyboard.SetTargetProperty(valueAnimation,
       new PropertyPath(GraphColumnValue.IntValueProperty));
```

Kód 9 - Definícia animácie textovej hodnoty.

Aby bolo možné vytvoriť animáciu textu s číselnou hodnotou, bolo potrebné vytvoriť vlastnú triedu, ktorá by to zabezpečovala. Preto bola vytvorená trieda `GraphColumnValue`, ktorá dedí vlastnosti z triedy `DependencyObject`. Do tejto triedy je potrebné zaregistrovať textový blok, ktorý zobrazuje danú hodnotu.

```
// vytvorenie hodnoty grafu s registráciou inštancie textového bloku „text“.
GraphColumnValue gValue = new GraphColumnValue(text);
// pridanie textovej prípony
gValue.Suffix = "%";
```

Obsahuje definovanú tzv. `DependencyProperty`, konkrétne `IntValueProperty` respektíve `DoubleValueProperty` (Kód 10), ktorá sa používa pri registrácii cieľovej premennej v animácii (Kód 9). Pri jej vytváraní je potrebné zadať jej meno, typ, typ triedy, v ktorej je definovaná a udalosť, ktorá bude automaticky spúšťaná pri zmene hodnoty tejto premennej.

```
public static readonly DependencyProperty IntValueProperty =
       DependencyProperty.Register("IntValue", typeof(Int32),
       typeof(GraphColumnValue),
       new FrameworkPropertyMetadata(OnValuePropertyChanged));
```

Kód 10 - Definícia tzv. dependecy premennej.

Animácia vždy pri zmene tejto hodnoty spúšťa udalosť `OnValuePropertyChanged`, ktorá automaticky prepíše vlastnosť triedy `IntValue` respektíve `DoubleValue`, ktoré menia text registrovaného textového bloky zobrazujúceho hodnotu v stĺpci grafu (Kód 11).

```
// vlastnosť pre typ Int
public int IntValue {
       get { return (Int32)GetValue(IntValueProperty); }
       set {
              // nastavenie hodnoty pre dependency premennú
              SetValue(IntValueProperty, value);
              // zmena textu textového bloku
              textBlock.Text = prefix + value + suffix; }}

// udalosť, ktorá sa spúšťa pri zmene Dependency premenných
private static void OnValuePropertyChanged(DependencyObject source,
       DependencyPropertyChangedEventArgs e){
       GraphColumnValue control = source as GraphColumnValue;
       // nastavenie novej hodnoty pre typ integer a double
       if (e.Property.Name == IntValueProperty.Name)
              control.IntValue = (Int32)e.NewValue;
       if (e.Property.Name == DoubleValueProperty.Name)
              control.DoubleValue = (Double)e.NewValue; }
```

Kód 11 - Udalosť automatickej zmeny textovej hodnoty s definíciou vlastnosti IntValue.

Inštanciu takto vytvorenej triedy je nakoniec potrebné zaregistrovať pod meno, ktoré je následne priradené animácii (Kód 9), ako cieľové meno inštancie triedy, ktorá má byť animáciou menená.

```
// registrácia hodnoty grafu k menu
this.RegisterName("Value", gValue);
```

Vytvorená animácia je nakoniec pridaná do vlastnej inštancie triedy `Storyboard`, ktorá zabezpečí jej chod.

```
// pridanie animácie do 'Storyboard-u'
ValueAnimation.Children.Add(valueAnimation);
```

7 Dokumentácia

Nasledujúca kapitola sa bude venovať popisu implementovaných úloh a rozšírení použitého frameworku z pohľadu užívateľa. Popis bude obsahovať informácie o ovládaní jednotlivých úloh, ich tvorbe a konfigurácii. Z hľadiska ovládania a konfigurácie celej implementovanej aplikácie je potrebné rozlišovať dva typy užívateľov a to lekára a pacienta. Lekár je osoba, ktorá je zodpovedná za nastavenie a tvorbu úloh podľa potreby pacienta, kdežto pacient sa zameriava len na riešenie úloh. Preto bude popis dokumentácie jednotlivých úloh rozlišovaný z pohľadu oboch typov užívateľov.

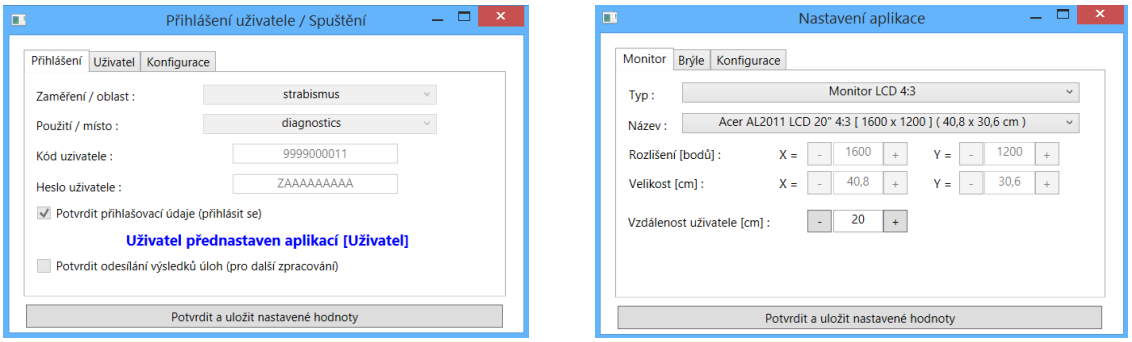
7.1 Spoločné súčasti frameworku

Užívateľské rozhranie hlavného menu sa skladá z 3 časti (Obrázok 7.1). Na ľavej strane je umiestnené menu úloh, ktoré slúži na prístup k jednotlivým skupinám úloh, alebo modulom určením na zobrazenie spracovaných dát (Hodnotení), úvodnému popisu aplikácie a kontaktom na jej tvorcov. V hornej časti sa nachádza hlavné menu slúžiace na prístup k spoločným základným nastaveniam aplikácie. Strednú časť tvorí aplikačné plátno, kde sa zobrazujú informácie o aplikácii alebo odkazy na spustenie jednotlivých úloh a samozrejme tiež samotné úlohy.

Obrázok 7.1 - Hlavné menu aplikácie.

Hlavné menu aplikácie je tvorené jej názvom a panelom tlačítok určených na počiatočné nastavenie celej aplikácie. **„Nastavení užívatele“** slúži na vloženie unikátneho identifikačného čísla pacienta spolu s jeho prístupovým heslom. Po kliknutí na toto tlačítko sa zobrazí dialógové

okno nastavení (Obrázok 7.2). Toto nastavenie identifikuje pacienta, aby mohli byť namerané dáta z úloh pridelené k tomuto identifikačnému číslu. Dialóg obsahuje ďalšie položky podľa použitých typov úloh.



Obrázok 7.2 - Dialógové okno nastavenia užívateľa (vľavo) a nastavenia parametra monitorov (vpravo).

Po otvorení nastavenia **„Další konfigurace“** sa sprístupní dialógové okno, kde je možné tieto nastavenia vykonať (Obrázok 7.2). Pri potrebe špecifického nastavenia monitora, je možné meniť parametre, ako typ monitora, názov monitora (ide o presný názov používaného monitora), jeho rozlíšenie v pixloch, veľkosť v centimetroch a vzdialenosť užívateľa od monitora.

Menu úloh je určené na prístup k úlohám na základe skupín. Užívateľ si vyberie skupinu úloh, na akú chce diagnostiku alebo precvičovanie zamerať. Po kliknutí na príslušné tlačítko sa zobrazia na aplikačnom plátne odkazy na spustenie jednotlivých úloh. Z nich si následne užívateľ vyberie úlohu a kliknutím na príslušné tlačítko úlohu spustí.

Po jej spustení sa zobrazí užívateľovi hracie plátno úlohy s ovládacou a informačnou lištou. Ovládacia lišta sa nachádza v podstate nad hracím plátnom úlohy. Tento panel obsahuje základné (**„Spustit“, „Zastavit“, „Celá obr.“ a „Zpět“**) (Obrázok 7.3), ale aj špecifické tlačítka pre danú úlohu. Základné tlačítka sa nachádzajú v každej implementovanej úlohe a spĺňajú rovnakú funkčnosť. Pomocou tlačítok **„Spustit“** a **„Zastavit“** užívateľ spúšťa respektíve zastavuje beh úlohy. **„Celá obr.“** slúži na roztiahnutie okna aplikácie na celú plochu monitora. Tlačítko **„Zpět“** slúži na návrt do hlavného menu aplikácie, čiže ide o ukončenie úlohy. Špecifické tlačítka sú pre každú úlohu iné, ich funkcionalita sa líši, preto bude popísaná konkrétne pri daných úlohách. Zväčša sú určené na tvorbu rôznych konfigurácií úlohy.

Obrázok 7.3 - Základné ovládacie tlačítka úloh.

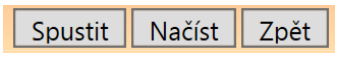
7.2 Úloha „Rozdiely“

Užívateľské rozhranie v tejto úlohe je možné rozdeliť na dve pomyselne časti, ovládaciu a konfiguračnú. Ich rozdielnosť spočíva hlavne v tom, pre akého užívateľa sú stavané. Ovládanie úlohy (spustenie, zastavenie, ukončenie) je určené zväčša pacientom, ktorý si sami ovládajú jej beh, pričom konfigurácia je určená hlavne dozornej osobe, čiže lekárovi, ktorý je zodpovedný za jej nastavenie, tak aby vyhovovala potrebám diagnostiky alebo rehabilitácie. Samozrejme konfiguráciu môže previesť aj pacient podľa pokynov lekára, z toho dôvodu sú všetky tlačítka, či už ovládacie alebo konfiguračné zatiaľ prístupné nepretržite.

Konfigurácia úlohy

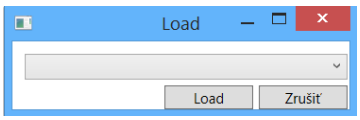
Konfigurácia úlohy spočíva vo vytvorení alebo úprave obrázka, v ktorom neskôr pacient hľadá rozdiely. Ide o úpravy spojené s rozmiestnením jednotlivých čiastkových obrázkov, ich zmenu veľkosti, nastavenie oblasti generovania pozície jednotlivých obrázkov a označenie statických objektov.

Tlačítka určené na konfiguráciu úlohy sú aktívne len vtedy, keď je úloha zastavená. Nie je preto možné meniť nastavenia počas jej behu. Po spustení úlohy sa užívateľovi zobrazia na ovládacej lište tlačítka **„Spustit“**, **„Načíst“** a **„Zpět“** (Obrázok 7.4).



Obrázok 7.4 - Základné tlačítka úlohy po jej spustení.

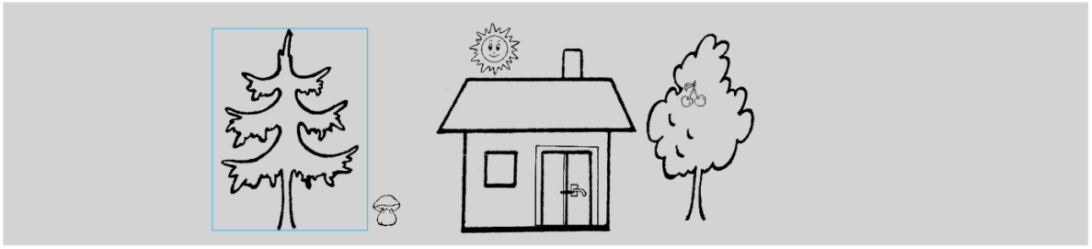
Dve z tlačítok slúžia na spustenie behu **(„Spustit“**) a ukončenie **(„Zpět“**) úlohy. Tretie menované tlačítko **(„Načíst“**) slúži na načítanie niektorej z konfigurácii úlohy pomocou dialógového okna, kde je možné vybrať z rolovacieho okienka konkrétnu konfiguráciu úlohy (Obrázok 7.5). Úloha sa následne prepne do tzv. dizajnovacieho módu, kde je možné meniť a upravovať načítanú konfiguráciu.



Obrázok 7.5 - Dialógové okno slúžiace na načítanie konfigurácie úlohy.

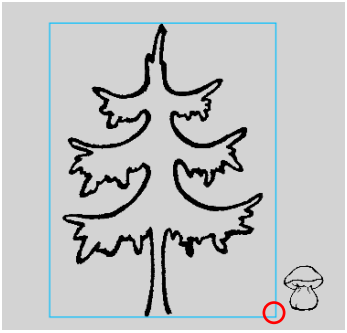
Po jej načítaní sa zobrazí na hornom hracom plátne obrázok, ktorý táto konfigurácia predstavuje. Je zložení z čiastkových menších obrázkov (objektov), s ktorými je možné pohybovať a meniť ich veľkosť. Upravovaný objekt je označený bledomodrým rámčekom (Obrázok 7.6).

Pre posunutie nejakého objektu, je potrebné podržať objekt ľavým tlačítkom myši (v prípade tabletu len prstom). Počas toho je možné pohybom myši po vrchnom hracom plátne (šedá oblasť) meniť pozíciu objektu. Po presunutí objektu na novú pozíciu, stačí uvoľniť ľavé tlačítko myši (v prípade tabletu zodvihnúť prst z obrazovky).



Obrázok 7.6 - Hracie plátno s objektmi (menšie obrázky) reprezentujúcimi celý obrázok (konfiguráciu) úlohy.

V prípade zmeny veľkosti niektorého z objektov, je potrebné tento objekt označiť kliknutím, pričom sa zobrazí okolo neho slabo modrý rámček, ktorý okrem úlohy označenia vybraného objektu, znázorňuje okraje tohto objektu, čiže reprezentuje jeho veľkosť. Pre zmenu veľkosti je ho potrebné držať v ľavom dolnom rohu rámčeka (Obrázok 7.7). Následne pohybom myšky (v prípade tabletu prstom) meniť jeho veľkosť.

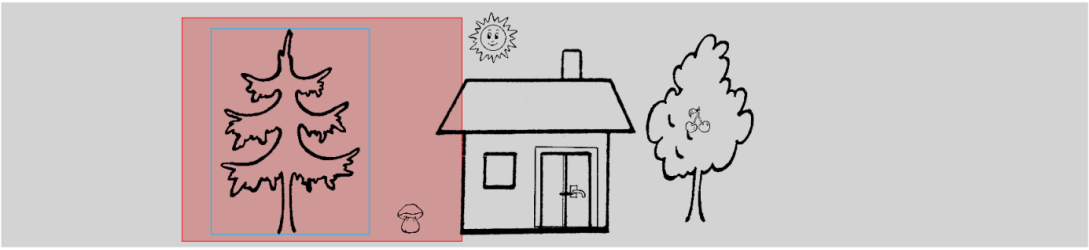


Obrázok 7.7 - Miesto určené na zmenu veľkosti objektu (červený krúžok).

Ďalšiu funkcionalitu, ktorú je možné na objektoch prevádzať je nastavenie oblasti, v ktorej sa budú rozdielne objekty náhodne zobrazovať. Pre označenie tejto oblasti je potrebné vybrať konkrétny objekt, ktorému túto oblasť chceme nastaviť a následne kliknúť na tlačítko **„Označit oblast“**, ktoré sa nachádza na ovládacej lište úlohy (Obrázok 7.8).

Obrázok 7.8 - Ovládacia lišta konfiguračného módu s aktívnym označovacím módom (zvýraznené tlačítko).

Po tomto kliknutí sa aktivuje mód (pokým je mód aktívny, je tlačítko zvýraznené), v ktorom je možné túto oblasť označiť kliknutím na ľubovoľné miesto vrchného hracieho plátna (vrchná šedá oblasť) a ťahaním myšky (v prípade tabletu prstom). Počas ťahu je potrebné neustále držať tlačítko myši aktívne. Označovanie je sprevádzané tvorením červeného obdĺžnika, ktorý reprezentuje presnú označovanú oblasť (Obrázok 7.9).

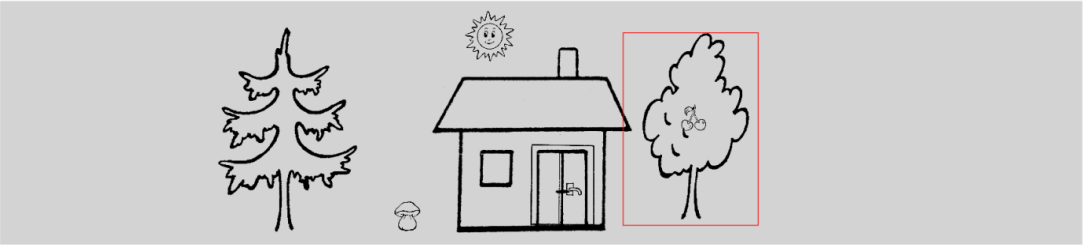


Obrázok 7.9 - Zvýraznená oblasť (červený štvorec), ukazujúca užívateľovi oblasť, ktorú už označil.

Po uvoľnení myšky (v prípade tabletu uvoľnení prsta) sa označená oblasť zmení na zelenú (Obrázok 7.10). Je potrebné dávať pozor na veľkosť tejto označovanej oblasti. Musí byť väčšia, než je veľkosť objektu, ktorému je táto oblasť definovaná. Pri zmene polohy objektu, ktorý má túto oblasť definovanú, sa automaticky táto oblasť zruší a je ju potrebné vytvoriť na novo.

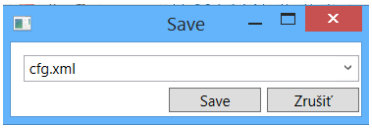
Obrázok 7.10 - Zelená oblasť reprezentujúca miesto, kde sa bude daný objekt náhodne generovať.

Dôležitou súčasťou tejto konfigurácie je možnosť určiť objekt ako statický. Takýto objekt pri generovaní úlohy, nebude nikdy odstránení. Pre takéto definovanie je potrebné kliknutím označiť objekt a následne kliknúť na tlačítko **„Označit jako statické“** (Obrázok 7.8). Objekt sa automatický zvýrazní červeným rámčekom, ktorý indikuje, že je objekt statický (Obrázok 7.11).



Obrázok 7.11 - Statický objekt konfigurácie (v červenom rámčeku).

Po dokončení je konfiguráciu potrebné úložiť. Tlačítkom **„Uložit“** sa otvorí dialógové okno, kde je možné z rolovacieho okienka buď vybrať už existujúcu konfiguráciu, ktorá sa prepíše, alebo zadefinovať novú napísaním názvu konfigurácie do tohto okienka. Následne je potrebné potvrdiť uloženie kliknutím na tlačítko uložiť (Obrázok 7.12). Automaticky sa konfiguračný mód prepne do hracieho.

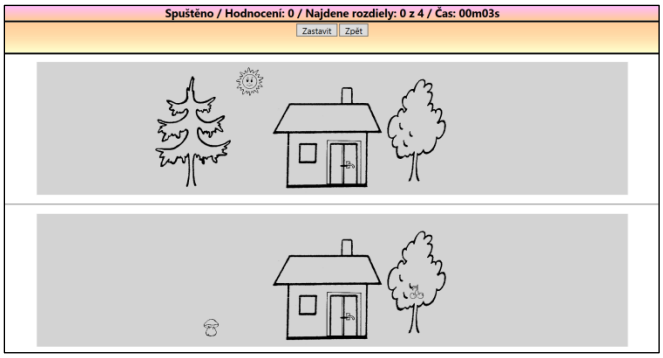


Obrázok 7.12 - Dialógové okno pre uloženie vytvorenej konfigurácie.

V prípade skoršieho ukončenia konfiguračného módu bez ukladania, je možné tak učiniť kliknutím na tlačítko **„Ukončit“**. To vráti stav úlohy do hracieho módu.

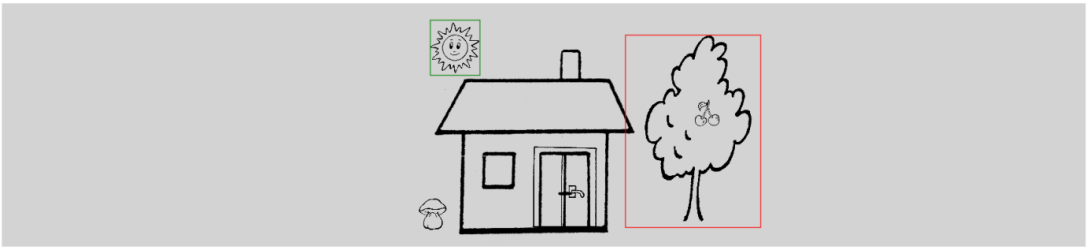
Ovládanie úlohy

Ako to už bolo popísané vyššie, po otvorení úlohy sa zobrazí pred užívateľom hracia plocha rozdelená na vrchné a spodné hracie plátno. V nich sa budú generovať obrázky, ktoré bude užívateľ porovnávať. Ovládacia lišta obsahuje len základné tlačítka (Obrázok 7.4). Pre spustenie úlohy je určené tlačítko **„Spustit“** (Obrázok 7.4). Po spustení sa náhodne vyberie jedna z možných konfigurácií a na základe nej sa vygeneruje na oboch plátnach ten istý obrázok s tým rozdielom, že oba budú obsahovať náhodný počet vymazaných objektov (Obrázok 7.13).



Obrázok 7.13 - Úloha po spustení s vygenerovanými obrázkami na oboch hracích plátnach.

Užívateľovou úlohou je následne nájsť rozdiely medzi týmito dvoma obrázkami. Výber objektov, ktoré sa nachádzajú na plátnach, sa prevádza kliknutím na tieto objekty. Ak ide o objekt, v ktorom sa obrázky líšia, ostane označený zeleným rámčekom (Obrázok 7.14). Ak užívateľ klikne na nesprávny objekt, ten sa po dobu kliku zvýrazní červeným rámčekom indikujúcim nesprávny výber (Obrázok 7.14).



Obrázok 7.14 - Hracia plocha s označeným správnym (zelený rámček) a nesprávnym (červený rámček) výberom objektu.

Po nájdení všetkých rozdielov sa automaticky úloha ukončí a načíta sa nové zadanie, ktoré je už spustené, čiže užívateľ môže hneď začať hľadať rozdiely. Ak už neexistuje žiadna ďalšia konfigurácia, ktorá ešte nebola spustená, úloha sa zastaví a je ju potrebné pustiť od začiatku. Pre zastavenie úlohy počas jej behu slúži tlačítko **„Zastavit“**, ktoré je viditeľné len keď je úloha spustená (Obrázok 7.15).

Obrázok 7.15 - Ovládacia lišta počas behu úlohy.

7.3 Úloha „Rôzne obrázky“

Úloha ponúka dva typy prístupu, na jednej strane je to ovládanie úlohy a na druhej konfigurácia úlohy. Obidve časti sú prístupné formou tlačítok na ovládacej lište úlohy, pričom konfigurácia sa prevádza aj externou tvorbou adresára s obrázkami, ktoré budú v úlohe použité.

Konfigurácia úlohy

Konfiguráciou úlohy sa zabezpečuje nie len to ako budú obrázky v úlohe generované, ale aj to aké obrázky bude úloha obsahovať. Preto sa delí konfigurácia na dve formy, externú a internú. Pod pojmom externej konfigurácie sa rozumie zadefinovanie obrázkov v konfiguračnom adresári úlohy. Umiestnenie tohto adresára je špecifikované až po inštalácii aplikácie na počítač užívateľa. Do tohto adresára sa nahrávajú tie obrázky, ktoré sa majú v úlohe zobrazovať. Pri tomto nahrávaní obrázkov je potrebné presne stanoviť názov každého obrázka, ktorý má nasledujúcu štruktúru:

0001-A,Mir,Rot.png

1. **Zelene označená časť názvu** – Ide o identifikačné číslo obrázka. Jeho úlohou je presná identifikácia jedného obrázka, alebo skupiny zloženej z dvoch obrázkov. Táto skupina dvoch obrázkov musí byť označená rovnakým identifikačným číslom.
2. **Červená označená časť názvu** – Úlohou tejto časti je definovanie typu obrázka a jeho možností pri zobrazení.

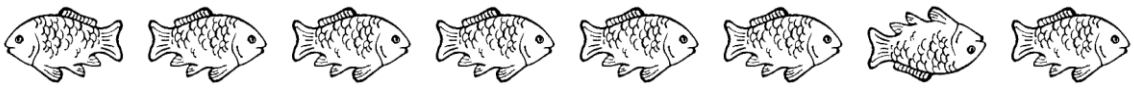
   Typ obrázka slúži na definovanie, či ide o obrázok, ktorý sa bude v úlohe vyskytovať častejšie (typ **A**) alebo o obrázok, ktorého výskyt bude menší (typ **B**) (pripomienka: úlohou užívateľa je nájsť z pomedzi skupiny obrázkov také rozdielne obrázky, ktorých výskyt v tejto skupine je menší). Typ je povinným parametrom obrázka iba v prípade, že je obrázok spojený s iným obrázkom ako konfiguračná dvojica (majú rovnaké identifikačné číslo), inak sa v názve nesmie objaviť. Ďalšie parametre v tejto časti názvu obrázka sú určené na definíciu rotácie a zrkadlového natočenia obrázka (Mir, Rot...atď.). Tabuľka 1 popisuje všetky parametre, ktoré môžu byť v názve použité.

Tabuľka 1 - Popis konfiguračných parametrov, ktoré je možné použiť pri definícii názvu obrázka.

Parameter	Popis parametru
A	Určuje typ obrázka, ktorý z konfiguračnej dvojice obrázkov bude v úlohe vygenerovaný vo väčšom počte.
B	Určuje typ obrázka, ktorý z konfiguračnej dvojice obrázkov bude v úlohe vygenerovaný v menšom počte. Čiže ide o obrázok, ktorý bude rozdielový.
Rot	Definuje náhodnú rotáciu obrázka. Pri generovaní obrázka v úlohe, bude náhodne rotovaný.
Rot90	Definuje rotáciu obrázka o 90°.
Rot180	Definuje rotáciu obrázka o 180°.
Rot270	Definuje rotáciu obrázka o 270°.
Mir	Definuje náhodné zrkadlové otočenie obrázka, buď podľa vertikálnej osy, horizontálnej osy, alebo oboch naraz.
MirV	Definuje zrkadlové otočenie obrázka podľa vertikálnej osy.
MirH	Definuje zrkadlové otočenie obrázka podľa horizontálnej osy.
MirHV	Definuje zrkadlové otočenie obrázka podľa vertikálnej a horizontálnej osy naraz.

Príklad konfigurácie použitím jedného obrázka

Užívateľ chce použiť jeden obrázok, ako jednu konfiguráciu, ktorá bude tvorená jedným typom obrázka, pričom hľadaný rozdiel v skupine obrázkov úlohy bude spočívať len v rotácii alebo zrkadlovom otočení niektorých obrázkov zo skupiny (Obrázok 7.16).

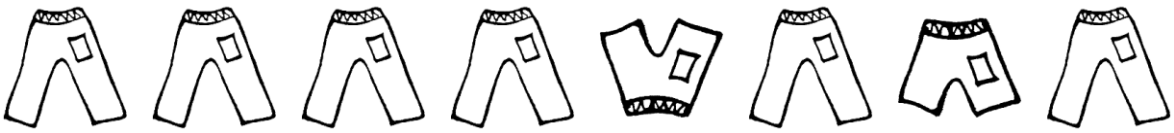


Obrázok 7.16 - Príklad použitia jedného obrázka, pri definícii konfigúrácie po jej zobrazení v úlohe.

Pri takejto definícii názvu tohto obrázka bude vyzerať jeho názov nasledovne: **0001-Mir.png**. Identifikačné číslo 0001 je možné zvoliť ľubovoľné. V konfiguračnej časti názvu je možné vidieť len parameter **Mir**, ktorý zabezpečí náhodné zrkadlové otočenie obrázka. Keďže ide o konfiguráciu, ktorá sa skladá len z jedného obrázka, jeho typ sa v názve nesmie nachádzať. Samozrejme názov môže obsahovať aj iné parametre ako napríklad Rot, Rot90, MirV a iné. Počet parametrov nie je obmedzený.

Príklad konfigurácie použitím dvoch rozdielnych obrázkov

Užívateľ chce definovať konfiguráciu skupiny dvoch odlišných obrázkov, z ktorých jeden bude generovaný v skupine obrázkov úlohy vo väčšom počte a druhý v menšom (Obrázok 7.17).



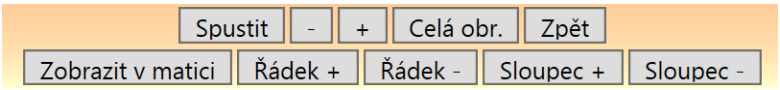
Obrázok 7.17 - Príklad definície konfigurácie použitím dvoch rozdielnych obrázkov, po jej zobrazení v úlohe.

Pri tejto definícii bude názov dvoch obrázkov, ktoré majú byť spojené v konfiguračnej dvojici, vyzerať nasledovne:

1. **0002-A.png** (prvý obrázok)
2. **0002-B,MirH.png** (druhý obrázok)

Ako je možné vidieť z názvov obrázkov, prvý má definovaný **typ A**, pričom neobsahuje žiadny iný konfiguračný parameter. Je to z tohto dôvodu, že pri tejto konfigurácii má byť obrázok typu A zobrazený v základnej forme, bez použitia rotácie alebo zrkadlového otočenia. Druhý obrázok má definovaný **typ B**, čiže pôjde o obrázok, ktorý bude generovaný v menšom počte. Má definovaný jeden parameter a to zrkadlové otočenie podľa horizontálnej osy (**MirH**). Čo je v konečnom dôsledku možné vidieť aj na obrázku (Obrázok 7.17). Identifikačné číslo oboch obrázkov je rovnaké, čo identifikuje obrázky ako konfiguračnú dvojicu. Taktiež aj v tomto prípade je možné, aby obrázky obsahovali konfiguračné parametre rotácie alebo zrkadlového otočenie bez ohľadu na ich typ. Všetko záleží len na tom ako majú byť obrázky v úlohe vygenerované.

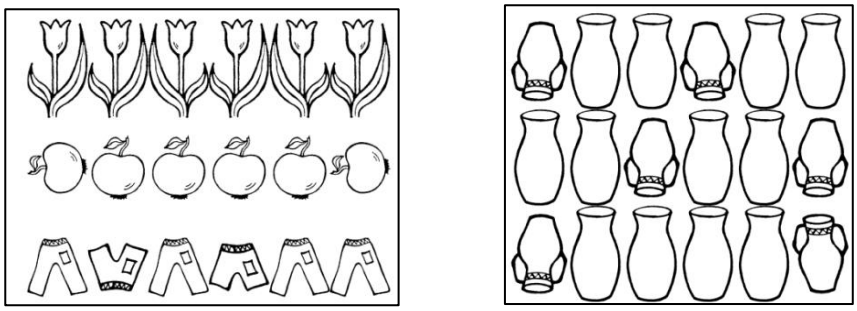
Interná konfigurácia úlohy spočíva v možnosti nastavenia zobrazenia obrázkov priamo v úlohe pomocou tlačítiek na ovládacej lište (Obrázok 7.18).



Obrázok 7.18 - Ovládacia lišta úlohy "Rôzne obrázky".

Obrázky sú na hracom plátne úlohy zobrazované v matici, pričom je možné meniť počet riadkov alebo stĺpcov pomocou tlačítiek **„Řadek +“**, **„Řádek -“**, **„Sloupec +“** a **„Sloupec -“**. Ich funkcia

presne zodpovedá ich názvu. V základnom nastavení úloha zobrazuje na plátne väčší počet rôznych konfigurácií (skupín obrázkov) v riadkoch (Obrázok 7.19).



Obrázok 7.19 - Zobrazenie rôznych skupín obrázkov v jednotlivých riadkoch (vľavo) a zobrazenie jednej skupiny obrázkov v maticovej forme (vpravo).

Je však možné zobraziť na celej ploche iba jednu konfiguráciu (jednu skupinu obrázkov) v maticovej forme a to pomocou tlačítka **„Zobrazit v matici“** (Obrázok 7.19).
Poslednými konfiguračnými tlačítkami sú **„+“** a **„-“**. Ich úlohou je možnosť vytvorenia väčších alebo menších rozstupov medzi jednotlivými obrázkami (Obrázok 5.11).

Ovládanie úlohy

Po otvorení úlohy sa zobrazia na hracej ploche skupiny obrázkov v riadkoch, alebo po zmene nastavenia v matici (tlačítko **„Zobrazit v matici“**). Z nich užívateľ hľadá také obrázky, ktoré sa v danej skupine líšia od väčšiny. Výber týchto obrázkov sa prevádza pomocou kliku myši (v prípade tabletou dotykom prsta) na jednotlivé obrázky. Keď bol vybraný správny obrázok, zobrazí sa pred ním zelená fajka, ak bol vybraný nesprávny obrázok, zobrazí sa po dobu kliku červený kríž (Obrázok 7.20).

Obrázok 7.20 - Ukážka správneho a nesprávneho výberu v úlohe "Rôzne obrázky".

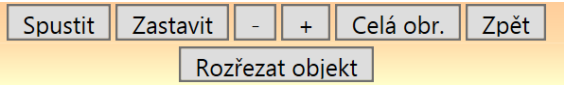
Pre spustenie úlohy slúži tlačítko **„Spustit“**, ktoré je prístupné len keď je úloha zastavená. Pre jej zastavenie je potrebné použiť tlačítko **„Zastavit“**, ktoré je viditeľné len za behu úlohy (Obrázok 7.3).

7.4 Úloha „Spájanie“

Keďže ide o úlohu, v ktorej je cieľom spájať rezy objektu, ponúka úloha okrem samotného užívateľské rozhrania pre jej riešenie, aj rozhranie na tvorbu týchto rezov objektu. Z tohto dôvodu je taktiež aj túto úlohu možné rozdeliť na konfiguračnú a ovládaciu časť.

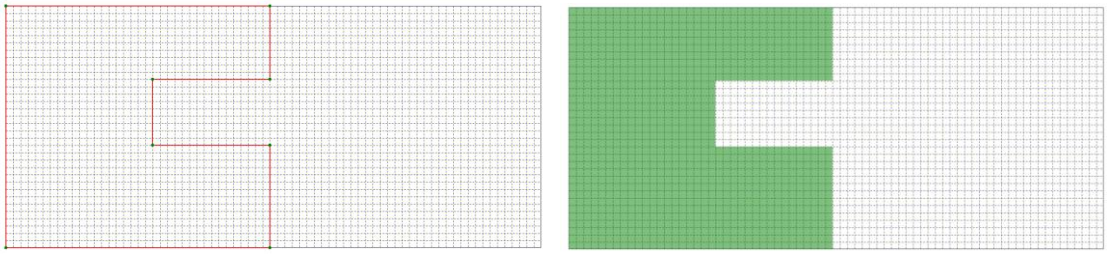
Konfigurácia úlohy

Po otvorení úlohy sa užívateľovi zobrazí hracie plátno, na ktorom sú rozmiestnené rozrezané objekty spolu s ovládacou lištou, kde je možné nájsť tlačítko s názvom **„Rozřezat objekt“** (Obrázok 7.21). Toto tlačítko slúži na prepínanie medzi užívateľským rozhraním s vygenerovanou úlohou a konfiguračným užívateľským rozhraním, kde môže užívateľ rozrezať objekt a tak vytvoriť novú konfiguráciu úlohy.



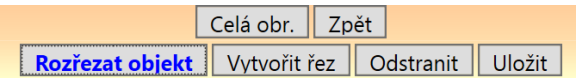
Obrázok 7.21 - Ovládacia lišta úlohy "Spájanie".

Po prepnutí na konfiguračné užívateľské rozhranie sa pred užívateľom zobrazí rezacia mriežka, ktorá predstavuje rozrezávaný objekt (obdĺžnik). Pomocou označovania bodov na mriežke sa postupne vytvorí tvar rezu (Obrázok 7.22). Počiatočný a koncový bod rezu musia byť na tom istom mieste mriežky. V prípade zlého označenia bodu je ho možné odstrániť opätovným kliknutím na tento bod.



Obrázok 7.22 - Ukážka tvorby rezu označovaním bodov (vľavo) a vytvoreného rezu konfigurácie (vpravo).

Aby bol rez hotový je potrebné kliknúť na tlačítko **„Vytvořit řez“**, ktoré daný rez vytvorí na základe označených bodov (Obrázok 7.23). Vytvorený rez sa automaticky zafarbí na zeleno (Obrázok 7.22).



Obrázok 7.23 - Ovládacia lišta konfiguračného rozhrania úlohy "Spájanie".

Takýmto spôsobom je potrebné vytvoriť všetky zvyšné rezy objektu. V prípade, že by bolo potrebné rez z nejakého dôvodu vymazať, jednoduchým kliknutím na vytvorený rez sa označí modrou farbou a pomocou tlačítka **„Odstranit“** sa vymaže (Obrázok 7.23). Po rozrezaní celého objektu je potrebné túto konfiguráciu uložiť pomocou tlačítka **„Uložit“** (Obrázok 7.23). Po uložení sa automaticky prepne konfiguračné rozhranie späť na rozhranie úlohy.

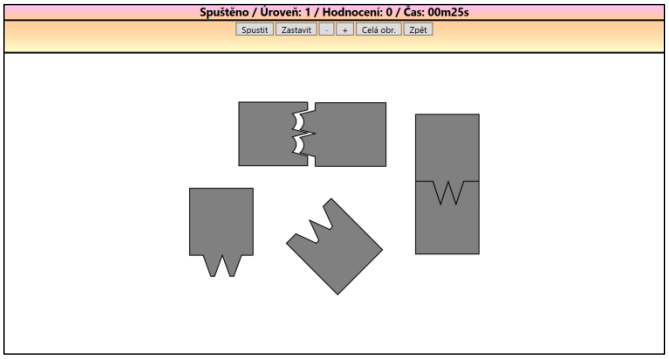
Okrem konfiguračného rozhrania, obsahuje úloha konfiguračné tlačítka **„Obarvit objekt“, „Obarvit řez“**, **„+“** a **„-“**, ktoré sú prístupné z užívateľského rozhrania úlohy. Tlačítko **„Obravit objekt“** zafarbí elementy jedného objektu rovnakou farbou, pričom každý objekt bude mať inú farbu. Na druhej strane **„Obarvit řez“** zafarbí každý element na hracom plátne inou farbou (Obrázok 5.17). Zoznam farieb, z ktorého sa jednotlivé farby vyberajú je definovaný v konfiguračnom súbore úlohy **„Colors.txt“**. V ňom je možné definovať farby jednoduchým pridaním riadka začínajúceho s písmenom **c**, za ktorým nasledujú farby zadávané vo forme RGB oddeľované bodkočiarkou. Počet farieb v riadku, tak ako aj počet riadkov s definovanými farbami je ľubovoľný.
Príklad správne definovaného riadka farby: **c 125 24 243; 0 0 125; 11 87 255**

Poslednými konfiguračnými tlačítkami sú **„+“** a **„-“**, pomocou ktorých je možné zväčšovať alebo zmenšovať vykreslené elementy na hracom plátne.

Ovládanie úlohy

Tak ako pri všetkých úlohách aj táto úloha sa spúšťa pomocou tlačítka **„Spustit“**, ktoré sa nachádza na ovládacej lište úlohy a opačne zastavuje pomocou tlačítka **„Zastavit“** (Obrázok 7.21). Úlohou užívateľa je pospájať všetky rozrezané elementy na hracom plátne do ucelených objektov. Elementy sa spájajú pomocou presunu jednotlivých častí k sebe pomocou kliku myšky a následným pohybom po plátne (v prípade tabletov pomocou prsta). Počas pohybu je potrebné neustále držať ľavé tlačítko myši aktívne. Jednotlivé časti je následne potrebné čo najpresnejšie spojiť hranami, ktoré do seba zapadajú. Elementy môžu byť po vygenerovaní úlohy aj rotované,

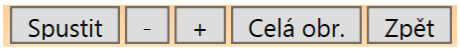
preto je ich možné počas ich spájania otočiť pomocou pravého kliku myši na príslušný element (Obrázok 7.24).



Obrázok 7.24 - Ukážka spustenej úlohy, kde je vidieť spájanie elementov a ich rotáciu.

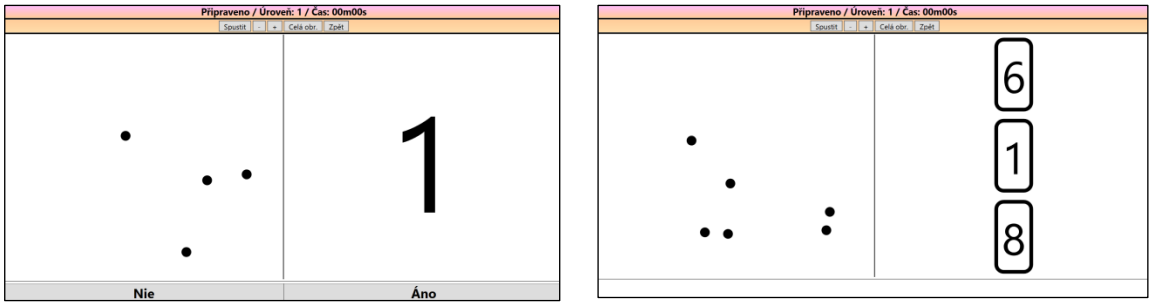
7.5 Úloha „Čísla“

Úloha je založená na porovnávaní počtu náhodne vygenerovaných bodiek so zobrazeným číslom, ktoré je taktiež generované náhodne. Počet bodiek, tak ako aj samotné číslo sú generované v rozmedzí čísel 0 až 10. Nie je preto potrebná žiadna významná konfigurácia, ktorá by zabezpečovala chod úlohy a jej variabilitu. Jedinou konfiguračnou možnosťou je úprava obtiažnosti úlohy, ktorá má za cieľ zmenu veľkosti oblasti, v ktorej sa generujú bodky. Tlačítko **„-“** zmenšuje danú oblasť a tlačítko **„+“** ju zväčšuje. Obidve tlačítka je možné nájsť na ovládacej lište úlohy (Obrázok 7.25).



Obrázok 7.25 - Ovládacia lišta úlohy "Čísla".

Úloha je rozdelená do dvoch modulov (**„ÁnoNie“** a **„Výber“**), ktoré predstavujú rozličný prístup jej riešenia. Ovládacia lišta je u oboch modulov rovnaká. Spúšťanie a zastavovanie úlohy u oboch modulov je rovnaké pomocou tlačítiek **„Spustit“** a **„Zastavit“**. Jednotlivé moduly sa odlišujú len vo výbere odpovede užívateľa. Pri module **„ÁnoNie“** je úlohou vybrať odpoveď na otázku, či sa počet bodiek rovná zobrazenému číslu. Na výber tejto odpovede slúžia tlačítka **„Áno“** a **„Nie“**, ktoré sú umiestnené v spodnej časti hracieho plátna (Obrázok 7.26).



Obrázok 7.26 - Moduly "ÁnoNie" (vľavo) a „Výber“ (vpravo) úlohy „Čísla“.

V module **„Výber“** je úlohou užívateľa vybrať z troch možností čísel to, ktoré je rovné zobrazenému počtu bodiek. Výber sa prevádza klikom na jedno zo zobrazených čísel (Obrázok 7.26).

7.6 Úloha „Správne slovo“

Aj táto úloha je rozdelená do dvoch modulov a to **„Zámena písmen“** a **„Vynechávanie písmen“**. Obidva sú založené na rovnakom princípe ovládania a konfigurácie. Preto je nasledujúci popis spoločný pre obidva moduly. Úlohou užívateľa v tejto úlohe je vybrať z pomedzi zobrazených slov to, ktoré je napísané gramaticky správne.

Konfigurácia úlohy

Konfigurácia úlohy prebieha pomocou konfiguračného textového súboru, ktorý je možné otvoriť pomocou tlačítka **„Upravit slova“**. Konfiguračný súbor sa otvorý v textovom editore, v ktorom je možné zmeniť jednotlivé nastavenia. Je možné v ňom definovať jednotlivé zoznamy slov, ktoré sa v úlohe generujú a zoznamy farieb, ktoré sa používajú v prípade, že majú byť vygenerované slová v úlohe zafarbené rôznymi farbami.

Zoznam slov sa v konfiguračnom súbore definuje ako jeden riadok. Začína číslom v hranatých zátvorkách, ktoré definuje úroveň obtiažnosti slova. Jeho hodnota môže nadobúdať len hodnoty 1, 2 alebo 3. Za týmto číslo nasleduje správne slovo, ktoré je ďalej nasledované nesprávnymi slovami. Medzi jednotlivými slovami, ale aj úrovňou slova a prvým správnym slovom je nutné vložiť medzeru, ktorá ich oddeľuje. Počet riadkov s takto definovanými slovami je neobmedzený. Príklad správne definovaného riadka so zoznamom slov:

[2] buben buden dubon duden bnbou bubeu

Zoznam farieb sa zadáva rovnakým spôsobom ako to bolo pri úlohe **„Spájanie“**. Farby sú definované pomocou škály RGB. Každý riadok popisujúci zoznam farieb musí začína malým písmenom **c**, za ktorým nasledujú definície jednotlivých farieb oddelené bodkočiarkou. Farba sa definuje pomocou škály RGB. Túto škálu tvorí rad za sebou idúcich čísel s hodnotou 0 – 255 oddelených medzerou. Počet definovaných farieb v riadku a počet riadkov s definovanými farbami je ľubovoľný. Príklad správne definovaného riadka so zoznamom farieb:

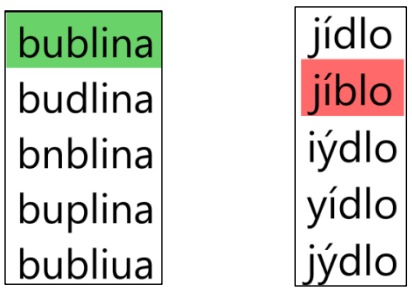
c 125 25 255; 0 145 0; 255 255 14

V konfiguračnom súbore je možné použiť aj komentovací symbol **„|“**. Pomocou neho je možné označiť tie riadky, ktoré nemajú byť pri načítaní konfigurácie použité. Symbol sa píše vždy na začiatok riadka, ktorý má byť označený ako komentár.

Okrem konfigurácie pomocou externého súboru, sa prevádza nastavenie úlohy aj pomocou tlačítiek **„+“**, **„-“** a **„Obarvit“** umiestnených na ovládacej lište úlohy. **„+“** a **„-“** slúžia na zvýšenie počtu zobrazených slov, čiže zvyšovanie alebo znižovanie obtiažnosti úlohy (Obrázok 5.26). Jednotlivé slová je možné aj zafarbiť pomocou tlačítka **„Obarvit“** (Obrázok 5.25). Ich farba je náhodne vybraná z farieb definovaných v konfiguračnom súbore.

Ovládanie úlohy

Úloha sa spúšťa a zastavuje pomocou štandardných tlačítiek **„Spustit“** a **„Zastavit“**, ktoré je možné nájsť na ovládacej lište úlohy. Po spustení úlohy užívateľ vyberá svoju odpoveď kliknutím na vybrané slovo. Ak je výber správny slovo sa zafarbí na zeleno, ak je nesprávny jeho farba je červená (Obrázok 7.27).



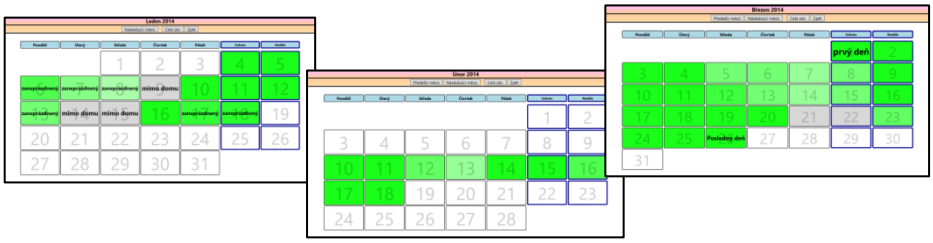
Obrázok 7.27 - Príklad správneho (vľavo) a nesprávneho (vpravo) výberu slova v úlohe "Správne slovo".

7.7 Rozšírenia frameworku

Keďže ide len o malé aplikácie slúžiace na zobrazenie spracovaných dát, ich ovládanie je veľmi minimalistické. Všetky rozšírenia obsahujú na ovládacej lište tlačítka na prepínanie medzi staršími alebo novšími dátami, ukončenie aplikácie (**„Zpět“**) a na zobrazenie aplikácie na celú plochu monitora. Niektoré obsahujú aj tlačítka na zväčšenie písma alebo priblíženie vykreslenia v prípade, že má užívateľ problém s čítaním malého textu.

Rošírenie „Používanie“

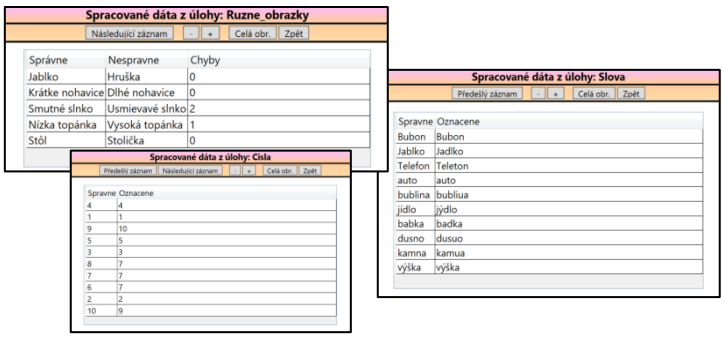
Toto rozšírenie používa na prepínanie medzi jednotlivými mesiacmi tlačítka **„Předešlý měsíc“** a **„Nasledujíci měsíc“** (Obrázok 5.28). Okrem nich neobsahuje žiadne špecializované tlačítka.



Obrázok 7.28 - Ukážka z rozšírenia frameworku "Používanie"

Rozšírenie „Tabuľka“

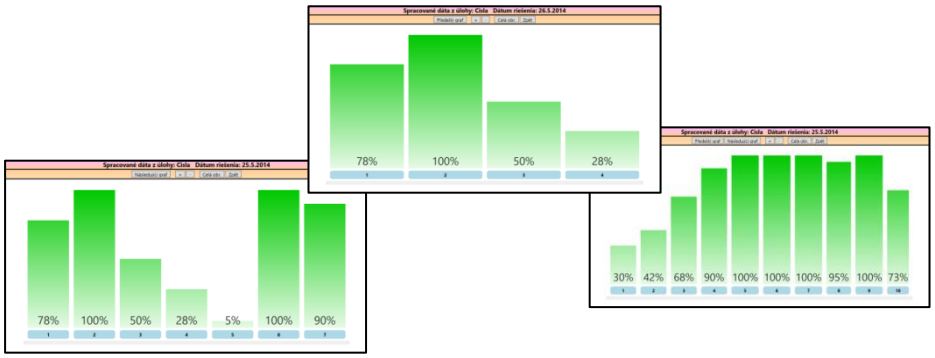
Aj pri tejto úlohe je potrebné prepínanie medzi jednotlivými spracovanými dátami. Na to slúžia tlačítka **„Předešlý záznam“** a **„Nasledujíci záznam“.** Okrem nich je možné pomocou **„+“** a **„-“** zväčšiť alebo zmenšiť veľkosť písma v tabuľke (Obrázok 5.34).



Obrázok 7.29 - Ukážka z rozšírenia frameworku "Tabuľka".

Rozšírenie „Graf“

Na prepínanie medzi jednotlivými výsledkami slúžia tlačítka **„Předešlý graf“** a **„Nasledujíci graf“**. Taktiež aj pri tejto úlohe je možne využiť tlačítka **„+“** a **„-“** na zväčšenie alebo zmenšenie vykreslenia grafu v prípade, že je graf príliš malý alebo veľký (Obrázok 5.36).



Obrázok 7.30 - Ukážka z rozšírenia frameworku "Graf".

8 Testovanie aplikácie

Testovanie aplikácie tohto druhu je možné všeobecne rozdeliť na dva typy:

- Testovanie z hľadiska spoľahlivosti programového kódu
- Testovanie z pohľadu užívateľskej vhodnosti a použiteľnosti

Dôležitejším typom testovania vytvorenej aplikácie v tejto práci bolo testovanie s takými užívateľmi a v takom prostredí, pre ktoré je táto aplikácia určená. Keďže sa vedúci tejto práce už dlhšiu dobu venuje vývoju podobného typu úloh a spolupracuje na ich tvorbe s lekármi, s ktorými sa pravidelne stretáva v ich ordináciách, bola využitá možnosť otestovania úloh priamo v tomto prostredí s pacientmi. Takéto testovanie nielenže pomôže odhaliť možné úpravy jednotlivých úloh z hľadiska ich použiteľnosti a vhodnosti, ale zároveň pomôže odhaliť aj prípadnú nefunkčnosť úloh z hľadiska programového kódu a ich behu. Čiže v podstate spojí obidva zmieňované typy testovania.

Testovanie prebiehalo v očnej ambulancii pod dozorom odborného personálu, konkrétne dvoch lekárov a vedúcim tejto práce, ktorý zároveň plnil úlohu technického dozoru nad testovaním. Skupinu užívateľov, ktorá prevádzala testovanie tvorili pacienti ordinácie, v ktorej testovanie prebiehalo za bežnej dennej prevádzky. Pacienti neboli vyberaný na základe žiadnych špecializovaných požiadavkou. Išlo o deti vo veku 5 – 10 rokov, ktoré trpia rôznymi očnými poruchami. Práve toto bolo jednou z výhod tohto testovania, pretože úlohy sú určené na testovanie už zmienených špecifických porúch učenia, čo sú v svojej podstate tieže veľmi často poruchy zrakového vnímania. Ich úlohou bolo otestovanie niektorých úloh z hľadiska ovládateľnosti a funkčnosti. Keďže išlo o deti v nižšom veku, nebolo možné pristúpiť k použitiu pretestového a posttestového dotazníka, preto bolo pristúpené k metóde záznamu postrehov, ktoré deti počas testovania poskytovali. Ďalšiu skupinu užívateľov, ktorý otestovali túto aplikáciu tvoril vyššie zmienení odborný personál ordinácie. Ich úlohou bolo hlavne ohodnotenie jednotlivých úloh z hľadiska grafickej a funkčnej vhodnosti a prípadné pripomienky k zmene alebo doplnení vlastností úloh, tak aby boli z odborného hľadiska čo najvhodnejšie. Pri takomto type aplikácie je takéto testovanie kľúčové a potrebné, pretože mnohokrát majú lekári odlišné požiadavky na jednotlivé úlohy, ktoré je potrebné včas odhaliť a konzultovať, aby mohli byť úlohy vhodne upravené. Samozrejme sa odkrývajú aj prípadné nedokonalosti pri samotnom behu aplikácie.

Aj v tomto prípade odhalilo testovanie nové poznatky a pripomienky zo strany odborného personálu, ale aj zo strany detí, ktoré mali pri plnení niektorých úloh problémy, ktoré zdraví človek len ťažko odhalí (aj keď nešlo priamo o cieľovú skupinu užívateľov trpiacich niektorou z tu zmienených porúch učenia). Nasledujúci zoznam špecifikuje nájdené problémy, z ktorých niektoré boli vyriešené okamžite a iné označené ako prípadná úprava do budúcnosti z dôvodu potreby dlhodobejšieho testovania a konzultácií s lekárom, aby boli nájdené čo najvhodnejšie riešenia ich úpravy.

Pripomienky z pohľadu pacientov

- Vhodnosť doplnenia niektorých úloh o zvukovú signalizáciu správnej a nesprávnej odpovedi – **Práca do budúcna**
- Vhodnejšia forma označenia správneho výberu pri niektorých úlohách – **Práca do budúcna**
- Zlá identifikácia rozdielov medzi obrázkami (ťažko rozoznateľné rozdiely) – **Spôsob nastavenia a aktuálne použitých obrázkov**
- Pri úlohe „Čísla“ sú tlačítka výberu „Áno“ a „Nie“ príliš malé – **Upravené**
- Úloha „Spájanie“ sa po poskladaní jedného objektu, zastavení úlohy a následnom spustení novej, ukončí skôr ako sú v novo spustenej konfigurácii poskladané všetky objekty – **Upravené**

+ Ocenenie modifikácie úloh pri ich opakovanom generovaní

Pripomienky z pohľadu odborného personálu

- V úlohe „Rôzne obrázky“ nejasnosť vo výbere úrovne, pretože pri najnižšej sú obrázky malé a pri najvyššej sú veľké, čo bolo pre personál zmätočné z dôvodu zvyku z iných úloh, ktoré používajú, kde je toto označovanie opačné – **Práca do budúcna**
- V úlohe „Rôzne obrázky“, by bolo vhodné rozlišovať obrázky podľa obtiažnosti, pretože niektoré skupiny zobrazených obrázkov v riadku boli zložitejšie ako skupina obrázkov v ďalšom riadku – **Spôsob nastavenia a aktuálne použitých obrázkov**
- Pri úlohe „Spájanie“ by bolo vhodná možnosť farebnej nápovedy, ktoré elementy majú ísť k sebe. – **Spôsob nastavenia a aktuálne použitých obrázkov**

- V úlohe „Čísla“ vytvoriť možnosť zväčšovania a farebného rozlíšenia vykreslených bodiek, poprípade možnosť zobrazenia aj iných geometrických tvarov (štvorce, trojuholníky, obdĺžniky) – **Práca do budúcna**
- V úlohách „Čísla“ a „Správne slovo“ pri výbere nesprávnej odpovede neprepínať úlohu na ďalšiu, ale ponechať možnosť ďalšieho výberu pokým pacient nevyberie správnu odpoveď – **Práca do budúcna**

\+ Ocenenie jednoduchého pridania / rozšírenia úloh, hlavne použitím textového súboru, ktorý je dobre zrozumiteľný

Pripomienky označené ako **„Upravné“** boli ešte v rámci tejto prace úplne vyriešené. Išlo o pripomienky, ktoré buď spôsobovali nesprávny chod úlohy, alebo išlo o jednoduchšie úpravy, pri ktorých nie je potrebné dlhodobejšie testovanie.

Pri pripomienkach označených ako **„Spôsob nastavenia a aktuálne použitých obrázkov“** ide o použité konfigurácie úloh hlavne zo strany použitých obrázkov. Obrázky neboli starostlivo vyberané odbornou osobou, čiže lekárom, ale programátorom pre overenie činnosti aplikácie. Ani ich konfigurácia nebola vytváraná lekárom, preto bolo v úlohách pri niektorých obrázkoch zložitejšie rozoznávanie odlišností, ktoré boli spôsobené nie vhodnou konfiguráciou týchto obrázkov. Preto nejde v tomto prípade o pripomienky, vďaka ktorým je potrebné meniť programový kód úloh, ale len ich nastavenie, ktoré bude v budúcnosti vykonávať odborná osoba.

Pripomienky, ktoré sú označené ako **„Práca do budúca“** sú prípadne možné rozšírenia aplikácie v budúcnosti. Je to hlavne z dôvodu, že ide o nastavenia, ktoré nie je možné bez dlhodobejšieho testovania a konzultácie s lekárom v ordinácii aplikovať. Je potrebné presné zistenie vhodnosti týchto rozšírení a hlavne ich najlepšej formy aplikácie na jednotlivé úlohy, čo poskytujú len výsledky dlhodobejšieho testovania. Z tohto dôvodu bude vhodné, keď bude aplikácia nejaký čas skutočne využívaná, vhodne tieto požiadavky spísané a až potom využité pri úprave aplikácie. Na druhej strane príliš rýchla úprava aplikácie podľa prvého názoru užívateľov nemusí byť vždy vhodná.

9 Záver

Cieľom tejto práce bol návrh a implementácia aplikácie zastrešujúcej úlohy určené na testovanie troch vybraných špecifických porúch učenia, ktorými sú dyslexia, dysgrafia a dyskalkulia. Hlavnou myšlienkou tvorby takéhoto typu aplikácie je obmedzený súčasný stav trhu, ktorý v dnešnej dobe neposkytuje vhodné aplikácie, ktoré by zoskupovali úlohy určené na testovanie týchto porúch. Zväčša ide len o úlohy v brožúrovanej podobe alebo úlohy, ktoré sú uniformné a neumožňujú tvorbu rôznych variant. Na začiatok bolo potrebné zozbierať a vytvoriť prehľad vhodných úloh. Z nich bola vybraná reprezentatívna vzorka 5 úloh, ktoré boli následne implementované v tejto aplikácii. Ide o úlohy **„Rozdiely“**, **„Rôzne obrázky“**, **„Spájanie“**, **„Čísla“** a **„Správne slovo“.** Pri návrhu jednotlivých úloh bolo potrebné brať do úvahy základné požiadavky na ich tvorbu a to konfigurovateľnosť, adaptativita, zber dát potrebných pri diagnostike stavu pacienta a v neposlednom rade aj použitá technológia (Microsoft WPF.NET). Najdôležitejšou požiadavkou je aby bolo úlohy možné konfigurovať, čo umožní premenenie jednej vytvorenej úlohy na množinu rôznych variantov, ktoré sa môžu svojim riešením zameriavať na rôzne špecifické poruchy učenia. Na druhej strane sa z úlohy stáva variabilnejší nástroj na diagnostiku a rehabilitáciu týchto porúch. Pri úlohách je podstatná aj adaptativita, ktorá vytvára možnosť prispôsobenia úlohy na základe zdatnosti užívateľa, čo rozširuje použiteľnosť úlohy na širšiu základňu cieľových skupín. Z pohľadu použitej technológie nebol návrh a ani implementácia úloh nijak obmedzená, keďže Microsoft WPF.NET poskytuje mnoho knižníc poskytujúcich podporu pri tvorbe základných 2D a 3D objektov na tvorbu grafických prvkov úloh až po rôzne animácie vhodné pre tvorbu rôznych pohybujúcich sa objektov. Na samotnú tvorbu úloh bol použitý framework založený na vybranej technológii (Microsoft .NET), ktorý bol vytvorený v roku 2011 vedúcim práce za účelom tvorby takýchto úloh a v dnešnej dobe je plne využívaný v praxi. Jeho výhodou je, že ponúka jednotné aplikačné rozhranie, ktoré zastrešuje vytvorené úlohy. Pre vývojárov ponúka základnú štruktúru úlohy, množstvo metód a dátových typov, ktoré uľahčujú ich tvorbu. Nemenej dôležitou súčasťou bolo zabezpečenie zberu dát. To však vďaka použitému frameworku, nevytvorilo žiadny problém, keďže obsahuje potrebné metódy a dátové typy určené na zber a následné odoslanie dát na server. Framework bol v rozsahu tejto práce taktiež rozšírený o nové súčasti určené na zobrazenie nameraných a spracovaných dát z riešenia úloh. Boli navrhnuté a vytvorené 3 typy zobrazenia dát a to kalendár používania, jednoduchá tabuľka a stĺpcový graf. Pri ich návrhu bolo potrebné dbať, aby boli všetky formy zobrazenia čo najjednoduchšie a zrozumiteľné, preto bol kladený dôraz hlavne na grafickú respektíve farebnú reprezentáciu dát. V časti implementácie aplikácie sa práca zaoberá neštandardnými riešeniami

použitými pri tvorbe úloh. Boli vybrané a popísané dve takéto riešenia, konkrétne spájanie objektov v úlohe „Spájanie“ a animácia stĺpcov a hodnoty grafu v rozšírení frameworku „Graf“. Po implementácii aplikácie, bola podrobená testovaniu s pacientmi v ordinácii. Testovanie prebiehalo pod odborným dohľadom zo strany lekárov, ale aj technického zo strany vedúceho práce, ktorý úzko spolupracuje s ordináciou, v ktorej bol test prevádzaný. Testovane skupiny užívateľov tvorili pacienti ordinácie vo veku od 5 do 10 rokov a dvaja odborný pracovníci ordinácie. Počas testovania boli zaznamenané niektoré pripomienky, z ktorých niektoré boli upravené hneď a niektoré ponechané ako práca do budúcna z dôvodu potreby dlhodobejšieho testovania a konzultácií s lekárom o najvhodnejšom riešení.
Výsledok tejto práce je vhodný ako prototypová a teda základná aplikácia pre použitie nielen v ordinácii, ale taktiež v domácom prostredí, v oblasti už spomenutých špecifických porúch učenia.

Z môjho pohľadu mi táto práca určite dala mnoho nových skúsenosti v tvorbe aplikácií pomocou technológie Microsoft WPF.NET, vo využívaní jej knižníc, komponentov a metód. Ako nezanedbateľný prínos by som určite označil aj zoznámenie sa so samotnými špecifickými poruchami učenia a tvorbou úloh na diagnostiku a ich liečbu, čo mi prišla ako veľmi zaujímavá a hlavne užitočná téma.

Použitá literatura

[1] Hartmann, Tesnoval. (2011) Harmann, Tesoval. [Online]. http://tensoval.sk/biely-plast.php

[2] Lechta, Viktor a kolektív. (1990) Logopedické Repetitórium. PDF.

[3] V. Pkorná. (2001) Teorie a náprava vývojových poruch učení a chování. PDF.

[4] K. Chroboková, "Specifické poruchy učení," Stredoškolská odborná činnosť, 2007.

[5] Univerzita Jana Evangelisty Purkyně v Ústí nad Labem. (2014) Medzinárodní konference psychomotoriky Univerzita Jana Evangelisty Purkyně v Ústí nad Labem. [Online]. http://www.psychomot.cz/?page_id=112

[6] M. Blahutková. (2006) Psychomotorika pre každého. PDF.

[7] Pedagogicko-psychologická poradna Ústeckého kraje. [Online]. http://www.pppuk.cz/soubory/chomutov-ss-a-sa_2.doc

[8] Psychosoft. Psychosoft. [Online]. http://d3dsacqprgcsqh.cloudfront.net/photo/aeN8zwm_460sa_v1.gif

[9] Y. Mintzker. (2002) Assessing the Cognitive Modifiability of Children and Youth with Down Syndrome. [Online]. http://www.riverbendds.org/index.htm?page=mintzker.html

[10] Centrum pedagogicko-psychologického poradenstva a prevencie. Centrum pedagogicko-psychologického poradenstva a prevencie. [Online]. http://www.cpppple.sk/odporuc%20alenka/pred%20rozvij%20spec%20funk.pdf

[11] Z. Matějček, *Vývojové poruchy čtení*, 3rd ed. Praha: SPN, 1975.

[12] K. Veselá, "Fonologické uvědomění a počáteční čtení," Jihočeská Univerzita Pedagogická fakulta Katedra pedagogiky a psychologie Diplomová práca, 2007.

[13] T. Repček. (2012, Oct.) ini.sk. [Online]. http://www.ini.sk/mam-doma-dyslektika-do-certa/

[14] anthony. (2011, Jan.) ux movement. [Online]. http://uxmovement.com/content/6-surprising-bad-practices-that-hurt-dyslexic-users/

[15] L. Bednarčíková, T. Ustohalová, and E. Suchožová. (2010) Reedukácia špecifických porúch učenia pomocou rozvíjania grafomotoriky. PDF.

[16] Občianske združenie PreDys. Občianske združenie PreDys. [Online]. http://www.predys.szm.com/poruc.htm

[17] Katedra speciální pedagogiky Pedagogickej fakulty Masarykovy unoverzity. Katedra speciální pedagogiky Pedagogickej fakulty Masarykovy unoverzity. [Online].

http://www.ped.muni.cz/wsedu/dyskalkulie/dyskalkulie.html

[18] The Absolute Value Of Mike. [Online]. http://theabsolutevalueofmike.weebly.com/dyscalculia.html

[19] Dyskalkulie - I dyskalkulik se může matematiku naučit. [Online]. http://dyskalkulie.webgarden.cz/rubriky/specificke-poruchy-uceni/typy-dyskalkulie

[20] J. Adamček, "Nejen psychologické testy jako webová aplikace," ČVUT Bakalárska práca, 2012.

[21] Masarykova univerzita Fakulta sportovních studií. [Online]. http://www.fsps.muni.cz/~tvodicka/data/reader/book-11/10.html

[22] Meditron. (2007,) Meditron. [Online]. http://www.meditron.de/en/hess-screen/hess-interp_results.shtml

[23] J. Jošt, *Čtení a dyslexie*. Grada, 2011.

[24] M. McDonald, *Pro WPF in C# 2010*. Apress, 2013.

[25] Nagel, Evjen, Glynn, and Skinner, *Professional C# 4 and .NET 4*. Wrox, 2010.

Príloha A

Vysvetlivky pojmov

ASP – Špecifikácia technológie Microsoft.NET, na tvorbu serverovo založených webových aplikácií

C# - Programovací jazyk používaný v technológii Microsoft.NET.

F# - funkcionálny jazyk spojený s objektovo-orientovaným prístupom používaný v technológii Microsoft .NET

Flash – Prostredie určené na vytváranie animácií, hlavne pre webové využitie. Vytvorené spoločnosťou Adobe.

Framework – Platforma uľahčujúca svojimi podpornými knižnicami, metódami a premennými, jednoduchú tvorbu aplikácií.

Funkcionálny jazyk – Jazyk, pomocou ktorého sa zapisuje programový kód v tvare výrazov, ktoré sa postupne zjednodušujú.

GUI (Graphic User Interface) – Grafické užívateľské rozhranie, pomocou ktorého užívateľ ovláda Aplikáciu

HTML (HyperText Markup Language) - je značkový jazyk určený na vytváranie webových stránok a iných informácií zobraziteľných vo webovom prehliadači.

JavaScript – Skriptovací programovací jazyk, vyžívaný najmä pri tvorbe webových aplikácií

Microsoft .NET - Je zastrešujúci názov pre súbor technológií v softvérových produktoch firmy Microsoft, ktoré tvoria celú platformu.

Mono – Špecifikácia technológie Microsoft .NET, ponúkajúca sadu nástrojov na tvorbu aplikácií používaných na operačnom systéme Linux.

PHP (Hypertext Preprocessor) – Skriptovací jazyk, ktorý sa využíva hlavne na programovanie klient-server aplikácií (na strane servera) a pre vývoj dynamických webových stránok.

Silverlight - Špecifikácia technológie Microsoft.NET, ponúkajúca sadu nástrojov na tvorbu webových aplikácií.

Visual Basic – Programovací jazyk používaný v technológii Microsoft .NET

WPF - Špecifikácia technológia Microsoft.NET, ponúkajúca sadu nástrojov na tvorbu desktopových aplikácií

Xamarin – Špecifická technológia Microsoft .NET, ponúkajúca sadu nástrojov na tvorbu mobilných aplikácií bežiacich na operačnom systéme Android

Príloha B

Obsah priloženého CD

Priložený CD disk obsahuje túto bakalársku prácu vo formáte WORD 2010 (.docx) a vo formáte PDF.

Na disku sa nachádza aj zoznam úloh vhodných na detekciu a liečbu špecifických porúch učenia, z ktorého boli vybrané implementované úlohy. Je vo formáte WORD 2010 (.docx), ale aj vo formáte PDF.

Súčasťou disku je aj preložená EXE verzia implementovanej aplikácie spolu s textovým súborom README.txt, kde je možné nájsť informácie k aplikácii. V prípade ďalších informácií ohľadom aplikácie smerujte svoje otázky na vedúceho tejto práce (Ing.Petr Novák, Ph.D., novakpe@labe.felk.cvut.cz).