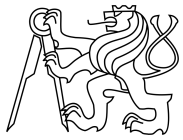
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	10K SAGE2 Dashboard: Sada widgetů pro velkoplošnou obrazovku
Student:	Aleh Kuchynski
Vedoucí:	Ing. Radek Richtr
Studijní program:	Informatika
Studijní obor:	Web a multimédia
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2017/18

Pokyny pro vypracování

Úkolem je vytvořit specializovaný dashboard sestávající ze sady SAGE2 (Scalable Amplified Group Environment, velkoplošné zobrazovací virtualizační zařízení tvořené 20 LCD displeji řízené sw. SAGE a SAGE2 postaveným na web. technologiích) aplikací schopných získat a vizualizovat textová (RSS, zprav. kanály, ...) a obrazová (mapy, grafy, ...) data.

1) Proveďte analýzu API SAGE2, stávajících aplikací, možnosti zobrazení různých dat a možnosti modulového systému jedné zastřešující aplikace kontrolující jednotlivé zobrazovací widgety.

2) Stanovte obecná i konkrétní omezení a doporučení pro budoucí vyvíjená ovládací rozhraní (vzhled, ovládání, ...).

3) Navrhněte:
- způsob realizace jednotlivých aplikací včetně jejich organizace, komunikace, konfigurace a ovládání,
- způsob získávání dat z externích zdrojů,
- minimálně 5 konkrétních aplikací (např.hodiny, čas. zóny, mapu glóbu, soc. sítě, ...).

4) Implementujte alespoň 3 z navržených aplikací.

5) Vytvořené aplikace vhodně otestujte.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 17. února 2017

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

Bakalářská práce

10K SAGE2 Dashboard: Sada widgetů pro velkoplošnou obrazovku

Aleh Kuchynski

Vedoucí práce: Ing. Radek Richtr

15. května 2017

Poděkování

Chtěl bych poděkovat Ing. Radku Richtrovi, vedoucímu mé práce, za odborné vedení a konzultace, cenné rady, trpělivost a čas, který mi v průběhu zpracování bakalářské práce věnoval. Dále bych rad poděkoval Jiřímu Kubiště, správci SAGElab, za pomoc během nasazení aplikací. V neposlední řadě bych chtěl poděkovat participantům testování použitelnosti realizovaných aplikací za jejich názory a své přítelkyně za psychologickou podporu během napsání teto práce.

Prohlášení

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

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

V Praze dne 15. května 2017 .....

Odkaz na tuto práci

Kuchynski, Aleh. *10K SAGE2 Dashboard: Sada widgetů pro velkoplošnou obrazovku*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

SAGE2 je platforma, která umožňuje týmům uživatelů manipulovat s jejich data na stěně složené z mnoha displejů. Pomocí webových technologií nabízí uživatelům možnost spolupráce s multimediálním kontentem v reálném čase. Cílem teto bakalářské práce je provést analýzu SAGE2 API (Application programming interface), navrhnout a implementovat sadu widgetů pro SAGE2.

Autorem byly realizované následující aplikace: digitální hodinky s možností zobrazení počasí a budíkem, SAGE2 klient pro sociální síť Instagram a univerzální konstruktor grafů. V teto práci čtenář najde zájmové problémy, se kterými se autor setkal během realizace widgetů a způsoby jejich řešení.

Klíčová slova telestěna, widgety, velké rozlišení, kolaborativní práce, Instagram, grafy, SAGE2

Abstract

SAGE2 is a platform that allows users' teams to manipulate with their data on a wall made up of multiple displays. Using web technologies, it offers users the ability of collaborating work with multimedia content in real-time. The aim of this bachelor thesis is analyze the SAGE2 API (Application programming interface), design and implementation of a set of widgets for SAGE2.

The following applications were made by author: digital watches with an alarm and weather display, the SAGE2 client for the social network Instagram and the universal chart constructor. In this work, the reader will find interesting issues that the author encountered during the implementation of widgets and ways to solve them.

Keywords display wall, widgets, high resolution, collaboration work, Instagram, charts, SAGE2

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Dávno bylo zavedeno, že společná práce nad projektem přináší spoustu výhod: větší počet názorů, rychlost, lepší kontrola atd. A jelikož v dnešní době máme prostředky k řešení různorodých IT problémů – aktivně se rozvijí produkty pro kolaborativní práci. Díky tomu vzniká mnoha software a hardware, které mají za cíl usnadnit a udělat takový přístup k práci mnohem pohodlnější. Většina takových nástrojů je ovšem zaměřená na sdílený přístup mezi jednotlivci za pomocí vzdáleného připojení k počítači a obrazovce. Jsou ale projekty, u kterých je preferovaná spolupráce více lide nad velkým objemem dat v reálném čase. Pro tyto účely je možné použít například panely dělených LCD displejů. Ovšem vzniká potřeba realizovat nastroj, který by umožnil takové panely řídit, tzn. vizualizovat data ve velkém rozlišení a umožnit je snadno manipulovat.

V roce 2009 vznikla platforma SAGE, která pomohla vyřešit tento problém pro vice než 100 významných instituci. V roce 2014 následuje SAGE2 – nová verze původního SAGE, která je na rozdíl od předchůdce, napsaného v programovacím jazyce C++, je založená na webových technologiích a spouští se v prohlížeči, což přináší ještě více výhod jak pro uživatele, tak i pro vývojáře [2]. Nezbytnou části tohoto nástroje jsou tak zvané widgety – menší aplikace, které běží v kontejneru SAGE2. Zmíněná platforma má svoje API, které pomáhá vývojářům jednoduše vytvářet vlastní widgety, vytvoření kterých je velkým přínosem pro rozvoj SAGE2, její popularity a funkcionality. Umožní to všem uživatelům teto platformy snadnější práci a pohodli, což je největší motivace autora při výběru téma bakalářské práce.

Cíl práce

Autor bakalářské práce stanovil před sebou cíl provést rešerši SAGE2 API, stávajících widgetů pro SAGE2 a zároveň možnosti a omezení pro vytvoření

vlastních aplikací. Na základě teto analýzy navrhnout nejméně pět různých widgetů a implementovat alespoň tři z nich.

Vytvořené aplikace by měli přinášet něco nového, být konfigurovatelné a dokumentované, mít možnost ovládáni za pomocí vstupního zařízení. Není cílem daného projektu zaměření na grafickém rozhraní. Modernizace a vylepšení vzhledu bude náplně dalších navázaných bakalářských prácí. Widgety musí být řádně otestované: a to jak funkcionalita, tak i uživatelské rozhraní ve FIT SAGElab[2].

Výsledkem teto práce by mělo být dosažení zmíněného cíle nebo dokázání, že postavený úkol nejde vyřešit v rámci dané bakalářské práce.

Struktura práce

Kapitola 1 bude věnovaná seznámení s SAGE2, její API, možnosti a omezení, analýze stávajících widgetů a vysvětlení, potřebných k pochopení teto bakalářské práce, pojmů.

V kapitole 2 bude následovat návrh řešení postaveného úkolu. Na základě výsledků analýzy z předchozí kapitoly a zadání práce budou stanovené funkční a nefunkční požadavky. Mimo jiné, čtenář zde najde návrh uživatelského rozhraní a prototypy budoucích aplikací.

Další kapitola 3 se bude věnovat implementaci navřených widgetů. Zde budou probrané všechny aspekty a problémy, které autor řešil během vývoje. Čtenáři budou představené zajímavé vlastnosti a funkce implementovaných aplikací.

Kapitola 4 bude obsahovat návrh testovacích scénářů uživatelského rozhraní a výsledky testování, které proběhne ve SAGElab (technická laboratoř Fakulty informačních technologií ČVUT, která mimo jiné, je vybavená panelem displejů s celkovým rozlišením 9600×4320).

Závěr se bude věnovat výsledkům a vyhodnocení provedené práce, její přínosem a výhledem do budoucnosti.

[2] https://sagelab.cesnet.cz/cz/

Kapitola 1

Analýza

V této kapitole jsou popsané současný stav platformy SAGE2, její výhody a nedostatky, porovnání s dalšími existujícími produkty pro velké panely displejů. Zde čtenář také najde analýzu SAGE2 API, na základě které budou v další kapitole stanovené funkční a nefunkční požadavky.

1.1 Terminologie

Předtím, než se čtenář pustí do zkoumání dalších částí, je vhodné se seznámit se základními pojmy a názvosloví, které jsou používané během teto bakalářské práce, čím pomohou lepé pochopit její obsah.

Middleware – obecný pojem pro programové vybavení, které slouží jako *lepidlo* pro jíž existující a často komplikované aplikace [3].

Widget – obecný typ aplikací, které mají vlastnost přenositelnosti mezi jednou či několika různými platformami [4].

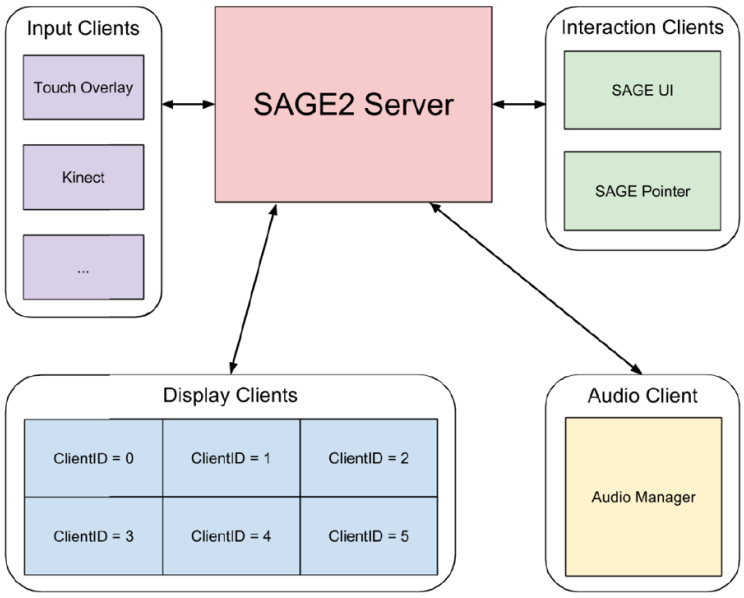
API – neboli rozhraní pro programování aplikací. Jedná se o balík knihoven, funkcí či nějakých procedur, které může programátor využívat a ovládat či komunikovat se softwarem [5].

OAuth 2.0 – je moderní autorizační protokol (resp. framework), který se stal standardem pro zabezpečení webových služeb [6].

1.2 Přehled SAGE2 a analogických produktů

Platforma SAGE byla rozpracovaná v roce 2009 skupinou studentů dvou univerzit: University of Hawai'i v Mānoa[3] a University of Illinois v Chicago[4]. Jak

[3] https://manoa.hawaii.edu/
[4] http://www.uic.edu/



Obrázek 1.1: Architektura SAGE2 [1].

je tomu napsáno na oficiálních stránkách SAGE: cílem tohoto middleware je umožnit lidem se dívat na velké množství informace a nacházet řešení rychleji, přesněji a úplněji [2]. V roce 2014 SAGE byla kompletně přepsaná a tím vznikla její nástupkyně SAGE2. Nová platforma je napsaná v JavaScriptu, používá webové technologie a běží v prohlížeči, což přináší nezávislost na platformách a operačních systémech, jednoduchou škálovatelnost widgetů atd.. Hlavní důraz v nové verzi je kladen na kolaborativní práci: každý uživatel může nadefinovat vlastní kurzor a takovým způsobem, víc lidí mají možnost současně spolupracovat na jedné ploše [7].

Podle [1] tým vývojářů SAGE2 měli několik důvodů pro přechod na použití webových technologií. Se vznikem HTML5 a WebGL webové prohlížeči začali podporovat 2D a 3D rendering. Byla standardizovaná WebSocket komunikace, která umožňuje oboustranné spojení mezi klientem a serverem. Navíc prohlížeči mají nativní podporu zpracování nastalých událostí od vstupních zařízení: myší, klávesnic a dotykových displejů. Webové prohlížeči lze najít na libovolném vizuálně výpočetním počítači. Nepotřebují obtížnou instalaci za pomocí technika a jsou nezávislé na platformě. Založena na webových technologiích komunikace také umožňuje spolupráci a sdílení dat v reálném čase.

Na obrázku 1.1 je zobrazena architektura SAGE2, která se skládá z následujících komponentů:

	Multi-user	Podpora pluginů nebo widgetů	Škálovatelnost a neomezené rozlišení	Open source	Veřejné API	Cross-platform	Free
POLYWALL			✓				*[8]
FrontFace	✓	✓	✓		✓		
Userful		✓	✓	✓	✓	✓	*[8]
9X Media Multi-Screen Manager	✓		✓				
CGLX		✓	✓	✓	✓		✓
eyeUNIFY	✓	✓	✓	✓	✓	✓	✓
SAGE2	✓	✓	✓	✓	✓	✓	✓

Tabulka 1.1: Porovnání middleware pro panely displejů[9]

- Server – server je postaven na Node.js[5]. Podporuje SSL, WebSocket[6] – protokol pro oboustrannou komunikaci mezi klientem a serverem, webové služby. Jednou z výhod Node.js je možnost použíti NPM-manažeru – správce balíčků, který slouží pro snadné přidání a řízení externích knihoven.

- Dislpay clients – v roli takových klientů v SAGE2 vystupují instance webového prohlížeče, které jsou připojené k serveru za pomocí URL.

- Audio clients – aplikace, které také běží v prohlížeči a jsou definované pro každý zobrazovací klient. Audio klienty mají za cíl řešit problém s přehráváním zvuků z více zdrojů. Takové klienty přijímají jednotlivé zvuky, následně probíhá jeho mixování a výsledek se přehrává jako jediný celek.

- Interaction clients – v případě, že uživatel navštíví za pomocí URL stránku s klientem pro interakci, která se nazývá SAGE UI, uvidí přehled panelu displejů a bude moct nastartovat SAGE2 aplikace, sdílet lokální dokumenty či svoji obrazovku, přidat kurzor pro manipulaci s widgety.

SAGE2 se primárně používá pro vizualizaci dat, sdílení a práci s mediálním kontentem na panelu displejů. V současné době popisovaný middleware je používaný ve 67 institucích a tento počet roste s každým rokem[8], z čehož lze udělat závěr, že platforma je populární a aktivně se rozvije SAGE2 má technickou podporu ve formě komunity uživatelů, kteří odpovídají na vzniklé otázky na oficiálním fóru[7].

[5] https://nodejs.org/en/
[6] https://w3c.github.io/websockets/
[7] https://groups.google.com/forum/sage2

Níže jsou popsané nejznámější konkurenti SAGE2.

POLYWALL[10] je software společnosti VISIOLOGIC. Umožňuje zobrazovat a pracovat s libovolnými daty na telestěnach. Podporuje scénáře pro automatické přehrávání kontentu. Platforma je spíše zaměřená na komerční použití, protože bezplatná verze je hodně omezená.

FrontFace[11] je platforma, která má bohatou funkcionalitu, podporu pluginů a veřejné API pro vývoj rozšíření. Nevýhodami tohoto software jsou kompatibilita jenom s operačním systémem Windows a absence bezplatné verze.

Userful[12] je open source projektem. Má vystavené veřejné REST API pro komunikaci se serverem a širokou sadu pozitivních vlastností. Nepodporuje však společnou práci více uživatelů.

9X Media Multi-Screen Manager[13] – kvalitní řešení pro podniky a větší společnosti. Mezi její zákazníky například patří NASA, armáda USA, Boeing atd. Neexistuje bezúplatná verze a podporuje jenom Windows.

CGLX[14] – open source middleware pro panely displejů. Je zdarma, má širokou funkcionalitu, ale nepodporuje možnost kolaborativní práce a nemůže být použitá s Windows.

eyeUNIFY[15] – na první pohled je velice dobrým řešením. Produkt je postaven na webových technologiích a má skoro stejné vlastnosti jako SAGE2. Ale na rozdíl od SAGE2 zahrnuje pěkně dokumentované API, což je velkou výhodou pro vývojáře pluginů.

Na základě krátkého porovnání 1.1 se dá udělat závěr, že SAGE2 má bohatší funkcionalitu než drtivá většina jiných populárních systémů.

Autor udělal výběr ve prospěch SAGE2 díky její velkému potenciálu, možnosti zkoušení svoje práce v SAGElab a podílet se na síření funkcionality a popularity teto platformy.

1.3 Analýza SAGE2 API

Jak již bylo zmíněno, SAGE2 je postavena na webových technologiích. Pro vývoj widgetů se používá JavaScript. Vývojáři mají možnost přidání a použití libovolných externích JavaScript knihoven. Jak je popsáno v dokumentaci k SAGE2 API [9], nová aplikace může být jednoduše vytvořena spuštěním příkazu `npm run newapp` v kořenovém adresáře SAGE2. Výsledkem by se měla stát vytvořena kostra nového widgetu, která má následující strukturu 1.2.

[8] Existuje omezená bezplatná verze

[9] Čím větší počet ✓ má software, tím lépe

[10] http://polywall.net/Home/About

[11] https://www.mirabyte.com/en/products/frontface-for-public-display

[12] http://www.userful.com

[13] http://www.9xmedia.com/new/products/multi-screen-software.php

[14] http://vis.ucsd.edu/ cglx/

[15] https://eyeunify.org/overview

```
[nazev aplikace]
├── [nazev aplikace].js ..... hlavní JavaScript soubor
├── [nazev aplikace].png ..... ikonka aplikace
└── instruction.json ..... iniciální nastavení widgetu
```

Obrázek 1.2: Struktura nově založeného projektu.

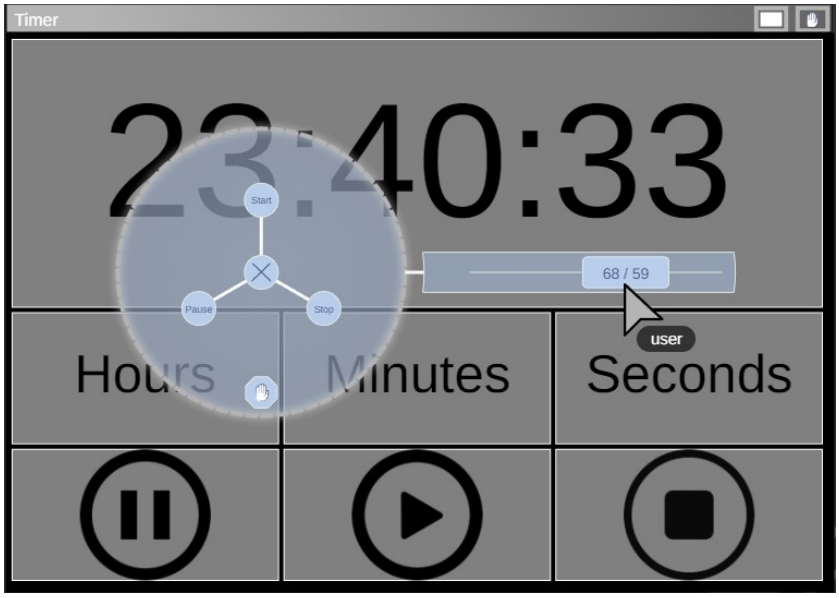
`[nazev aplikace].js` obsahuje třídu se stejným názvem jako widget, která rozšiřuje třídu `SAGE2_App` ze SAGE2 API. `SAGE2_App` slouží vstupním bodem pro tvoření vlastního widgetu a obsahuje mnoha metod, které mají za cíl usnadnit práci programátora. Zde autor bakalářské práce představí z jeho pohledu jenom nejvíce užitečné metody:

- `init(data)` – metoda se vyvolá na začátku procesu počáteční inicializace widgetu. Parametr `data` – objekt, který obsahuje iniciální hodnoty, například souřadnice, šířku a výšku okna widgetu atd..
- `draw(date)` – vyvolá se v případě překreslení aplikace. Metodu lze zavolat implicitně v kódu nebo nastavit časový interval pro překreslení (frekvence záběrů) v souboru `instruction.json`. Parametr `date` je aktuální datum.
- `resize(date)` – metoda se vyvolá v okamžik změny uživatelem rozměru okna widgetu. Parametr `date` je aktuální datum.
- `event(type, position, user, data, date)`, kde

type	typ vzniklé události
position	souřadnice události `x` a `y`
user	objekt, obsahující informace o uživateli, který způsobil spuštění události
data	objekt, který obsahuje extra data o události
date	aktuální datum a čas

  Metoda se vyvolá v případě vzniku takových událostí, jako kliknutí myši, stisknutí klávesy, pohyb kurzoru atd. Kompletní seznam podporovaných typy událostí je popsán v číselníku uvnitř třídy `SAGE2_App`.

SAGE2 API zahrnuje i užitečné kontextové menu. API obsahuje několik připravených elementů, které se dají v tomto kontextovém menu použít. Jsou to tlačítko, slider a textové pole pro uživatelský vstup. Na obrázku 1.3 je vidět menu widgetu Timer (je součástí distribuce SAGE2) s definovanými tlačítky 'Start', 'Pause', 'Stop' a sliderem pro nastavení počtu minut.



Obrázek 1.3: Kontextové menu widgetu Timer.

Největší problém, který autor vidí v SAGE2 je chudá dokumentace API. Aktuálně se dokumentace pro vývojáře nachází na oficiální webové stránce a v repositáři SAGE2 na BitBucket[16]. Dokumentace popisuje jenom část funkcí: k čemu slouží, jaké mají parametry atd. Navíc to, co je popsáno, je opravdu jenom základem používání API. V případě, že vývojář má na mysli vytvořit vlastní pokročilý widget, bude muset hledat informaci a klást dotazy na oficiálním technickém fóru SAGE2[17] (fórum není veřejně dostupný – je potřeba požádat o přístup). Druhá možnost je analyzovat kód již existujících aplikací a snažit se pochopit principy práce s SAGE2 API na příkladech. Autor bakalářské práce předpokládá, že takový problém existuje jenom kvůli tomu, že produkt je ještě mladý a věří, že dokumentace bude v budoucnu kompletně dopracovaná.

1.4 Analýza stávajících widgetů

Na začátku roka 2017 se spustil SAGE Appstore[18] – webový repositář, kde každý má možnost nahrát vlastní nebo stáhnout již existující widgety pro SAGE2. V okamžiku napsaní bakalářské práce v Appstore byly přístupné čtrnáct aplikace. Ještě jedenáct jsou dodávané spolu s distribuci SAGE2. Většina widgetů jsou zaměřené na práci z video, obrázky a dokumenty. Není v tom ale

[16] https://bitbucket.org/sage2/sage2
[17] https://groups.google.com/forum/#!forum/sage2
[18] http://apps.sagecommons.org/

nic nečekaného, protože to odpovídá účelům samotné SAGE2, kde vizualizace a manipulace s multimediálním kontentem je její základním použitím.

Během analýzy stávajících widgetů bylo zjištěno, že většina aplikací je tvořena pomocí SVG. Autor předpokládá, že důvodem pro používání SVG místo například HTML5 Canvas je to, že SVG má řadu výhod:

- Je to standardizovaná a doporučená W3C konsorciem technologie [10].
- SVG udržuje referencí pro každý jim vytvořený objekt. Každý prvek SVG je skutečným elementem v DOM (Document Object Model).
- SVG má podporu událostí: kliknutí myši, stisknutí kláves atd..
- SAGE2 API zahrnuje d3.js[19] – JavaScript knihovna pro vizualizaci a animaci dat pomocí SVG a DOM.

Všechny výšeuvedené výhody dělají SVG dobrou volbou během vývoje aplikací pro SAGE2.

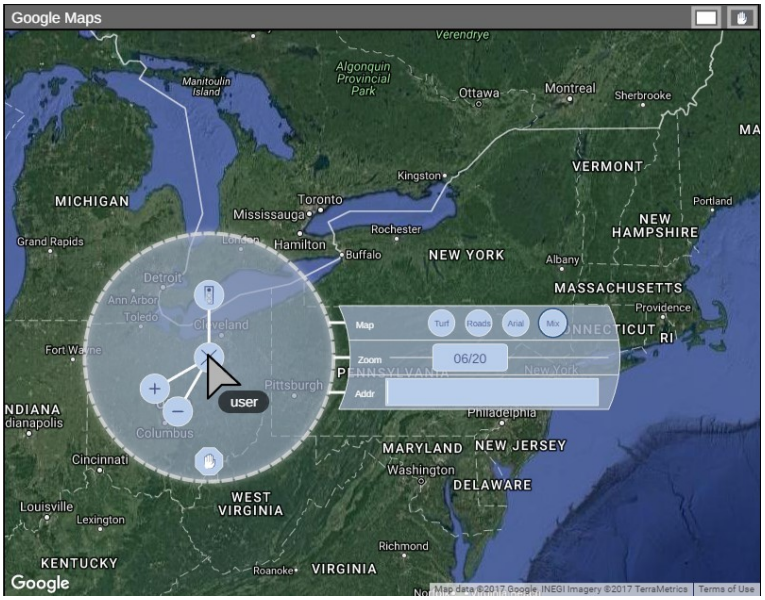
Dále autorem byly zvoleny dva widgety a byla provedená jejich detailnější analýza. Díky teto rešerši, autor dostal lepší představu o možnostech a obvyklých způsobech tvoření widgetů pro SAGE2.

První z analyzovaných aplikací je Timer – časoměřič, distribuovaný spolu se SAGE2. Aplikace umožňuje zadat čas (hodiny, minuty a sekundy), po uplynutí kterého, změní se barva pozadí widgetu a zobrazí se popisek 'END' 1.4. S aplikace lze manipulovat za pomocí SAGE2 kurzoru a klávesnice. Kontextové menu aplikace 1.3 obsahuje slider, pomocí kterého lze nastavit počet minut. Autor však myslí, že widget není moc uživatelské přívětivý: není na první pohled jasné, jak lze zadávat počet hodin a vteřin a navíc aplikace neposkytuje žádný návod k použití. Jenom po analýze kódu autorem bylo zjištěno, že se to dělá stisknutím odpovídajícího tlačítka 'Hours', 'Minutes' nebo 'Seconds' a následným zadáním hodnoty pomocí klávesnici. Takový postup nemusí být zřejmý pro uživatele a není jasné, proč kontextové menu neobsahuje slidery pro nastavení poctu hodin a vteřin. Pomocí analýzy kódu se autor dozvěděl, že aplikace je tvořená pomocí SVG a používá knihovnu d3.js, což odpovídá i většině ostatních widgetů SAGE2.

Ještě jedním analyzovaným widgetem se stal GoogleMaps. Jak je jasné z názvu, aplikace používá GoogleMaps API a umožňuje zobrazovat a nastavovat mapu, což vhledem k neomezenému rozlišení je docela užitečným widgetem pro SAGE2. Aplikace také obsahuje SAGE2 kontextové menu 1.5, ale na rozdíl od widgetu Timer obsahuje popisky a prvky menu jsou dobře srozumitelné. Widgetem lze ovládat nejenom za pomocí SAGE2 kurzoru, ale

[19] http://www.d3.org

Obrázek 1.4: Widget v okamžiku uplynutí času.



Obrázek 1.5: Kontextové menu widgetu GoogleMaps.

i pomocí kurzoru operačního systému. Aplikace je tvořená jedním HTML elementem `<div>`, za obsah kterého se stará integrované GoogleMaps API. Během analýzy kódu daného widgetu, autor zjistil, ze GoogleMaps API už je zahrnuté do SAGE2 API a vývojář ho nemusí importovat zvlášť. Stáčí jenom použít globální proměnnou `google`, která je už inicializovaná objektem `google.maps.Map` z GoggleMaps API.

Výhody nezávislých na sobě widgetů	**Výhody aplikací „vše v jednom“**
Jednoduchá rozšiřitelnost (škálovatelnost)	Jediný styl grafického rozhraní
Lepší udržitelnost	Není nutnost otevírat každý jednotlivý widget zvlášť
Libovolné umístění na ploše	
Žádný z widgetů neovlivňuje jiny	
Jednodušší postup při odhalení chyb a testování	

Tabulka 1.2: Porovnání výhod tvorby nezávislých na sobě widgetů oproti aplikacím „vše v jednom“

1.4.1 Nezávislé widgety oproti aplikacím „vše v jednom“

Během analýzy stávajících widgetů si autor všiml, že mezi existujícími aplikacemi nejsou takové, které by byly určené pro několik účelu najednou. Naopak stávající SAGE2 widgety jsou nedělitelnými a nezávislými na sobě prvky, kde každý z nich slouží pro dosažení jednoho cíle.

Podle porovnání 1.2 a názoru autora bakalářské práce, vytváření jednotlivých a nezávislých na sobě widgetů mnohem lépe, než aplikací, které vykonávají roli „vše v jednom“.

Kapitola 2

Návrh

Kapitola obsahuje návrh pěti widgetů. Čtenář zde najde přehled zvolených technologií, stanovené funkční a nefunkční požadavky, vytvořené nákresy budoucích widgetů.

Jelikož všechny widgety jsou určený pro použití ve stejném prostředí, tak způsob jejich implementace je velice podobný. Tím pádem navržené aplikace mohou mít drtivou většinu nefunkčních požadavků shodnými. Autor si vybral podle jeho názoru nejdůležitější z nich, zejména:

- *Doba odezvy.* Pro stanovení limitů doby odezvy lze použít například klasifikaci Jakoba Nielsena, kterou tento výzkumník popsal ve svém článku [11]. Podle něj existuje tři základní limity a to jsou:
  - 0.1 vteřiny: limit, při kterém uživatel má pocit přímé manipulace s objekty v uživatelském rozhraní
  - 1 vteřina limit, při kterém uživatel má pocit volné navigace bez nutnosti zbytečného čekání na odezvu aplikace
  - 10 vteřin: limit, při kterém uživatel udržuje svoji pozornost na vykonaném úkolu

  S ohledem na výšeuvedené rozmezí je rozumné stanovit následující požadavek: doba odezvy 95% požadavků nesmí být delší než dvě vteřiny a zároveň žádná operace nesmí překročit hranici v deset vteřin, jinak by se pravděpodobně ztratila pozornost uživatele a zájem o další používání aplikace.
- *Ošetření uživatelských vstupů.* Je základním požadavkem libovolného systému. Všechna pole pro uživatelský vstup musí být ošetřená a v případě chybného vstupu uživatel musí být na to upozorněn.
- *Vysoká modularita systémů.* Aplikace by měly být rozdělené na jednotlivé smysluplné části, které jsou mezi sebou nezávislé a slabě provázané.

To přináší dobrou udržitelnost – možnost opravy nedostatků systému bez toho, aby se ovlivnila některá další část [12].

- *Vysoké pokrytí kódu dokumentací.* Je velice pravděpodobné, že na tuto bakalářskou práci budou navazovat další, které budou mít za cíl starat se o vizuální vzhled implementovaných widgetů. Proto je potřeba se zaměřit na napsání dokumentaci ke všem metodám a oblastem kódu, které potřebují podrobnější objasnění.

Při tvorbě widgetů pro SAGE2 je zvykem používat konfigurační soubory pro inicializaci proměnných, nutných ke správnému nastartování aplikace. Z toho nejsou vyloučeny ani widgety, navržené autorem bakalářské práce. Navíc každý vytvořený widget musí být distribuován spolu s uživatelskou příručkou, ve které bude popsán stručný návod pro práci s aplikací.

2.1 Digitální hodinky

Jelikož v moment psání bakalářské práce chybí pro SAGE2 takový základní widget jako hodinky s rozšířenou funkcionalitou: autor se rozhodl takovou aplikaci navrhnout. Kromě zobrazení času a data, by měl widget používat externí veřejné API pro získání informace o aktuálním počasí. Hodinky by také měly obsahovat budík, který lze používat jako připomenutí během práce v SAGE2. Dále by bylo vhodné implementovat automatický přechod ze zimního na letní (a naopak) čas. Zároveň má aplikace umožnit uživateli její nastavení za běhu, například: změna časového pásma, formát data a času atd.

Jedna z hlavních výhod SAGE2 je podpora neomezeného rozlišení a škálovatelnosti. Jak již bylo zmíněno dříve v 1.4: dosáhnout těchto vlastností u widgetů lze pomocí SVG. SVG může obsahovat i HTML kód uvnitř, což přináší doplňkový stupeň volnosti. Proto bylo rozhodnuto udělat widget ve formě SVG. Pro manipulace s SVG a dynamickou vizualizaci dat byl zvolen framework d3.js, který je zahrnut do SAGE2 API a zaměřený pravě pro tyto účely.

Pro získání informace o počasí bylo zvoleno externí API OpenWeatherMap[20] (OWM). OWM API je produktem společnosti VANE[21]. Webová služba sbírá a poskytuje data z tisíců dálkových a zemních senzorů, umístěných po celém světě a trvale se rozšiřuje [13].

API má jeden tarif zdarma a několik předplatných, které se odlišují větším počtem možností a lepšími podmínkami [14]. Pro účely navřené aplikace bude potřeba získávat aktuální počasí pro konkrétně město. K dosažení tohoto cíle

[20] https://openweathermap.org/
[21] http://owm.io/about

Obrázek 2.1: Návrh grafického rozhraní hlavního okna widgetu DigitalClock.

je bezplatný tarif bohatě postačující. Umožní posílat do šedesáti požadavků za minutu, obnovení informace o počasí prochází jednou za dvě hodiny.

Autorem byly vytvořené nákresy grafického rozhraní. Na obrázku 2.1 je vidět jak má vypadat hlavní okno aplikace. Zbytek nákresů lze najít v přílohách B.1 k dané bakalářské práci.

2.2 Instagram klient

Sociální sítě a media jsou v naší době úplně běžné věci, které drtivá většina z nás používá hodně často. Proto autor předpokládá, že se bude hodit pro SAGE2 klient pro jednou z populárních sociálních sítí. Za takovou si autor zvolil Instagram[22] – populární servis pro ukládání a zveřejnění fotografií. Instagram má svoje veřejné Javascript API, které umožňuje vytvářet vlastní aplikaci.

Od roku 2016 však Instagram provedl velké změny v politice poskytování svého API pro vývojáře. Jednou z takových málo příjemných změn je rozdělení API na dva režimy: Sandbox a Live [15]. Práce s API v režimu Sanbox slouží jenom pro vyzkoušení a seznámení s API. Zájemci, kteří chtějí používat API v Live režimu, budou muset veřejně zpřístupnit svoji aplikaci a projít schvalovacím procesem [15], což není zrovna tak jednoduché.

[22] https://instagram.com/

Omezení v Sandbox režimu:

- omezený počet požadavků
- maximálně deset uživatelů a jenom dvacet posledních příspěvků z každého.
- uživatelé musejí být registrované na webu Instagram Developer[23] a dat svůj souhlas na používání informací o jejich účtech pro danou aplikaci.

Autor by měl brát v potaz a počítat se všemi těmito omezeními při tvorbě vlastních widgetů.

Byly stanovené následující funkční požadavky:

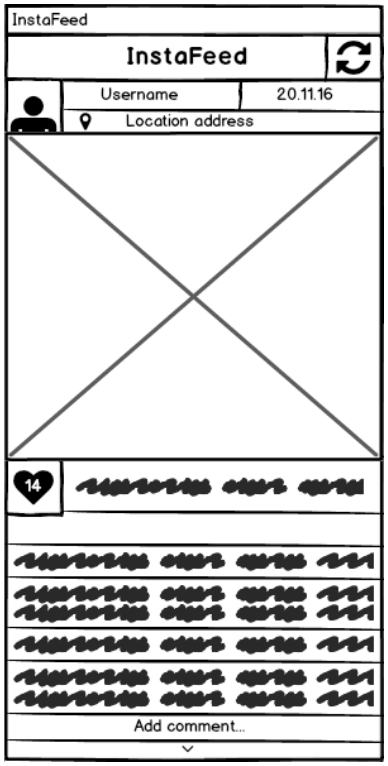
- přihlašovaní za pomocí OAuth
- zobrazení n-příspěvků
- zobrazovat/přidávat komentáře
- přidávat/odebírat „like“ (označení příspěvků, které se líbily)
- tlačítko na obnovení příspěvků
- stránkování
- nastavení: počet zobrazených na stránce příspěvků, interval automatického obnovení)

Jak již bylo zmíněno 2, jedním z nefunkčních požadavků je to, že doba odezvy aplikace v 95% případech nesmí přesáhnout dvou sekund. Jelikož widget je klientem webové služby je důležité zajistit asynchronní volání Instagram API, jinak nezbude možnost dosáhnout příjemné doby odezvy aplikace. Navíc bude potřeba implementovat cache pro již získaná pomocí API data. Takový požadavek je stanoven nejenom kvůli rychlosti práce aplikace, ale i kvůli omezením na počet požadavků k Instagram API.

Autorem bylo navržené následující grafické rozhraní klientské aplikace 2.2. Takový vzhled aplikace je velice podobný na oficiální webový a mobilní klienty Instagram, což přinese více pohodlí uživatelům, kteří jsou zvyklé používat tuto sociální síť.

Uživatel bude moct ovládávat aplikací pomocí SAGE2 kurzoru a klávesnice. Aplikace nebude podporovat SAGE2 kontextové menu, protože autor myslí, že by bylo zbytečné a vedlo by k zmatku uživatele.

[23] https://www.instagram.com/developer



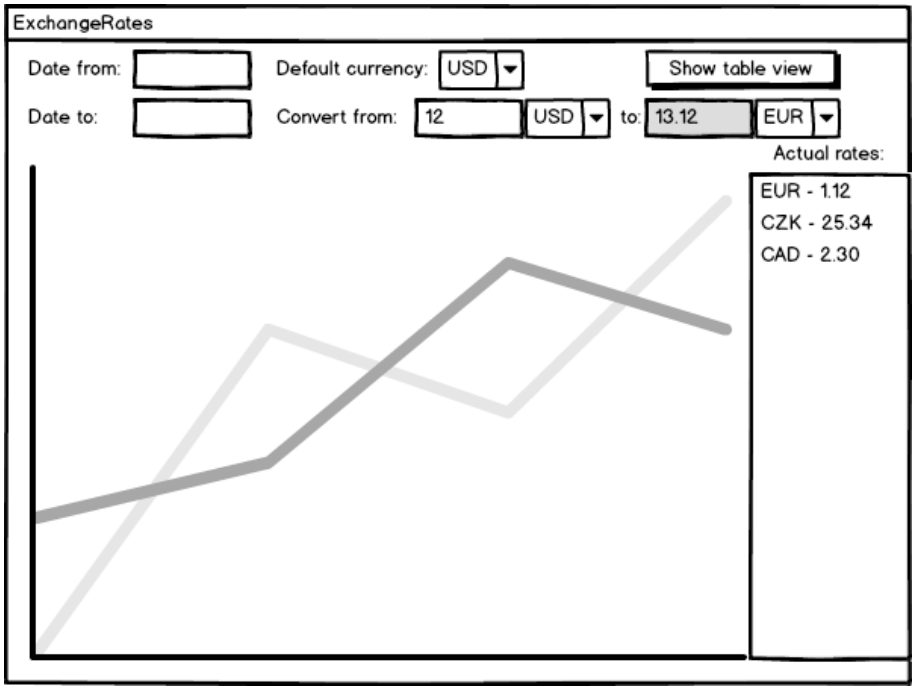
Obrázek 2.2: Nákres grafického rozhraní widgetu InstaFeed.

2.3 Widget pro vizualizaci kurzů měn

SAGE2 se často používá ve výzkumných institucích: sociálních, technologických a financových [1]. Pro poslední z nich autor vyřešil připravit widget, který bude schopen vizualizovat informaci o kurzech v podobě grafů. Aplikace bude získávat data online pomocí integrovaného externího API.

Tento widget by měl splňovat následující funkční požadavky:

- nastavení časového intervalu
- konvertor měn
- napojení na externí API pro získání aktuálních kurzů
- vizuální představení dat ve formě grafů
- výběr základní měny
- možnost přepnutí mezi pohledem s grafem a pohledem s tabulkou, která obsahuje kurzy všech dostupných měn oproti základní



Obrázek 2.3: Nákres widgetu ExchangeRates.

Pro získání dat bylo zvoleno Open Exchange Rates API[24]. Webová služba poskytuje jak aktuální online informaci o kurzech, tak i historická data. API sbírá data z několika zdrojů a pomocí algoritmů je smíchává, čím zaručuje přesně a čestné výsledky [16].

Pro použití API je potřeba registrace. Následně vývojář dostane AppID – unikátní klíč, který se používá pro identifikaci během každého požadavku k webové službě.

Bezplatná verze API má následující funkcionalitu: poskytnutí informace o aktuálních kurzech (obnovuje se každých 30 minut), za libovolný den (od 01.01.1999) a seznam kódu všech podporovaných měn [16].

Nákres widgetu je zobrazen na obrázku 2.3. Zbytek nákresů je v příloze B.2.

2.4 RSS čtečka

RSS (Rich Site Summary) je jednoduchý a moderní způsob informování čtenářů o novém obsahu webových stránek. Funguje tak, že pokud se nějaký web

[24]https://www.openexchangerates.org/

rozhodne podporovat RSS, jeho autor umístí na samostatnou adresu speciálně připravený soubor, kterému se říká RSS feed.

Pro zpracování takových souborů se používá speciální program – RSS čtečka. Aplikace načte feed a zobrazí uživateli náhled a v případě jeho zajmu i celý obsah příslušného článku. RSS čtečky z pravidla umožňují nastavovat více než jeden RSS kanál. To znamená, že uživatel bude mít k dispozici aktuální kontent ze všech svých preferovaných webů na jednom místě. Ještě jednou výhodou takových čteček je to, že umožňuje nastavení časového intervalu, v rámci kterého bude čtečka kontrolovat, zda se obsah webů změnil. V případě nalezení nového kontentu aplikace na to upozorní a nabídne ho ke čtení.

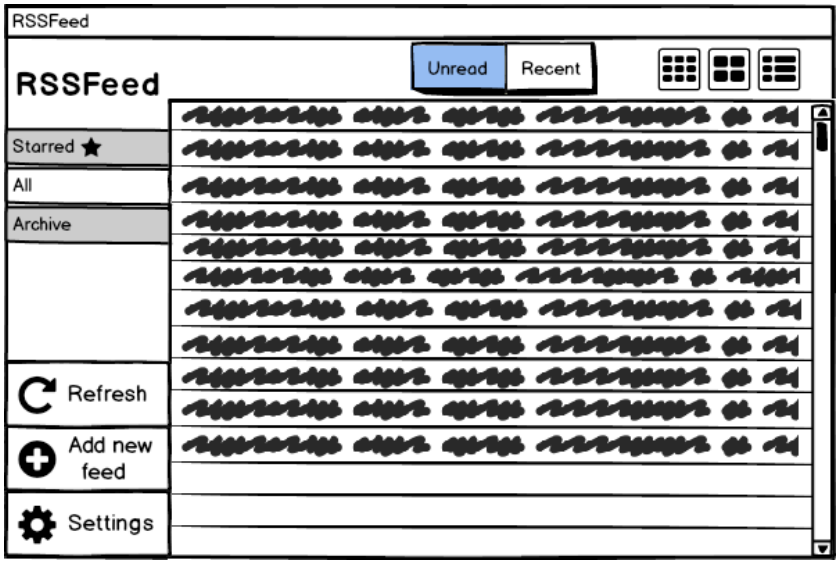
SAGE2 je určen pro vizualizaci různorodé informace a RSS kanály jsou jedním z jejich zdrojů. Tím pádem bylo rozhodnuto navrhnout widget pro čtení RSS feedů. RSS čtečka by měla podporovat následující funkcionalitu:

- zpracování feedů
- možnost označovat příspěvky pro pozdější přečtení
- možnost archivace příspěvků
- zobrazení náhledů a detailů příspěvků
- nastavení časového intervalu pro kontrolu nového obsahu
- tlačítko pro manuální kontrolu nového obsahu
- uložení/načtení přidaných RSS kanálů do/z konfiguračního souboru
- konfigurovatelný vzhled (možnost měnit druh zobrazování náhledů příspěvků: seznam, destičky atd.)
- implementovat možnost vložení adresy RSS z bufferu za pomocí klávesové zkratky

Nákres hlavního okna programu je vidět na obrázku 2.4. Zbytek nákresů se nachází v příloze B.3.

2.5 Konstruktor grafů

Telestěny, které mají schopnost zobrazovat data ve velkém rozlišení, jsou ideálními nástroji pro vizualizaci statistické informace, například ve formě grafů. V souvislosti s tím autor si myslí, že se pro SAGE2 bude hodit widget pro vykreslení grafů na základě dat, získaných z externích souborů.



Obrázek 2.4: Nákres grafického rozhraní hlavního okna widgetu RSSFeed.

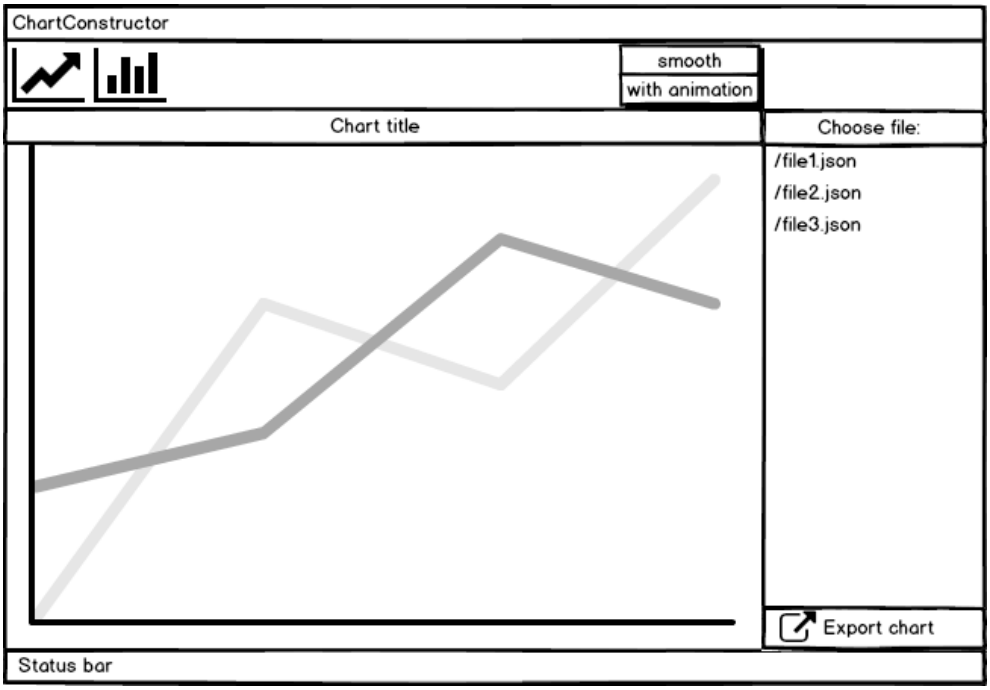
Byly stanovené následující funkční požadavky:

- automatické parsování a analýza dat
- automatické vykreslení grafů za pomocí d3.js
- podpora několika druhů grafů:
  - sloupcový
  - jednočárový (špičatý nebo interpolovaný)
  - vícečárový (špičatý nebo interpolovaný)
- animované zobrazení
- zobrazení legendy grafu
- možnost měnit název grafu
- export grafu do SVG souboru s následující možnosti stažení do lokálního počítače

Podporovanými typy datových souboru byly zvolené:

- JSON[25] (JavaScript Object Notation) – je odlehčený formát pro výměnu dat. Je jednoduše čitelný i zapisovatelný člověkem a snadno analyzovatelný i generovatelný strojově [17].
- CSV (Comma Separated Values) – je jednoduchým formát určeným pro výměnu tabulkových dat. Soubor ve formátu CSV sestává z řádků, ve kterých jsou jednotlivé položky oddělené čárkami [18].

[25] http://www.json.org/json-cz.html



Obrázek 2.5: Nákres grafického rozhraní widgetu GraphConstructor.

Nákres aplikace lze vidět na obrázku 2.5.

Aby navržený widget mohl samostatně vytvářet grafy, uživatel bude povinen dodržovat určitou strukturu datových souborů.

2.5.1 Podporovaná struktura JSON souborů

Pro správnou vizualizaci musí JSON soubor obsahovat pole objektů:

```
[
{ "xKey": value, "yKey": value, ... , "nKey": value}
...
{ "xKey": value, "yKey": value, ... , "nKey": value}
]
```

Klíče objektů mohou mít libovolný název, ale je podstatné pořadí klíčů v prvním objektu (pak v ostatních není důležité). Parser načte klíče z prvního řádku a bude je používat následujícím způsobem:

Pro čárový graf:

- hodnoty prvního klice – hodnoty pro osu X v grafu
- hodnoty ostatních klíčů (1..n) – body pro osu Y
- klíče (1..n) budou navíc použité jako názvy příslušných čar v grafu

Pro sloupcový graf:

- hodnoty prvního klice – hodnoty pro osu X
- hodnoty druhého klice – hodnoty pro osu Y
- zbytek klíčů se ignoruje, protože sloupcové grafy jsou obvykle určené pro zobrazení dat ve formě klíč-hodnota, kde klíč – hodnota na ose X a hodnota – bod na ose Y

2.5.2 Podporovaná struktura CSV souborů

První řádek má obsahovat názvy jednotlivých sloupců dat. Tyto názvy budou použité v legendě čárových grafů, jako označení jednotlivých čar. Hodnoty z prvního sloupce budou použité pro osu X, hodnoty z druhého sloupce – pro osu Y, zbytek sloupců (jestli jsou definované) – pro vykreslení dalších čar v grafu.

2.5.3 Datové typy

Hodnoty pro zobrazení na ose X mohou mít libovolný datový typ. V případě, že je odlišný od číselného, bude možnost použit jenom sloupcový graf, kde hodnoty budou sloužit jako popisky k jednotlivým sloupcům. Jak je zvykem, hodnoty pro zobrazení na ose Y mohou mít jenom číselný datový typ.

V případě použití spatného datového typu, uživatel bude oznámen v informačním poli dole popiskem chyby.

2.5.4 Import datových souborů

Je potřeba umožnit uživatelům nahrávat na server SAGE2 datové soubory a následně je používat pro vykreslení grafů. Třida `SAGE2_App` obsahuje metodu `registerFileListHandler(callback)`, kde `callback` je funkce, která se má vyvolat po prvním načtení nebo následném obnovení adresáře.

Metoda provádí rekurzivní skenovaní adresáře `[SAGE2RootDirectory]/public/upload/`, který slouží pro nahrání veřejně přístupných souborů. Výsledkem metody je informace o obsazených v adresáři souborech: adresa umístění, rozměr, typ atd.

2.5.5 Zvolené technologie

Tento widget se neodlišuje vybranými technologii od ostatních navržených autorem. Aplikace bude tvořená jako SVG a bude používat knihovnu d3.js.

d3.js zahrnuje mnoha užitečných funkcí pro vizualizaci dat, například: několik druhů animací, vytvořeni os grafu, nastavení rozměru a formátu popisků na osách grafu a mnoha dalších.

Kapitola 3

Realizace

Podle zadání bakalářské práce je potřeba realizovat minimálně tři z navržených widgetů. V této kapitole čtenář najde popis zajímavých problémů a jejich řešení, se kterými se autor setkal během vývoje aplikací. Jsou zde také probrané jak obecné detaily, které platí pro všechny aplikace, tak i podrobnosti implementace každého jednotlivého widgetu.

3.1 Digitální hodinky

V kapitole s návrhem (2.1) byly popsané výhody tvoření widgetů pro SAGE2 pomocí SVG. V souvislosti s tím bylo vyřešeno realizovat mechanizmus pro tvoření struktury widgetů na základě SVG, které by se dalo použít s minimálními úpravami u dalších widgetů. Tím pádem vytvořená aplikace se představuje SVG, které vystupuje v roli kontejneru, do kterého budou umístěné různé elementy-potomky, například:

- `rect`, který bude sloužit podobným způsobem jako HTML element `<div>`
- `text` pro všechny popisky a textovou informaci
- `image` pro obrázky a ikonky

Strukturu widgetu DigitalClock s popiskami elementů, které byly použité, lze vidět na obrázku 3.6.

Bylo rozhodnuto udělat layout za pomocí mřížky, která rozdělí hlavní kontejner na `n`-řádků a `m`-sloupců. Tím pádem lze snadno definovat výšku/šířku vnitřních elementu určováním počtu řádku/sloupců. Umožní to jednoduše rozmisťovat prvky uvnitř kontejneru a škálovat aplikaci.

Pro realizaci univerzalního generatoru layoutů byla vytvořena nová třída `LayoutGenerator` 3.2. Grafické rozhraní widgetu je tvořeno polem objektů (potomky rodičovského kontejneru). Princip práce teto třídy je takový, že vývojář nadefinuje JavaScript objekty, které mají specifické atributy, obsahující informaci pro následné vykreslení aplikace. Například počet řádků/sloupců, číslo řádku/sloupce (začátek umístění elementů uvnitř rodičovského kontejneru), text, který bude zobrazen v popisku, obrázek atd. Příklad definice objektu grafického rozhraní:

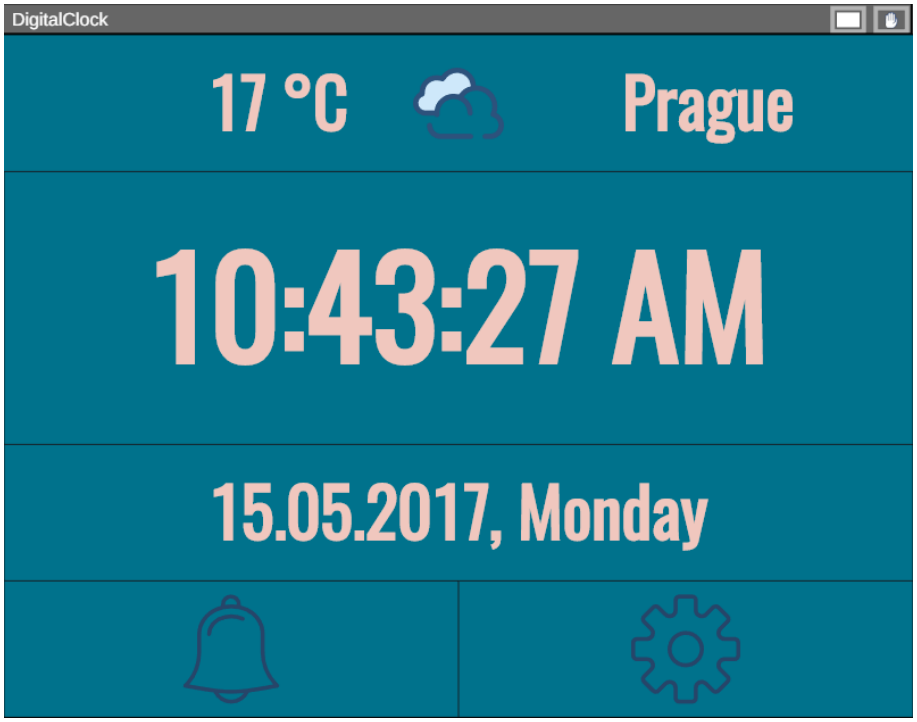
```
this.mainInterface = [
{
  id: "WeatherTemp",
  text: Math.round(weather.main.temp) + " " + this.weatherUnit,
  align: 'start', stroke: "0,50,150", r: 0, c: 0,
  cSpan: 45, rSpan: 20
},
{
  id: "WeatherIcon", image: weatherIcon, r: 0, c: 45,
  cSpan: 10, rSpan: 20
},
{
  id: "Time", text: timeToVisualize, r: 20, c: 0, cSpan: 100,
  rSpan: 40
},
{
  id: "Date", text: dateToVisualize, r: 60, c: 0, cSpan: 100,
  rSpan: 20
},
{
  id: "SettingsButton", command: "true",
  action: this.setupSettingsAction,
  image: this.imgPath + "settings.svg", r: 80, c: 50,
  cSpan: 50, rSpan: 20
}];
```

Listing 3.1: Objekt rozhraní pro hlavní okno aplikace.

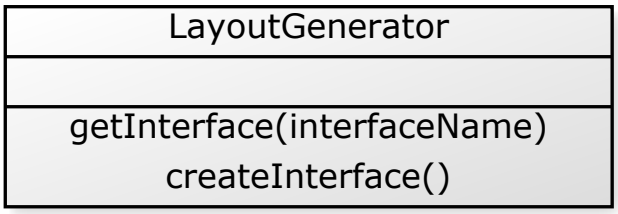
Následně stačí zavolat metodu `getInterface(interfaceName)`, kde `interfaceName` je název rozhraní. Odkaz na vytvořený objekt rozhraní bude uložen do proměnné `currentInterface`. Posledním krokem je vyvolání metody `drawCurrentInterface()` třídy `Render` 3.3. Tato metoda odpovídá za dynamický výkres rozhraní pomocí frameworku d3.js. V rámci metody na základě poskytnuté informace z objektu rozhraní se vypočítají všechny potřebné parametry: výška/šířka každého elementu, jeho poloha, barva pozadí, textový obsah atd. Celý proces je znázorněn na sekvenčním diagramu 3.4 a vzhled vytvořeného widgetu je vidět na obrázku 3.1.

3.1.1 Interakce

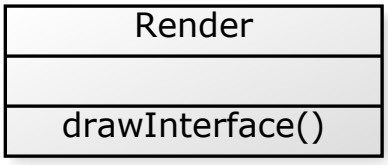
Pro možnost působit na widget pomocí myši a klávesnici bylo potřeba zjišťovat souřadnice kurzoru, a kterému elementu tyto souřadnice odpovídají.



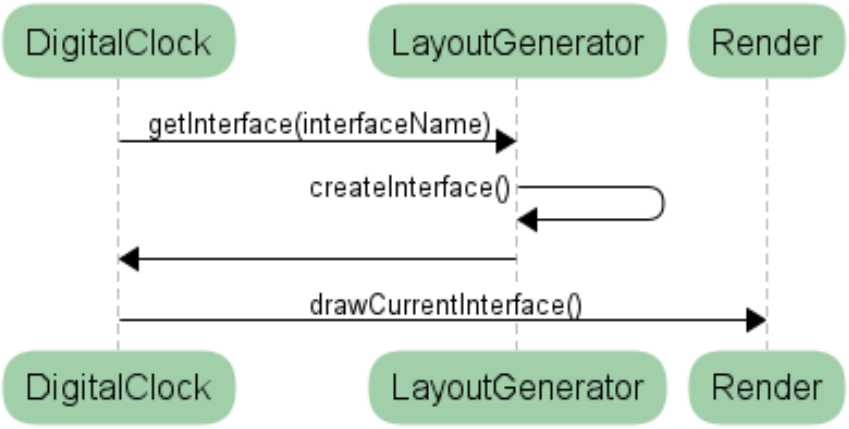
Obrázek 3.1: Vizuální vzhled widgetu DigitalClock.



Obrázek 3.2: Diagram třídy `LayoutGenerator`.



Obrázek 3.3: Diagram třídy `Render`.



Obrázek 3.4: Sekvenční diagram procesu vykreslení grafického rozhraní.

S nalezením souřadnic žádný problém nebyl, jelikož rodičovská třída widgetu `SAGE2_App` obsahuje metodu `event`, která slouží pro zpracování nastalých událostí (kliknutí myši, stisknutí klávesy atd.) 1.3. Všechno co má udělat vývojář: předefinovat tuto metodu a v případě, že nějaká událost nastane, tak se tato metoda automaticky vyvolá a jako její parametry dostane vývojář typ události, pozice kurzoru, čas vzniku události a symbol (v případě stisknutí klávesy).

Byl realizován mechanizmus pro vyvolání akci po kliknutí na element, který lze používat i u dalších widgetů. Stačí vytvořit metodu, která se má zavolat po kliknutí na určitý element a přidat odkaz na tuto metody jako hodnotu parametru `action` objektu příslušného rozhraní. Priklad lze videt na obrazku 3.1 u objektu `SettingsButton`, po kliknutí na který se vyvolá metoda `setupSettingsAction`.

3.1.2 Pole pro textový vstup

SAGE API umožňuje vývojářům jednoduše nadefinovat pole pro uživatelsky vstup v kontextovém menu widgetu. Příklad takového vstupu lze najít v standardním widgetu SAGE2 Timezone 3.5. Po kliknutí pravým tlačítkem myši se zobrazí menu, kde lze zadat název výchozího města pro nastavení časového pásma. Z pohledu autora takový postup není pro uživatele vždycky zřejmý. Navíc není jasné, jak se dá v takovém případě provádět kontroly a případné ošetření chyb uživatelského vstupu.

Jedněmi z funkčních požadavků pro widget DigitalClock jsou implementace budíku a zobrazení aktuálního počasí. Pro účely nastavení času budíku a

definici města pro počasí bylo vyřešeno vytvořit vlastní textové pole pro vstup. Navíc by se tuto funkcionalitu dalo použít i u dalších widgetů.

Všechno co je potřeba udělat pro vytvoření pole pro uživatelský vstup: přidat objektu příslušného grafického rozhraní objekt `inputField`. Seznam podporovaných třídou `Render` 3.3 atributů objektu `inputField`:

name — název pole, který slouží pro jeho identifikaci

action — reference na funkci, která se má vyvolat po aktivaci uživatelského vstupu

min — minimální počet znaků na vstupu

max — maximální počet znaků na vstupu

preLabel/postLabel — objekt pro popisek, který se zobrazí před/za polem pro vstup. Obsahuje následující atributy:

- id — textový identifikátor elementu
- labelText — textový popisek

Pole pro uživatelský vstup je tvořeno za pomocí SVG elementu `<rect>`. Prvek se odlišuje od ostatního rozhraní barvou pozadí a rámečkem. Má to za cíl naznačit uživateli, že je to element, se kterým se dá interagovat.

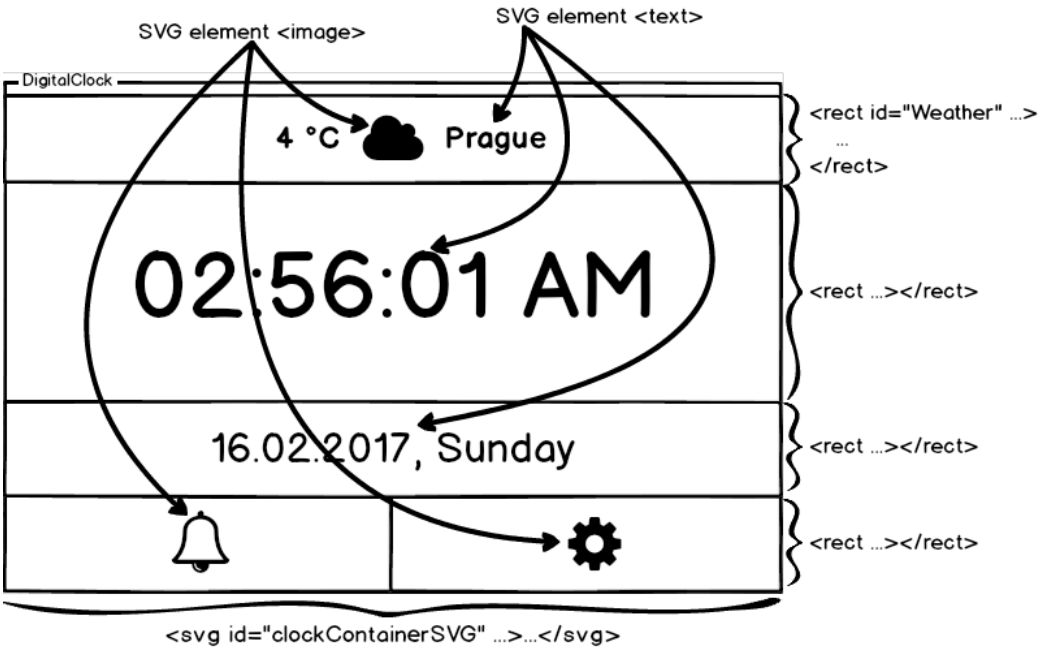
Pomocí hodnot definovaných atributů metoda `Render.drawInterface()` dokáže vykreslit pole pro vstup. Funkce, která je definovaná v atributu `action` aktivuje vstup, bude načítat uvedené uživatelem znaky, ukládat je do proměnné a zobrazovat každý symbol v poli.

3.2 InstaFeed

Grafické rozhraní realizovaného widgetu InstaFeed lze vidět na obrázku 3.7. V rámci implementace Instagram klientu autor narazil na několik problémů při použití Instagram API v Sandbox režimu, kvůli čemu realizace daného widgetu nebyla úplně shodná s představami autora.

První z nich byl autentifikace uživatele. Pro tyto účely Instagram používá OAuth 2.0 protokol. Výsledkem autorizace uživatele by měl být získaný `access_token` – klíč (obecně řetězec znaků), který obsahuje v sobě informaci o identitě a právech uživatele. Klíč se musí představit při každém použití webové služby jako parametr v URL.

Obrázek 3.5: Pole pro uživatelský vstup v kontextovém menu widgetu Timezone.



Obrázek 3.6: Struktura widgetu DigitalClock.

Obrázek 3.7: Grafické rozhraní realizovaného widgetu InstaFeed.

Pro definici skupin *endpoint* (koncových bodů webové služby), ke kterým uživatel má přístup, Instagram API používá termín *scope*. Toto názvosloví pochází ze specifikace OAuth 2.0. Každý uživatel má implicitně nastavený základní scope, který umožní přístup ke ctění informace o uživateli a jeho media kontentu [15]. V případě potřeby použití doplňkového scope (například pro přístup ke komentářům, veřejným datum ostatních uživatelů atd.) je potřeba během získaní `access_tokenu` předat parametr `scope` v URL, obsahující seznam názvů scope, rozdělených mezi sebou za pomocí znaku „+“, například: `https://api.instagram.com/oauth/authorize/?client_id=CLIENT-ID &response_type=token&scope=likes+comments`. Podle dokumentaci [15] Instagram poskytuje následující scope:

- `basic` – pro ctění informace o účtu uživatele a jeho media
- `public_content` – pro ctění informace o veřejných účtech uživatelů a jejich media kontentu
- `follower_list` – pro získání seznamu sledovatelů a sledovaných uživatelem účtů
- `comments` – pro vložení a smazaní komentářů
- `relationships` – umožňuje začít nebo ukončit sledování příslušného účtu
- `likes` – pro přidání nebo odebrání „like“

Typickým postupem pro získání `access_tokenu` jsou následující kroky:

1. Přesměrováni uživatelů na URL-adresu, určenou pro přihlášení. Spolu s tím je potřeba předat takové parametry, jako `client_id` – identifikátor, získaný po registraci aplikace, `redirect_uri` – URL, na kterou bude vracená odpověď, a `scope` – seznam scope, ke kterým uživatel bude mít přístup.

2. Uživatel musí povolit aplikací používat jeho Instagram data.

3. Po přihlášení uživatel bude přesměrován na adresu, uvedenou v parametru `redirect_uri` a`access_token` bude součástí tohoto URL (příklad 3.2).

```
http://your-redirect-uri#access_token=ACCESS-TOKEN
```

Listing 3.2: Odpověď serveru po uspešném získání `access_token` [15]

Jelikož widgety běží uvnitř SAGE2 prostředí není možnost přesměrovat uživatele na stránky Instagram pro přihlášení. Vzhledem k tomuto omezení bylo rozhodnuto přídávat `access_token` do konfigurace aplikace ručně. Zde existuje i další problém: klíč má omezenou dobu platnosti. V takovém případě se proces pro získání `access_tokenu` má opakovat. V nastaveních widgetu byla implementována možnost zjistit datum expirace klíče, aby uživatel věděl, kdy bude muset požádat o nový.

3.2.1 Obalování textu

Délka hlavičky příspěvků nebo komentářů může dosáhnout 300 znaků [15]. To znamená, že během vývoje InstaFeed widgetu byla nutnost brát v potaz implementaci obalování textu, tak aby ten nepřesahoval hranice rodičovského kontejneru a správné se rozděloval na jednotlivé řádky.

V souvislosti s tím byla vytvořená funkce

```
wrap(text, containerId, xCoord, width, height);
```

text text, který má být obalen

containerId textový identifikátor rodičovského kontejneru (obal textu)

xCoord horní levá souřadnice rodičovského kontejneru

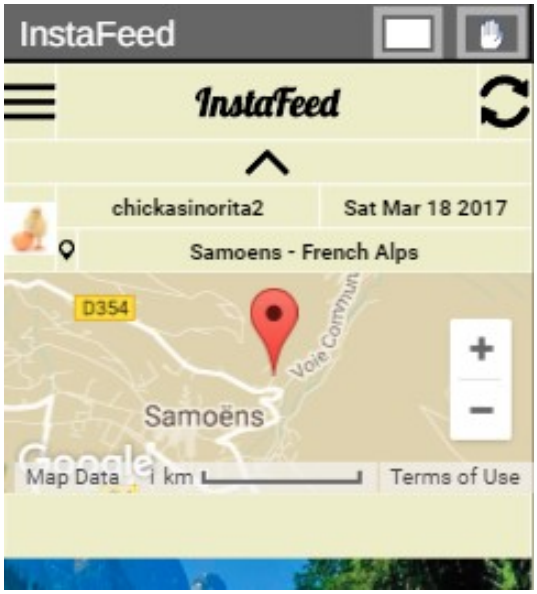
width šířka rodičovského kontejneru

height výška rodičovského kontejneru

Funkce je univerzální a dá se ji použít i při tvorbě dalších widgetů. Princip teto funkce se spočívá v následujícím algoritmu:

1. Text se rozdělí na jednotlivá slova.
2. Provede se kontrola: kolik slov vejde na následující řádek.
3. V případě, že text má přetečení – vytvoří se SVG element `<tspan>`, do kterého se vloží text odpovídajícího řádku.
4. Kroky č. 2 a č. 3 se opakují, dokud se nedokončí kontrola celého vstupního textu.
5. Nastaví se výška rodičovského kontejneru podle počtu vytvořených řádku.

Zároveň pro přidání komentářů k příspěvkům byla použitá implementace textového pole pro uživatelský vstup, která byla realizovaná během vývoje widgetu DigitalClock.



Obrázek 3.8: Zobrazení polohy fotografie pomocí GoogleMaps.

3.2.2 Integrace s GoogleMaps API

Jedna se zajímavých implementovaných vlastností InstaFeed je integrace s GoogleMaps API, což umožňuje zobrazovat na vystavené mapě polohu, kde byla vytvořena fotografie 3.8.

Je mnoho výhod SAGE2 API a jedna z nich to, že jíž v sobě zahrnuje GoogleMaps JavaScript API a poskytuje možnost s ním jednoduše pracovat. Všechno co musí udělat vývojář – importovat globální proměnnou `google` (instance GoogleMaps API třídy `google.maps.Map`). Pak pomocí tohoto objektu lze snadno vytvářet mapy a provádět její konfiguraci.

Však bylo potřeba použít dynamické vytváření HTML `<div>` elementů, do kterých se umisťovali mapy a vypočítávat jejich souřadnice. Navíc po kliknutí na pole s polohou by se mapa měla zobrazovat pod popiskem adresy a ostatní elementy, které se nacházejí níže, posouvat se dolu. Elementy byly vytvořené za pomocí JavaScriptu a DOM. Jejich souřadnice se vypočítávali díky souřadnicím nadřazených elementů.

3.2.3 Cache

Jak jíž bylo zmíněno v návrhu 2.2 je potřeba implementovat cache pro dotažená pomocí API data. V Sandbox režimu není možné nastavit nebo omezit počet příspěvků a komentářů v odpovědi od serveru. V daném režimu Instagram API poskytuje vždycky dvacet posledních příspěvků. Endpoint pro

získání komentářů je jiná, než u příspěvků. Tím pádem se informace o příspěvcích stahuje zvlášť a občas mají velký objem.

Vytvořená cache funguje takovým způsobem, že se při prvním volání API získaná data ukládají do proměnných a dále se používají jenom cachována data. K obnovení cache a opakovanému volaní API dochází jenom v případě ručního obnovení klientu za pomocí příslušného tlačítka či automatického obnovení.

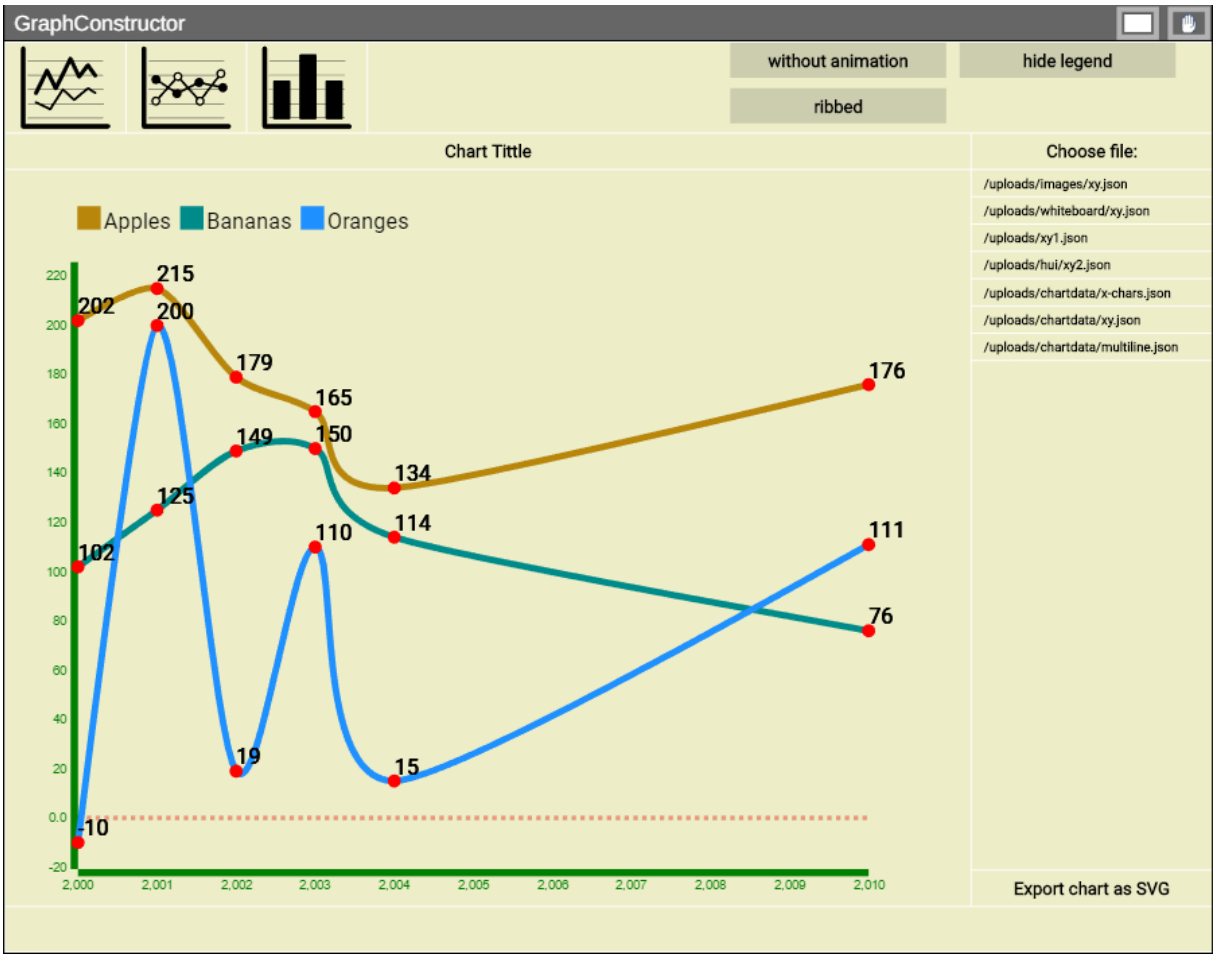
3.3 Konstruktor grafů

Widget GraphConstructor byl realizován podle návrhu 2.5. Všechny funkční 2.5 a nefunkční 2 požadavky byly splněné. Vzhled implementované aplikace lze vidět na obrázku 3.9.

Pro realizaci možnosti měnit název grafu bylo použité dříve implementované pole pro uživatelský vstup 3.1.2.

Rozměr všech prvků rozhraní je závislý na aktuální šířce a výšce okna widgetu, čím zaručuje dobrou škálovatelnost aplikace.

Jelikož widget se představuje v podobě SVG, export grafu do souboru byl implementován poměrně jednoduše. Stačilo vzít kořenový SVG element grafu, přidat mu jako atributy verze a jmenný prostor SVG a následně z toho vytvořit soubor.



Obrázek 3.9: Vzhled implementované aplikace GraphConstructor.

Kapitola 4

Testování

Testování je důležitou části vývoje aplikací. Autor prováděl jak testování jednotlivých částí programů, tak i jejich celků. Většinou se aplikace testovali na lokálním počítači, bylo však několik krát vykonáno nasazení a zkoumání widgetů ve skutečném prostředí v SAGElab.

Kapitola se zabývá především testováním uživatelského rozhraní implementovaných aplikací a zkoumá, zda je intuitivní a srozumitelné pro uživatele.

4.1 Testování během vývoje

Během vývoje bylo potřeba kontrolovat a testovat průběžné výsledky práce. V souvislosti s tím autor několik krát prováděl testování widgetů v SAGElab. Umožnilo to ošetření správného chování aplikací ve skutečném prostředí s velkým rozlišením, její škálovatelnost a náležitou konfiguraci příslušných widgetů.

Také byla otestovaná i shoda s některými nefunkčními požadavky, například dobou odezvy, která během testování v ani jednom z realizovaných widgetů nepřesáhla 1.8 sekund (průměr je 0.8 vteřin), což odpovídá postavenému úkolu.

4.2 Uživatelské testování použitelnosti v SAGElab

Uživatelské testování použitelnosti je jedním z nejdůležitějších druhů testování nejenom pro webové, ale i pro stolní a mobilní aplikace. Výhodou takového testování je to, že dává přehled o tom, s jakými problémy se během používání aplikace setkávají skutečné uživatelé. Je potřeba na to dávat pozor, protože pohled ze strany vývojářů se často liší od uživatelského. I když vývojář má pocit, že použití aplikace je úplně triviální, tak to nemusí platit pro uživatele. Potom to vede k tomu, že ztratí uživatele, který si zvolí jiný produkt.

Díky tomu, že SAGElab zahrnuje laboratoř pro testování použitelnosti, autor měl možnost provést takové testování pro svoje aplikace. SAGElab UX laboratoř má samostatný prostor pro testování, kde je umístěn počítač, několik kamer, který dělají audiovizuální záznamy z různých rakursů a nastroj pro sledování pohybů lidského oka. Dále v SAGElab je oddělená místnost pro pozorovatele, kteří mohou sledovat průběh testování na panelu displejů řízený SAGE2, což jednoduše umožňuje představovat najednou záznamy ze všech kamer, pohyb očí testera a přehrávat zvuk. Takové rozdělení místností je typickým a velmi dobrým řešením pro UX laboratoř, protože odpovídá jedinému z hlavních principů testování použitelností – pohodli testera. Jinak tlak, který participant od přítomných kolem něj pozorovatelů může mít silný vliv na experiment, a výsledky takového testování mohou být odlišné od skutečných.

Podle [19], klasický postup při uživatelském testování použitelnosti obsahuje následující kroky:

1. Analýza cílových skupin a jejich potřeb – na začátku je nutné vědět, kdo jsou uživatelé a co by jim měla aplikace nabídnout. Jedině tak je možné testování postavit tak, aby byly jeho výsledky relevantní a užitečné.

2. Vytvoření scénáře testování – hlavním těžištěm testování jsou úkoly, které jsou předložené jednotlivým testerům. Jedná se o většinou typické akce cílových skupin.

3. Výběr testerů – vzorek účastníků testování by měl odpovídat cílovým skupinám. Rovněž je třeba získat zhruba rovnoměrný podíl pokročilých a nezkušených uživatelů – každá skupina totiž narazí na zcela jiný typ problémů.

4. Samotný průběh testování – pod dohledem zkušeného odborníka na použitelnost plní testeři úkoly dle scénáře testování.

5. Analýza výsledků testování – je třeba setřídit poznatky získané během testování, zanalyzovat je a především vymyslet nejvhodnější řešení vedoucí ke zvýšení použitelnosti aplikace.

4.2.1 Příprava k testování

Příprava k uživatelskému testování použitelností je poměrně velká a odpovědná část celého procesu. Byla provedena kontextová analýza, stanovené uživatelské persony a byl udělán heuristický průchod. Autorem také byly připravené vstupní a výstupní dotazníky a scénáře pro každý z widgetů, které lze najít v přílohách C.

4.2.1.1 Konstrukce person

Jelikož widget-hodinky není specifický, cílovou skupinu pro DigitalClock tvoří libovolné uživatelé SAGE2, z čehož plyne, že personou pro danou aplikaci může být uživatel SAGE2 bez žádných konkrétních omezení nebo vlastností.

Co se týče klientu pro Instagram, je rozumné zvolit jako personu uživatele dané sociální sítě. V takovém případě bude seznámen s webovým a mobilním klientem, znát jejich smysl, základy jejich použití a k čemu jsou určené jednotlivé prvky rozhraní.

Konstruktor grafů je také obecným widgetem, který je určen pro širokou audienci. Personou pro daný program byl zvolen výzkumník, který pracuje se statistickými daty a potřebuje je analyzovat.

4.2.1.2 Kognitivní a heuristický průchody

Kognitivní průchod slouží pro kontrolu průchodu (cesty) uživatele a pravděpodobnosti správné volby na základě znalosti uživatele [20]. V roli expertu byly přihlášený dva studenti FIT ČVUT, kteří měli k dispozici charakteristiky uživatele (persony 4.2.1.1) a samozřejmě testované artefakty (widgety).

Byly stanovené následující úlohy:

- pro DigitalClock:
    1. Nastavte město pro počasí Paříž.
    2. Nastavte budík na libovolný čas.
- pro InstaFeed:
    1. Přidejte komentář.
    2. Přejděte na další stránku.
    3. Zobrazíte mapu prvního příspěvku.
- pro GraphConstructor:
    1. Naimportujte uploads/xy.json.
    2. Zobrazíte sloupcový graf.
    3. Exportujte graf do SVG.
    4. Změňte název grafu.

Experti neměli žádný problém během splnění stanovených úkolů a oba se shodli, že cesty pro dosažení cílů uživatele jsou jednoznačné a všechny prvky vzhledově odpovídají svým účelům.

V rámci heuristického průchodu se kontrolují nedostatky ve všech navštívených částech artefaktu [20]. Kontrola probíhá podle různých seznamů otázek, kterým se říká heuristiky. Nejznámější je Nielsenova heuristika[26].

Pro testování widgetů byly použité heuristiky, které jsou zjednodušením Nielsenových pravidel [20].

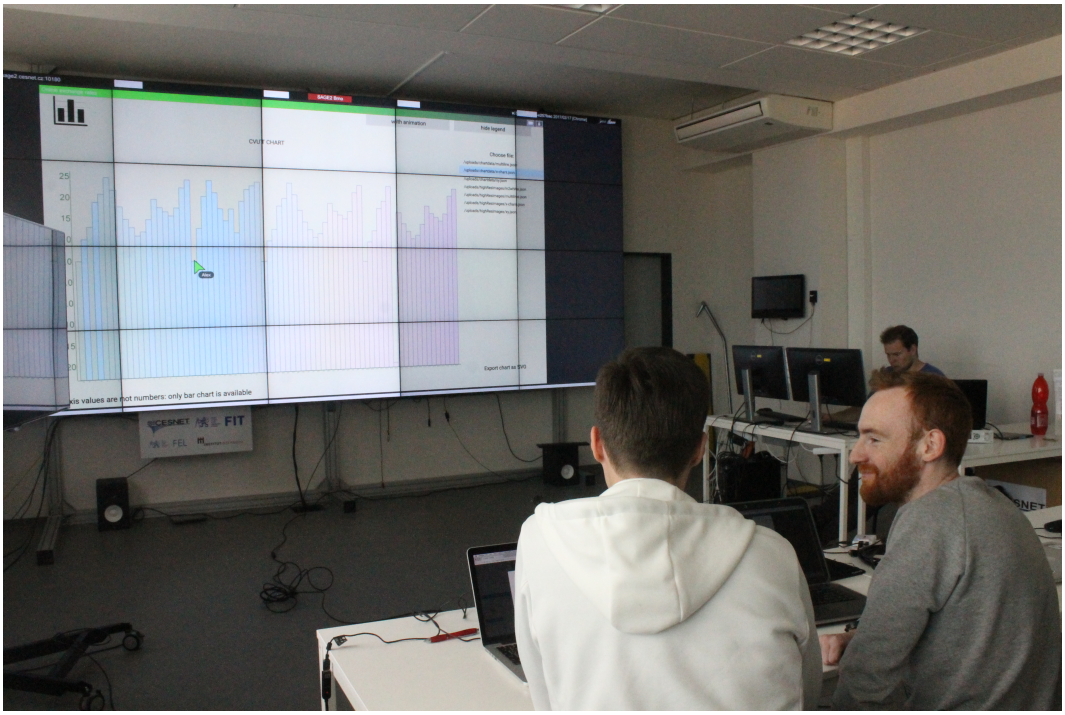
1. Ví uživatel vždycky, kde se nachází (navigace). Ví uživatel vždycky, v jakém je aplikace stavu?
2. Jsou názvy, popisky a jinak používané termíny v souladu s terminologií cílové skupiny?
3. Jsou názvy, popisky a jinak používané termíny srozumitelné cílové skupině? Jsou názvy kategorií (například v menu) voleny s ohledem na srozumitelnost pro cílovou skupinu?
4. Poskytuje každá akce zpětnou vazbu srozumitelnou pro uživatele?
5. Může se uživatel dostat z každé situace? Obejde se takové opuštění bez nutnosti opakovat dlouhou sekvenci akcí?
6. Jsou všechny objekty, akce a jiné prvky vidět, kdykoliv je jich potřeba, aniž by si je uživatel musel pamatovat?
7. Jsou všechny akce, uspořádání a význam konzistentní s očekáváním cílové skupiny? Odpovídá vzhled aplikace a ovládacích prvků cílové skupině?
8. Je plocha zobrazení využita přiměřeně (vzhledem k celkovému vzhledu)?

Skoro na všechny otázky pro všechny widgety experti dali kladnou odpověď. Výjimku tvořil widget DigitalClock, kde na otázku 3, oba experti odpověděli, že název 'WOEID' na stránce nastavení počasí nemusí být pro cílovou skupinu zřejmý.

4.2.2 Průběh testování

Uživatelské testování použitelnosti probíhalo 09.05.17 v laboratoři SAGElab. Testu se zúčastnili tři participanti. Autor bakalářské práce vystoupil v roli moderátora a pozorovatele. Každý tester prošel všemi třemi scénáři. Testování se konalo v místnosti, obvykle určenou pro pozorovatele. Důvodem k tomu bylo to, že panel displejů řízeny SAGE2 se nachází v tomto prostoru. Během tohoto procesu byl nahraný audio-vizuální záznam, který umožnil autorovi udělat podrobnou analýzu a získat přesnější výsledky. Celkové testování trvalo kolem hodiny.

[26] https://www.nngroup.com/articles/ten-usability-heuristics/



Obrázek 4.1: Během uživatelského testování widgetu GraphConstructor.

4.2.3 Analýza výsledků testování

Všechny úkoly byly splněné, ale s některými z nich účastnici testu měli potíže C.10. To nám říká, že vytvořené aplikace jsou zcela použitelné a uživatel během práce je schopny samostatně dosáhnout svých cílů.

Během testování v aplikaci však byly nalezené některá problémová místa z hlediska použitelnosti. Na základě analýzy interpretovaných dat a doporučení od testerů byly provedené opravy uživatelského a grafického rozhraní. Byly vydané nové verze widgetů, testování kterých autor plánuje v červnu 2017.

Závěr

Cílem této bakalářské práce bylo provést analýzu systému SAGE2, jeho API a stávajících widgetů. Návrh vycházel dle zadání z možností a omezení SAGE2 a dostupných technologií. Dále bylo potřeba implementovat nejméně tři vlastní aplikaci.

Všechny body zadání byly splněny a výsledkem práce jsou tři widgety: digitální hodinky, Instagram klient a konstruktor grafů. Autor věří, že se během analýzy povedlo vybrat nejvhodnější technologii a najít dobrá řešení pro vzniklé během realizace problémy.

Aplikace byly nasazené ve skutečném prostředí v SAGElab a kromě funkčního testování, bylo provedeno i testování uživatelského rozhraní. Widgety získali většinou dobré hodnocení od participantů testování, s čehož vyplývá, že již v současné implementaci je řešení použitelné.

V budoucnu by bylo vhodné pokračovat ve vývoje a rozšíření funkcionality realizovaných widgetů. Nejvíc se to týče konstruktoru grafů, kde existuje možnost rozšíření podporovaných datových typů souborů, detailního nastavení grafů atd. Lze také udělat pokus projít schvalovacím procesem Instagam klientu pro přechod do Live režimu a následnou implementaci funkcionality, která tím byla omezená.

Literatura

[1] Marrinan, T.; Aurisano, J.; Nishimoto, A.: *SAGE2: A New Approach for Data Intensive Collaboration Using Scalable Resolution Shared Displays [online]*. [cit. 2017-03-23]. Dostupné z: `http://sage2.sagecommons.org/download/534/`

[2] EVL: *SAGE^{TM} [online]*. [cit. 2017-03-23]. Dostupné z: `http://sage.sagecommons.org/`

[3] Rouse, M.: *Definition – middleware [online]*. TechTarget, červen 2015, [cit. 2017-03-28]. Dostupné z: `http://searchmicroservices.techtarget.com/definition/middleware`

[4] IT-SLOVNIK.cz: *Widget [online]*. [cit. 2017-04-30]. Dostupné z: `https://it-slovnik.cz/pojem/widget/`

[5] IT-SLOVNIK.cz: *API [online]*. [cit. 2017-04-30]. Dostupné z: `https://it-slovnik.cz/pojem/api/`

[6] FIT ČVUT: *OAuth 2.0 [online]*. 2017, [cit. 2017-04-23]. Dostupné z: `https://rozvoj.fit.cvut.cz/Main/oauth2`

[7] EVL: *SAGE2^{TM} [online]*. [cit. 2017-03-23]. Dostupné z: `http://sage2.sagecommons.org/`

[8] Leigh, J.: *SAGE2^{TM}. Birds of a Feather Meeting 2016 [online]*. [cit. 2017-03-24]. Dostupné z: `http://sc16.supercomputing.org/sc-archive/bof/bof_files/bof117s2.pdf`

[9] EVL: *SAGE2 Application API [online]*. červen 2016, [cit. 2017-03-23]. Dostupné z: `https://bitbucket.org/sage2/sage2/wiki/SAGE2ApplicationAPI`

[10] WWW Consorcium: *Scalable Vector Graphics (SVG) 1.1 Specification [online]*. [cit. 2017-03-23]. Dostupné z: http://www.w3.org/TR/2003/REC-SVG11-20030114/

[11] Nielsen, J.: *Response Times: The 3 Important Limits [online]*. Nielsen Norman Group, leden 1993, [cit. 2017-04-10]. Dostupné z: https://www.nngroup.com/articles/response-times-3-important-limits/

[12] FIT VUT: *Požadavky a jejich specifikace [online]*. [cit. 2017-03-20]. Dostupné z: http://www.fit.vutbr.cz/study/courses/PPS/public/pdf/5_2.pdf

[13] VANE: *About [online]*. [cit. 2017-04-24]. Dostupné z: http://owm.io/about

[14] VANE: *Current weather and forecasts collection [online]*. [cit. 2017-04-30]. Dostupné z: https://openweathermap.org/price

[15] Instagram: *Instagram API [online]*. 2015, [cit. 2017-04-14]. Dostupné z: https://www.instagram.com/developer

[16] OXR: *Open Exchange Rates API documentation [online]*. [cit. 2017-04-28]. Dostupné z: https://docs.openexchangerates.org/

[17] Crockforde, D.: *Introducing JSON [online]*. [cit. 2017-04-18]. Dostupné z: http://www.json.org/json-cz.html

[18] Shafranovich, Y.: *Common Format and MIME Type for Comma-Separated Values (CSV) Files [online]*. říjen 2005, [cit. 2017-04-03]. Dostupné z: https://tools.ietf.org/html/rfc4180

[19] interval.cz: *Testování použitelnosti webu bez předsudků [online]*. [cit. 2017-03-03]. Dostupné z: https://www.interval.cz/clanky/testovani-pouzitelnosti-webu-bez-mytu-a-predsudku/

[20] Art Design Institut: *Testování [online]*. [cit. 2017-04-23]. Dostupné z: http://adi.wr.cz/dokuwiki/multimedia/web/e-learning/ucd/predikce/start

Příloha **A**

Seznam použitých zkratek

API Application programming interface

CLI Command Line Interface

ČVUT České vysoké učeni technické v Praze

FIT Fakulta informačních technologií

IT Informační technologie

RSS Rich Site Summary

SAGE Scalable amplified group environment

SSL Secure Sockets Layer

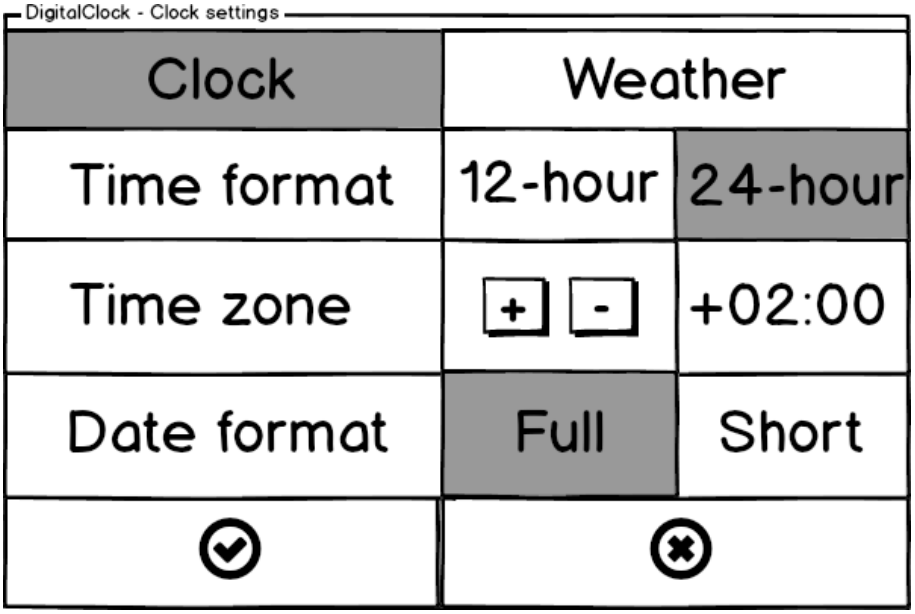
SVG Scalable vector graphics

Příloha B

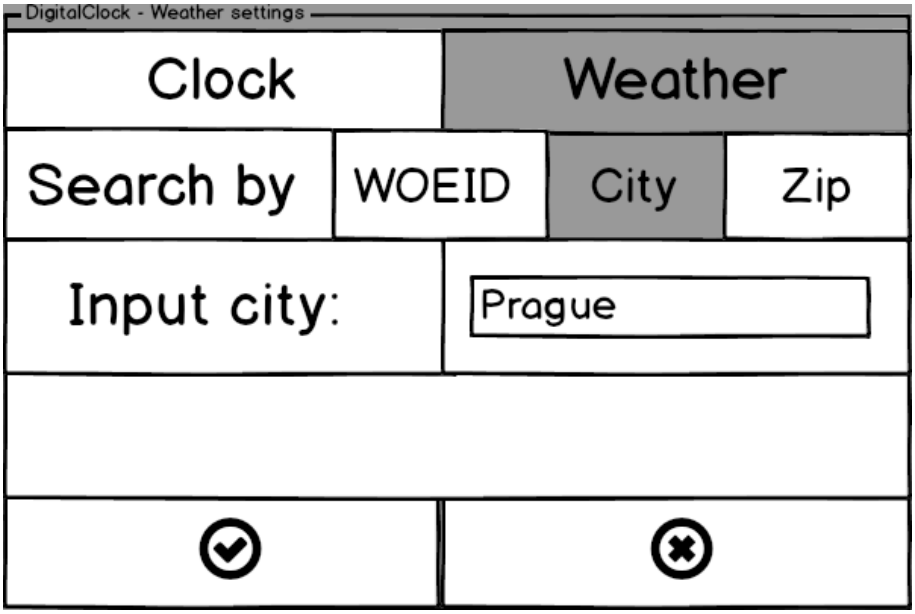
Nákresy navržených widgetů

B.1 DigitalClock

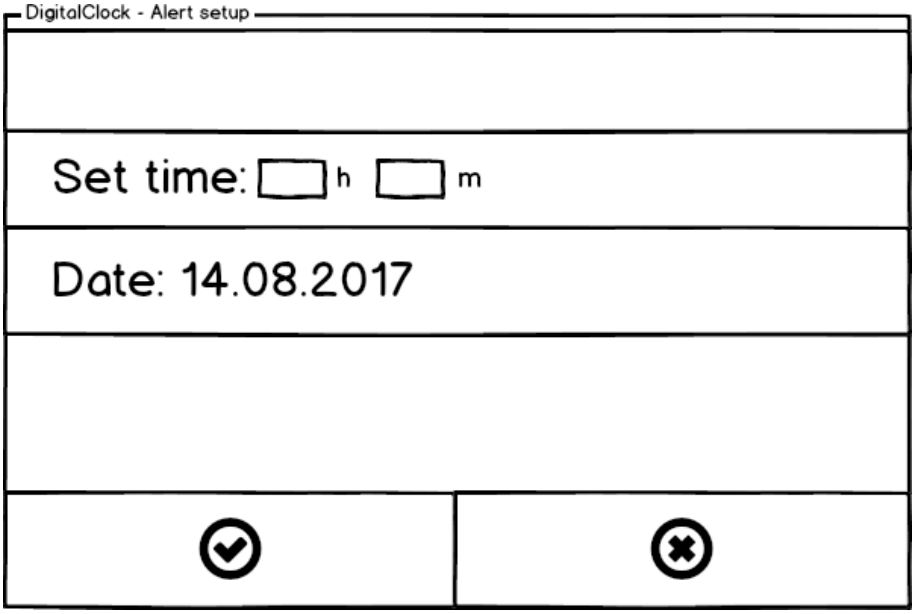
B.1.1 DigitalClock – okno nastavení hodinek



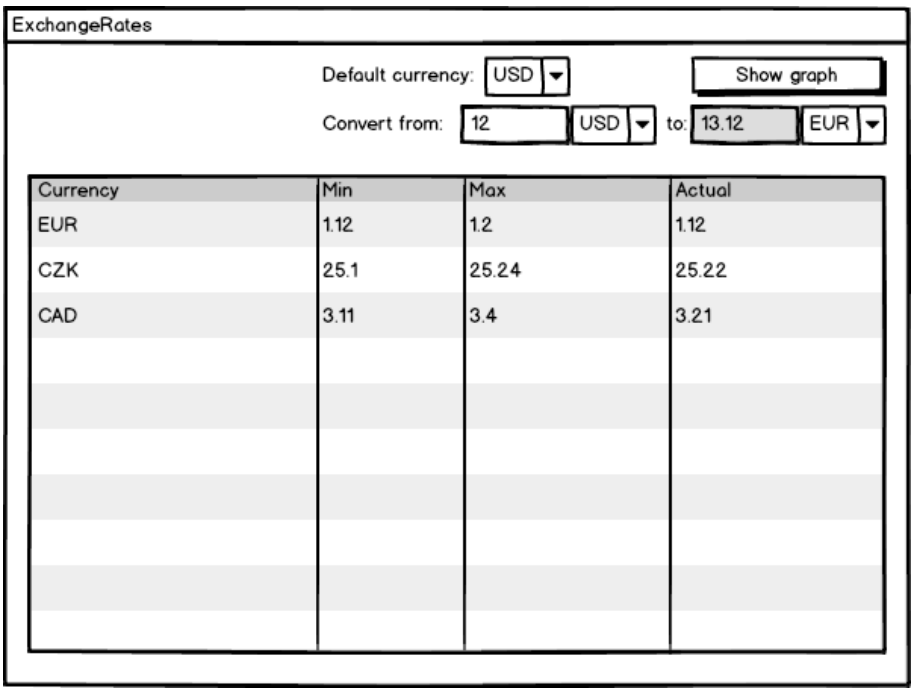
B.1.2 DigitalClock – okno nastavení počasí



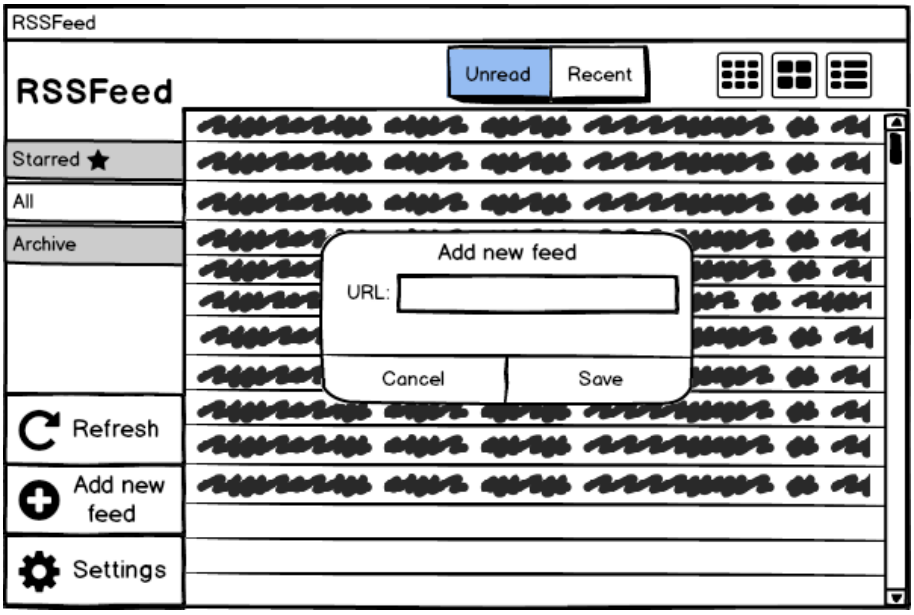
B.1.3 DigitalClock – okno nastavení budíku



B.2 ExchangeRates – okno s informací ve formě tabulky



B.3 RSSFeed – přidání nového feedu



Příloha C

Podklady pro uživatelské testování použitelnosti

C.1 DigitalClock – vstupní dotazník

Připravujete se k testování widgetu DigitalClock pro SAGE2. Vyplňte prosím tento dotazník, který nám pomůže vylepšit výsledky testování.

1. Jak často používáte widgety během práce s SAGE2?

   a) často **b)** občas **c)** zřídka **d)** nikdy

2. Víte co je WOEID (Where On Earth IDentifier) a k čemu slouží?

   a) ano **b)** ne

3. Víte, v jakém časovém pásmu se aktuálně nacházíte?

   a) Ano. Odpověď:

   b) Ne

   c) Nejsem si jistý. Předpokládaná odpověď:

C.2 DigitalClock – scénář pro participanta

Testování použitelnosti widgetu DigitalClock

Úvod:
DigitalClock je widgetem pro SAGE2 a představuje sebou digitální hodinky s možnosti zobrazení času, data a aktuálního počasí. Navíc aplikace nabízí nastavení budíku.

SCENÁŘ

Výchozí stav: Hlavní okno widgetu.

Počáteční nastavení:

- město pro počasí: Praha
- časové pásmo: +00:00 (UTC)
- 12 - hodinový formát času
- zkráceny formát data

Cílový stav: Splnit všechny body

Dneska budeme testovat widget pro SAGE2 DigitalClock. Poprvé spouštíte danou aplikaci a chcete provést její základní nastavení.

1. Nastavte město Brno pro zobrazení počasí.
2. Nastavte 24 - hodinový formát casu.
3. Nastavte časové pásmo, odpovídající Praze.
4. Nastavte budík na hodinu dopředu od aktuálního času.
5. Vraťte se do hlavního pohledu programu.
6. Deaktivujte budík.

C.3 DigitalClock – scénář pro moderátora

Testování použitelnosti widgetu DigitalClock

Text a poznámky pro moderátora jsou uvedené tučným fontem.

SCENÁŘ

Výchozí stav: Hlavní okno widgetu.

Počáteční nastavení:

- město pro počasí: Praha
- časové pásmo: +00:00 (UTC)
- 12 - hodinový formát času
- zkráceny formát data

Cílový stav: Splnit všechny body.

Dneska budeme testovat widget pro SAGE2 DigitalClock. Poprvé spouštíš danou aplikaci a chceš provést základní nastavení. Nemusíš se bát a stresovat. Netestujeme tobě, ale aplikaci. Pokud ti to nevadí, poprosím, abys říkal nahlas, co děláš anebo co ti není jasné.

1. Nastavte město pro zobrazení počasí Brno.
   Uživatel klikne na tlačítko s nastaveními a přejde na položku s nastaveními počasí. Dál si zvolí kritérium, podle které nastaví město (WOEID, PSC, název města) a do příslušného pole zadá hodnotu odpovídající městu Brno.
2. Nastavte 24 - hodinový formát času.
   Uživatel přejde na položku s nastaveními hodinek a změní formát času na „24-hour“.
3. Nastavte časové pásmo, odpovídající Praze.
   Za pomocí tlačítek „+“ a “-“ nastaví časové pásmo na +01:00, což odpovídá pásmu na území ČR.
4. Uložte aktuální nastavení.
   Uživatel klikne na tlačítko se zelenou ikonkou a rozhraní se přepne do hlavního okna.
5. Nastavte budík na hodinu dopředu od aktuálního casu.
   Uživatel klikne na tlačítko se zobrazením budíku a do vstupních políček zadá hodiny a minuty. Zmáčkne tlačítko pro uložení nastavení a bude automatické vracen do hlavního pohledu.
6. Deaktivujte budík.
   Uživatel klikne na tlačítko se zobrazením budíku (když aktivní – je podsvícený zelenou barvou) a klikne na tlačítko pro zrušení nastavení. Následně bude automatické vracen do hlavního okna.

C.4 InstaFeed – vstupní dotazník

Připravujete se k testování widgetu InstaFeed pro SAGE2. Vyplňte prosím tento dotazník, který nám pomůže vylepšit výsledky testování.

1. Jak často používáte sociální síť Instagram?

   a) často **b)** občas **c)** zřídka

2. Jaký Instagram klient používáte častěji?

   a) webový **b)** mobilní **c)** oba dva stejně

3. Máte představu, jak probíhá autentifikace uživatelů v Instagram z technického hlediska (OAuth 2.0 protokol)?

   a) ano **b)** ne **c)** mám mlhavou představu

C.5 InstaFeed – scénář pro participanta

Testování použitelnosti widgetu InstaFeed

SCÉNÁŘ

Výchozí stav: Spouštěný SAGE2 UI.

Nastavení:

- počet příspěvku na jedné stránce: 2
- interval obnoveni příspěvku: 3 minutu
- access_token je prázdný

Cílový stav: Splnit všechny body

Dneska budeme testovat widget pro SAGE2 InstaFeed. Poprve spouštíte danou aplikaci a chcete provést základní nastavení.

Jelikož dany widget musí vynakládat s omezení, kladenými Instagram API v Sandbox režimu je potřeba ručně nastavit access_token pro autentifikaci a autorizaci uživatele. Dolu najdete podrobný popis, jak se to dělá.

Postup pro získání a nastavení access_token:

1. Přepněte se do prohlížeče a přejdete podle ukázaného URL: https://api.instagram.com/oauth/authorize/?client_id=453b9302b22b40e185c68bc4370186d5&redirect_uri=http://localhost:3000&response_type=token&scope=basic+public_content+comments+relationships+likes
2. Přihlaste se s použitím následujících údajů. Uživatel *: tester1*, heslo *: tester1234*
3. Zobrazí se stránka, na které se souhlasíte s použitím Vašich dat a informaci o účtu pro Instagram a následně budete přesměrován na adresu: *http://localhost:3000/#access_token=ACCESS_TOKEN*, kde ACCESS_TOKEN je Vašim hledaným klíčem.
4. Nakopírujte parametr a vložte ho jako hodnotu klíče „access_token“ do konfiguračního souboru widgetu instruction.json

Pokračujte scénářem:

1. Nastavte zobrazeni pouze ječného příspěvku na stránku
2. Nastavte časový interval pro aktualizaci na jednu minutu.
3. Přidejte „like“ k prvnímu příspěvku.
4. Zobrazíte následující příspěvek.
5. Přidejte libovolný komentář.
6. Zobrazíte polohu prvního příspěvku na mapě.
7. Proveďte aktualizaci příspěvku.

C.6 InstaFeed – scénář pro moderátora

Testování použitelnosti widgetu InstaFeed

Text a poznámky pro moderátora jsou uvedené tučným fontem.

SCÉNÁŘ

Výchozí stav: Otevřený SAGE2 UI.

Nastavení:

- počet příspěvku na jedné stránce: 2
- interval obnoveni příspěvku: 3 minutu
- access_token je prázdný

Cílový stav: Splnit všechny body

Dneska budeme testovat widget pro SAGE2 InstaFeed. Poprve spouštíš danou aplikaci a chceš provést základní nastavení. Nemusíš se bát a stresovat. Netestujeme tobě, ale aplikaci. Pokud ti to nevadí, poprosím, abys říkal nahlas, co děláš anebo co ti není jasné.

Jelikož daný widget musí vynakládat s omezení, kladenými Instagram API v Sandbox režimu je potřeba ručně nastavit access_token pro autentifikaci a autorizaci uživatele. V daném dokumentu najdeš podrobný popis, jak se to dělá.

Postup pro získání a nastavení access_token:

1. Přepněte se do prohlížeče a přejdete podle ukázaného URL: https://api.instagram.com/oauth/authorize/?client_id=453b9302b22b40e185c68bc4370186d5&redirect_uri=http://localhost:3000&response_type=token&scope=basic+public_content+comments+relationships+likes
2. Přihlaste se s použitím následujících údajů. Uživatel *: tester1*, heslo *: tester1234*
3. Zobrazí se stránka, na které se souhlasíte s použitím Vašich dat a informaci o účtu pro Instagram a následně budete přesměrován na adresu: *http://localhost:3000/#access_token=ACCESS_TOKEN*, kde ACCESS_TOKEN je Vašim hledaným klíčem.
4. Nakopírujte parametr a vložte ho jako hodnotu klíče „access_token“ do konfiguračního souboru widgetu instruction.json

Pokračujte scénářem:

1. Nastavte zobrazeni pouze ječného příspěvku na stránku

Uživatel přejde do menu aplikace a v poli „Posts on page“ zadá hodnotu: 1

2. Nastavte časový interval pro aktualizaci na jednu minutu.

Uživatel se zůstane v menu aplikace a v poli „Update posts interval“ zadá hodnotu: 1

3. Přidejte „like“ k prvnímu příspěvku.

Uživatel klikne na tlačítko s černou ikonkou „srdce“.

4. Zobrazíte následující příspěvek.

Uživatel klikne na tlačítko s šipkou dolu, které se nachází pod posledním příspěvkem na stránce.

5. Přidejte libovolný komentář.

Uživatel klikne na tlačítko „Add comment“. Aktivuje se pole pro uživatelský vstup, do kterého uvede text komentáře. Po zmáčknutí klávesy Enter: komentář se uloží.

6. Vraťte se k předchozímu příspěvku.

Uživatel klikne na tlačítko s šipkou nahoru, které se nachází nad prvním příspěvkem na stránce.

7. Proveďte aktualizaci příspěvku.

Uživatel klikne na tlačítko pro aktualizaci příspěvku, které se nachází v pravém horním rohu stránky.

C.7 GraphConstructor – vstupní dotazník

Připravujete se k testování widgetu GraphConstructor pro SAGE2. Vyplňte prosím tento dotazník, který nám pomůže vylepšit výsledky testování.

1. Mate nějaké zkušeností s tvorbou grafů?

   a) ano **b)** ne

2. Víte, jak v SAGE2 lze nahrávat uživatelské soubory a do kterého adresáře se mají ukládat?

   a) ano **b)** ne

3. Používáte animaci pro zobrazení grafů?

   a) ano **b)** ne **c)** občas

C.8 GraphConstructor – scénář pro participanta

Testování použitelnosti widgetu GraphConstructor

Text a poznámky pro moderátora jsou uvedené tučným fontem.

SCENÁŘ

Výchozí stav: Hlavní okno widgetu.

Cílový stav: Splnit všechny body.

Dneska budeme testovat widget pro SAGE2 GraphConstructor. Poprvé spouštíte danou aplikaci a chcete provést základní nastavení.

1. Naimportujte do widgetu soubor *uploads/chartdata/multiline.json*.
2. Zobrazíte sloupcový graf.
3. Uložte obrázek s grafem do lokálního počítače.
4. Zobrazíte interpolovaný graf s popisky, ale bez animace a legendy.
5. Zvolte soubor */uploads/chartdata/xy.json*. Je jasné, proč lze vykreslit jenom sloupcový graf?

C.9 GraphConstructor – scénář pro moderátora

Testování použitelnosti widgetu GraphConstructor

Text a poznámky pro moderátora jsou uvedené tučným fontem.

SCENÁŘ

Výchozí stav: Hlavní okno widgetu.

Cílový stav: Splnit všechny body.

Dneska budeme testovat widget pro SAGE2 GraphConstructor. Poprvé spouštíš danou aplikaci a chceš provést základní nastavení. Nemusíš se bát a stresovat. Netestujeme tobě, ale aplikaci. Pokud ti to nevadí, poprosím, abys říkal nahlas, co děláš anebo co ti není jasné.

1. Naimportujte do widgetu soubor *uploads/chartdata/multiline.json*.
   Uživateli se zobrazí čárový graf.
2. Zobrazíte sloupcový graf.
3. Uložte obrázek s grafem do lokálního počítače.
   Uživatel stiskne tlačítko ‚*Export chart to SVG*' v pravém nižním rohu aplikace a obrázek se uloží do adresáře *[SAGE2dir]public/uploads*. Nasledne je lze stahnout v ‚Media browser' v SAGE UI.
4. Zobrazíte interpolovaný graf s popisky, ale bez animace a legendy.
   Uživatel provede nastavení grafu za pomocí příslušných tlačítek.
5. Zvolte soubor */uploads/chartdata/xy.json*. Je jasné, proč lze vykreslit jenom sloupcový graf?
   V pole s informací dolu lze najit odpověď: *xy.json* neobsahuje číselné hodnoty na ose X, což nedává smysl pro kreslení carových grafů.

C.10 Vyhodnocení a výsledky testování

Výsledné hodnocení uživatelského testování použitelnosti widgetů pro SAGE2

Scénář pro DigitalClock

Splnění úkolů

Tester	Úkol 1	Úkol 2	Úkol 3	Úkol 4	Úkol 5	Úkol 6
1	✓	✓	!	✓	✓	✓
2	✓	✓	✓	✓	✓	✓
3	✓	✓	!	✓	✓	✓
Úspěch	3	3	3	3	3	3
Procent úspěchu	100%	100%	100%	100%	100%	100%

✓ - Splnil ✕ - Nesplnil ! - Splnil s potížemi

Čas splnění (v sekundách)

	Úkol 1	Úkol 2	Úkol 3	Úkol 4	Úkol 5	Úkol 6
Tester 1	11	8	**16**	11	2	6
Tester 2	12	6	14	11	3	8
Tester 3	9	9	**28**	13	2	6
Celkové (průměr)	20	8	**19**	12	2	7

(tučným fontem jsou zvýrazněné časy, které pro daný úkol jsou neočekávané velké)

Hodnocení splnění úkolů

Pro hodnocení splnění úkolů autor použil několik kritérii. Hodnoty ve sloupečku „Shoda očekávaného výsledku“ byly získané z poznámek testerů. Zbytek jsou průměrné hodnoty, které byly přidělené během zpracování a interpretace AV záznamů na základě splnění úkolů účastníků. Hodnoty jsou v rozsahu od 0 do 5 (vetší znamená lepší).

Úkol	Rychlost nalezení	Jednoduchost použití	Shoda očekávaného výsledku	Celkově
1	5	4.8	5	4.9
2	5	5	5	5
3	5	**3.9**	5	4.6
4	5	5	5	5
5	5	5	5	5
6	5	5	5	5

U úkolu číslo 3, kde bylo potřeba nastavit časové pásmo pro Prahu, dva testeři nevěděli, jaké časové pásmo se má použít. Tým pádem se museli vrátit zpět do výpisu času a porovnat s aktuálním nastavením.

Scénář pro InstaFeed

Splnění úkolů

Tester	Ú1	Ú2	Ú3	Ú4	Ú5	Ú6	Ú7
1	✓	✓	✓	!	✓	✓	✓
2	✓	✓	✓	✓	✓	✓	✓
3	✓	✓	✓	✓	✓	✓	✓
Úspěch	3	3	3	3	3	3	3
Procent úspěchu	100%	100%	100%	100%	100%	100%	100%

✓ - Splnil ✕ - Ne ! - Splnil s potížemi

Čas splnění (v sekundách)

	Ú1	Ú2	Ú3	Ú4	Ú5	Ú6	Ú7
Tester 1	9	5	4	**9**	10	6	5
Tester 2	13	8	3	3	13	3	4
Tester 3	11	7	3	3	7	5	5

Celkové	10	7	3	5	10	5	5

(tučným fontem jsou zvýrazněné časy, které pro daný úkol jsou neočekávaně velké)

Čas splnění skoro u všech úkolů je velice dobrý, z čehož plyne, že uživatelské rozhraní bylo navřeno dobře. V tom ale není nic divného. Aplikace má skoro stejný vzhled jako oficiální Instagram klienti. První tester během 4. úkolu měl potíži s nalezením tlačítka pro stránkování, které mu přišlo moc malé.

Hodnocení splnění úkolů

Úkol	Rychlost nalezeni	Jednoduchost použití	Shoda očekávaného výsledku	Celkové
1	5	5	5	5
2	5	5	5	5
3	5	5	5	5
4	**3**	5	5	4
5	5	5	5	5
6	5	5	5	5
7	5	5	5	5

Scénář pro GraphConstructor

Splnění úkolů

Tester	Ú1	Ú2	Ú3	Ú4	Ú5
1	✓	✓	✓	✓	✓
2	✓	✓	!	✓	!
3	✓	✓	!	✓	✓
Úspěch	3	3	3	3	3
Procent úspěchu	100%	100%	100%	100%	100%

✓ - Splnil ✕ - Ne ! - Splnil s potížemi

Čas splnění (v sekundách)

	Ú1	Ú2	Ú3	Ú4	Ú5
Tester 1	5	5	8	9	**13**
Tester 2	7	9	**18**	8	9
Tester 3	5	6	**14**	11	7
Celkové	6	7	13	9	10

(tučným fontem jsou zvýrazněné časy, které pro daný úkol jsou neočekávaně velké)

Dva participanti měli problém s nalezením tlačítka pro export grafu do SVG.

Hodnocení splnění úkolů

Úkol	Rychlost nalezeni	Jednoduchost použití	Shoda očekávaného výsledku	Celkové
1	5	5	5	5
2	5	5	5	5
3	**3**	5	5	4
4	5	5	5	5
5	5	5	4	4.5

Zpětná vazba od testerů

DigitalClock

Pozitivní dojmy

- Dobrý vzhled
- Jasnost ovládacích prvků
- Shoda s oceňovaným výsledkem

Negativní dojmy

- Většina testerů (2 z 3) požádali o rozumnější implementaci nastavení časového pásma.

- Dva testeři se shodli, že popisky tlačítek „Short“ a „Full“, která slouží pro nastavení formátu datu, nejsou úplně jasné. Není zřejmé, jaký formát dostane uživatel ve výsledku.

InstaFeed

Pozitivní dojmy

- Jednoduchost použití
- Má velkou podobu s existujícími mobilním a webovým Instagram klienty
- Možnost zobrazení mapy

Negativní dojmy

- Všichni se shodli, že získání a nastavení access_tokenu je poměrně obtížný proces.
- Jeden z testeru měl potíži s nalezením tlačítka pro zobrazení dalších příspěvků. Podle něj prvek je příliš malý a málo zvýrazněn oproti ostatním prvkům rozhrání.

GraphConstructor

Pozitivní dojmy

- Jednoduchost použití
- Jasnost ovládacích prvků

Negativní dojmy

- Většina testerů měla problém s nalezením tlačítka pro export grafů do SVG. Přijde jim, že se nachází v nečekaném místě (zprava dolu) a je dost malé.

Doporučení

Nabídnuta změna	Důvod	Priorita
DigitalClock		
Popisky s názvy měst, které odpovídají aktuálnímu časovému pásmu (tak jak to je např. v OS Windows)		Vysoká
Náhled výsledného formátu datu v menu s nastavení	Není jasné, jaký bude výsledný formát	Normální
InstaFeed		
Převést aplikaci do Live režimu a implementovat přihlášení	Ruční získání access_tokenu není uživatelské přívětivý proces	Vysoká
Zvýraznit prvky, které slouží pro stránkování	Jeden z testerů měl potíži s nalezením takového prvku na stránce	Nízká
GraphConstructor		
Zvýraznit tlačítko pro export grafu a přemístit nahoru k ostatním ovládacím prvkům	Spatně se odlisuje od ostatních prvků rozhrání a nachází se v neočekávaném místě	Vysoká

C.11 Výstupní dotazník

Odpovězte nám prosím na pár otázek, o tom jaký máte z testování dojem.

1. Přišly vám jednotlivé kroky v scénáře jasné a bezproblémové?

   a) Všechny jasné **b)** Většina jasných **c)** Půl na půl **d)** Většina nejasných **e)** Všechny nejasné

2. Pokud bylo něco nejasné, napište co a proč:

3. Jak jste se orientovali v aplikaci?

   a) Vůbec **b)** Málo **c)** Průměrně **d)** Velmi **e)** Zcela

4. Poznali jste, k čemu jsou určené jednotlivé ovládací prvky?

   a) Vůbec **b)** Málo **c)** Průměrně **d)** Velmi **e)** Zcela

5. Jste spokojeni s tím, jak se vám pracovalo?

   a) Vůbec **b)** Málo **c)** Průměrně **d)** Velmi **e)** Zcela

6. Máte nějaký pozitivní dojem?

7. Máte nějaký negativní dojem?

Příloha D

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `impl` ..... zdrojové kódy implementace
    - `DigitalClock` ..... zdrojové kódy widgetu DigitalClock
    - `SocialMedia` ..... zdrojové kódy widgetu SocialMedia
    - `GraphConstructor` ..... zdrojové kódy widgetu GraphConstructor
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF