VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

KALENDÁŘNÍ APLIKACE V PROSTŘEDÍ WEBASSEMBLY

CALENDAR APP IN WEBASSSEMBLY ENVIRONENT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **MARTIN KONEČNÝ**
AUTHOR

VEDOUCÍ PRÁCE **Dr. Ing. PETR PERINGER**
SUPERVISOR

BRNO 2022

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2021/2022

Zadání bakalářské práce

Student: **Konečný Martin**
Program: Informační technologie
Název: **Kalendářní aplikace v prostředí WebAssembly**
Calendar App in WebAsssembly Environent
Kategorie: Informační systémy

Zadání:

1. Seznamte se s existujícími aplikacemi typu plánovací kalendář a s prostředím *WebAssembly*. Prostudujte algoritmy a datové struktury používané v aplikacích typu kalendář. Seznamte se s datovými formáty a možnostmi synchronizace kalendářních dat.
2. Navrhněte knihovnu potřebných datových struktur a algoritmů pro kalendářní aplikace. Navrhněte aplikaci, která umožní plánování událostí s možností synchronizace dat a práce v "*offline*" režimu. Zvolte vhodnou knihovnu pro grafické uživatelské rozhraní.
3. Implementujte navrženou knihovnu a aplikaci v C++ jak pro desktopové prostředí, tak i pro prostředí *WebAssembly*. Knihovnu i aplikaci řádně otestujte.
4. Zhodnoťte dosažené výsledky a navrhněte možná vylepšení.

Literatura:

- RFC5545: Internet Calendaring and Scheduling Core Object Specification (iCalendar) IETF, 2009.
- Domovská stránka projektu *WebAssembly*. https://webassembly.org/
- Další dle pokynů vedoucího.

Pro udělení zápočtu za první semestr je požadováno:

- První dva body zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Peringer Petr, Dr. Ing.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 11. května 2022
Datum schválení: 3. listopadu 2021

Abstrakt

Tato práce se zabývá tvorbou kalendářní aplikace, použitelné jako desktopová aplikace i jako webová aplikace v prostředí *WebAssembly*. Nejprve je provedena analýza již existujících kalendářních aplikací a datových struktur v nich použitých. Dále je vytvořen návrh vlastní aplikace, který obsahuje návrh grafického rozhraní a návrh knihovny datových struktur. Aplikace je následně implementována v jazyce *C++* pomocí nástrojů *QT creator* a *Emscripten* a je provedeno její testování.

Abstract

This thesis deals with the creation of a calendar application usable both as a desktop application and as a web application in the *WebAssembly* environment. First, the existing calendar applications and the data structures used in them are analyzed. Next, the design of the application itself is created, including the design of the graphical user interface and the library of data structures. The application is then implemented in *C++* programming language using the *QT creator* and *Emscripten* tools and then properly tested.

Klíčová slova

Kalendářní aplikace, C++, Qt, WebAssembly, uživatelské rozhraní

Keywords

Calendar application, C++, Qt, WebAssembly, user interface

Citace

KONEČNÝ, Martin. *Kalendářní aplikace v prostředí WebAssembly.* Brno, 2022. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Dr. Ing. Petr Peringer

Kalendářní aplikace v prostředí WebAssembly

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Dr. Ing. Petra Peringera. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Martin Konečný
7. května 2022

Poděkování

Děkuji vedoucímu práce panu Dr. Ing. Petru Peringerovi za jeho rady a pomoc při tvorbě této práce.

Obsah

Kapitola 1

Úvod

Tato práce má za cíl navrhnout a vytvořit kalendářní aplikaci umožňující pomocí jednoduchého uživatelského rozhraní plánovat a zobrazovat události, a to jak v podobě desktopové aplikace, kterou si uživatel uloží do počítače, tak v podobě webové aplikace. Výsledkem by měla být jednoduchá, snadno ovladatelná aplikace s efektivním uživatelským rozhraním, která pro svůj chod nepotřebuje instalaci, vytvoření a správu účtu, ani (v případě desktopové aplikace) přístup k internetu.
Kapitola 2 se věnuje průzkumu několika vybraných kalendářních aplikací a způsobu, jakým zobrazují kalendářní události. Dále je zanalyzován standard *ICalendar* popisující jednotný formát, ve kterém jsou přenášena data mezi kalendářními aplikacemi. Poté je prozkoumáno prostředí *WebAssembly*, které umožňuje vytvářet webové aplikace v programovacích jazycích, které tomu nejsou normálně uzpůsobeny.
Třetí kapitola se zabývá návrhem vlastní aplikace. Jeho součástí je návrh struktury uživatelského rozhraní a popis prvků, které by toto rozhraní mělo obsahovat. Dále obsahuje návrh knihovny datových struktur, která bude sloužit k implementaci daného rozhraní a ke zpracovávání kalendářních dat. Zároveň tato část popisuje technologie a další knihovny, použité při samotné implementaci aplikace.
Čtvrtá kapitola popisuje implementaci aplikace v jazyce *C++* za použití frameworku *Qt* pro tvorbu grafického rozhraní, a nástroje *Emscripten* pro generování kódu v prostředí *WebAssembly*. Popisuje princip fungování aplikace umožňující splnění požadavků zadání. Tato část se také zabývá testováním aplikace.
Na závěr je práce zhodnocena a je navržen další vývoj aplikace.

Kapitola 2

Analýza kalendářních standardů a existujících aplikací

Tato kapitola má za cíl prozkoumat existující řešení v problematice kalendářních aplikací. Je zde popsán standard *ICalendar* sloužící k uchovávání dat o kalendářních událostech, který je používán většinou dnešních kalendářních aplikací. Dále je prozkoumáno prostředí *WebAssembly*, díky kterému je aplikaci možné zprovoznit na webu. Zároveň je v této kapitole posuzováno několik kalendářních aplikací, a je provedena analýza toho, jakým způsobem se uživateli zobrazují kalendářní informace.

2.1 Standard *ICalendar*

ICalendar [1] je standardizovaný formát popisu kalendářních událostí. Je navržen tak, aby obsahoval veškeré potřebné informace o událostech v kalendáři, které je pak možné exportovat a podle nich zobrazit dané události na libovolném jiném kalendáři. Standard slouží pouze k popisu dat, samotnou interpretaci a zobrazení nechává na konkrétním kalendáři. Soubory s tímto formátem jsou obvykle označeny příponou *.ics* nebo *.ical*.
ICalendar byl poprvé definován v roce 1998 organizací *IETF* (Internet Engineering Task Force) jako standard *RFC 2445* [12]. Jeho hlavním cílem bylo zajištění interoperability mezi jednotlivými kalendářními aplikacemi. Tento standard byl poté v roce 2009 upraven společností *Oracle* jako standard *RFC 5545* [14]. Ze standardu byly odstraněny některé již nepotřebné vlastnosti a bylo vyřešeno několik nejednoznačností, které se v originálním standardu vyskytovaly. V roce 2016 byl standard *ICalendar* doplněn o další vlastnosti, jako například podpora konferenčních systémů, a vznikl z něho standard *RFC7986* [15].

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//ZContent.net//Zap Calendar 1.0//EN
CALSCALE:GREGORIAN
METHOD:PUBLISH
BEGIN:VEVENT
SUMMARY:Abraham Lincoln
UID:c7614cff-3549-4a00-9152-d25cc1fe077d
SEQUENCE:0
STATUS:CONFIRMED
TRANSP:TRANSPARENT
RRULE:FREQ=YEARLY;INTERVAL=1;BYMONTH=2;BYMONTHDAY=12
DTSTART:20080212
DTEND:20080213
DTSTAMP:20150421T141403
CATEGORIES:U.S. Presidents,Civil War People
LOCATION:Hodgenville\, Kentucky
GEO:37.5739497;-85.7399606
DESCRIPTION:Born February 12\, 1809\nSixteenth President (1861-1865)\n\n\n
 \nhttp://AmericanHistoryCalendar.com
URL:http://americanhistorycalendar.com/peoplecalendar/1,328-abraham-lincol
 n
END:VEVENT
END:VCALENDAR
```

Obrázek 2.1: Obsah kalendářního souboru podle standardu *ICalendar* [1]

Na obrázku 2.1 vidíme příklad kalendáře ve formátu *ICalendar*. Začátek a konec kalendáře je ohraničen pomocí klíčových slov `BEGIN:VCALENDAR` a `END:VCALNEDAR`. Kromě klasických kalendářních událostí rozeznává standard také klíčová slova pro další typy událostí jako `VTODO` pro seznam, `VJOURNAL` pro deník nebo `VTIMEZONE` pro informaci o časových zónách.
Pomocí dalších klíčových slov je specifikována verze standardu (verze 2.0 odpovídá výše zmíněnému standardu *RFC5545*), typ kalendáře (v tomto případě gregoriánský) a identifikace kalendáře, ve které byla daná událost vytvořena. Dále zde může být uvedeno časové pásmo a identifikace uživatele, který danou událost vytvořil.
Kalendář obsahuje jednu událost ohraničenou klíčovými slovy `BEGIN:VEVENT` a `END:VEVENT` (uvnitř událostí se ještě mohou vyskytovat informace o tom, kdy nás má aplikace na danou událost upozornit, ohraničené pomocí `BEGIN:VALARM` a `END:VALARM`). Klíčové slovo `SUMMARY` udává název události a `DESCRIPTION` její podrobnější popis. `UID` udává jednoznačný identifikátor dané události.
Čas začátku a konce události definují klíčová slova `DTSTART` a `DTEND`. V nich je zapsáno datum ve formátu YYYYMMDD, poté následuje písmeno T, a za ním se nachází čas ve formátu HHMMSS. Na konci řetězce se ještě může vyskytovat písmeno Z, které značí, že se jedná o čas *UTC*. Před samotným řetězcem lze také specifikovat časové pásmo v podobě `TZID=Europe/Prague` (toto samozřejmě nelze kombinovat s písmenem Z na konci). Řetězec lze také zapsat i jako datum bez času. V takovém případě se použije klíčové slovo `VALUE=DATE`.
Kromě začátku a konce události standard ještě podporuje atribut `DTSTAMP`, který udává, kdy byla daná událost vytvořena, a `LAST-MODIFIED`, který udává, kdy byla naposledy mo-

difikována. Události ještě také mohou obsahovat klíčové slovo `SEQUENCE` udávající, kolikrát byla daná událost modifikována.
Klíčové slovo `RRULE` stanovuje pravidla pro opakující se události. V něm je pomocí dalších klíčových slov jako `FREQ`, `INTERVAL` nebo `BYMONTH` přesně stanoveno, ve které dny se má událost opakovat. Můžeme zde stanovit pravidla pro opakování události v konkrétní dny v měsíci nebo v týdnu, a jdou také stanovit výjimky pro toto pravidlo. Toho lze dosáhnout dvojím způsobem: Pomocí klíčového slova `EXDATE` v popisu události lze stanovit, které instance opakování se mají vynechat. Je možné také vytvořit novou událost se stejným `UID`, která se pomocí `RECURRENCE-ID` odkazuje na datum, ve kterém by se měla daná událost normálně opakovat, a tuto instanci opakování nahradit. Na obrázku je vidět definice pravidla pro opakování, která stanovuje, že se má událost opakovat jednou ročně každý dvanáctý den druhého měsíce v roce.
Událost může dále obsahovat řadu dalších informací jako například potvrzení konání (`CONFIRMED`), souřadnice specifikující místo konání (`GEO`), nebo odkaz na internetovou stránku s podrobnějším popisem (`URL`).

2.2 Prostředí *WebAssembly*

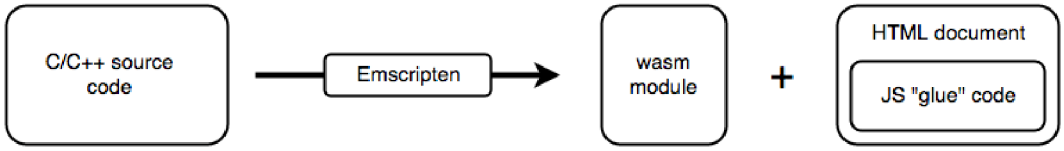
WebAssembly [17] je formát kódu vyvíjený pro tvorbu vysoce výkonných webových aplikací. Jedná se o nízkoúrovňový jazyk podobný assembleru s kompaktním binárním formátem. Tento jazyk poskytuje vyšším jazykům, jako jsou C/C++, cíl kompilace, aby mohly běžet na webu. Můžeme tedy díky němu vytvořit webovou aplikaci v teoreticky libovolném programovacím jazyce bez toho, aby daný jazyk musel být pro běh na webu předem uzpůsobený. Je také navržen tak, aby mohl běžet vedle *JavaScriptu* a umožňoval oběma jazykům pracovat společně.

WebAssembly je vyvíjen skupinou *W3C Community Group*. Jejím hlavním cílem bylo vyvinout *WebAssembly* tak, aby pracoval téměř stejnou rychlostí jako původní kód v desktopovém prostředí, a zároveň dodržoval stejné bezpečnostní zásady jako *JavaScriptový* kód a byl také kompatibilní s ostatními webovými programy.
Virtuální stroj, který zpracovává samotný kód webové aplikace, byl před příchodem *WebAssembly* schopen zpracovávat pouze kód v jazyce *JavaScript*. To většinou nebyl problém, protože *JavaScript* je dostatečně výkonný na to, aby se dokázal vypořádat s většinou nároků vyplývajících z běžného používání webových stránek. Problémy se však začnou vyskytovat, pokud se někdo pokusí přímo na webu provádět některé náročnější činnosti, jako jsou 3D hry, virtuální a rozšířená realita, počítačové vidění, úprava obrázků/videí, atd.
WebAssembly v tomto případě není zamýšlen k tomu, aby jazyk *JavaScript* kompletně nahradil. Místo toho je navržen tak, aby *JavaScript* doplňoval a spolupracoval s ním. Vývojářům webových aplikací tak umožní využívat silné stránky obou jazyků: na jedné straně flexibilitu a expresivitu *JavaScriptu*, včetně přístupu k jeho frameworkům, na straně druhé potom vysoký výkon prostředí *WebAssembly* a jeho možnost sloužit jako kompilační cíl pro ostatní programovací jazyky jako je například *C++*.
Nyní je tedy virtuální stroj, zpracovávající kód pro webové aplikace, schopný načítat a spouštět dva typy kódu - *JavaScript* a *WebAssembly*. Tyto dva typy kódu se mohou navzájem volat podle potřeby - rozhraní *WebAssembly JavaScript API* obaluje exportovaný kód *WebAssembly* funkcemi *JavaScriptu*, které lze volat normálně, a kód *WebAssembly* může

importovat a synchronně volat normální funkce *JavaScriptu*.

WebAssembly funguje na následujících principech: Soubory *WebAssembly* jsou prohlížečem zkompilovány do spustitelného strojového kódu nazývaného *modul*. Modul je bezstavový, a proto jej lze explicitně sdílet mezi jednotlivými částmi prohlížeče. Modul také může deklarovat exporty a importy.
Jako paměť slouží lineární pole bajtů s proměnlivou velikostí, do kterého může *WebAssembly* zapisovat a číst z něj pomocí nízkoúrovňových instrukcí. Data, která z nějakého důvodu nejdou uložit jako série bajtů, jsou místo toho uložena jako záznam v referenční tabulce, ze které se na ně lze odkazovat.
Modul spárovaný se vším, co používá za běhu, včetně paměti, tabulky a sady importovaných hodnot, se nazývá instance. Rozhraní *JavaScript API* poskytuje vývojářům možnost vytvářet moduly, paměti, tabulky a instance. Vzhledem k instanci *WebAssembly* může kód *JavaScriptu* synchronně volat její exportované funkce, které jsou vytvořeny jako běžné funkce *JavaScriptu*. Kód *WebAssembly* může synchronně volat také libovolné funkce *JavaScriptu*, a to formou importu těchto funkcí v rámci modulu *WebAssembly*. Vzhledem k tomu, že *JavaScript* má plnou kontrolu nad tím, jak je kód *WebAssembly* stahován, kompilován a spouštěn, mohli by vývojáři *JavaScriptu* dokonce považovat *WebAssembly* pouze za vlastnost *JavaScriptu* pro efektivní generování vysoce výkonných funkcí.

WebAssembly kód je možné psát přímo, je ovšem rychlejší ho napsat v jiném programovacím jazyce, a do jazyka *WebAssembly* ho přeložit. Pro jazyk *C/C++* k tomu slouží nástroj *Emscripten* [3]. Ten napřed pro kód *C++* zavolá překladač (*clang*), a poté výsledek překladače transformuje na soubor ve formátu *.wasm*. Samotný *WebAssembly* kód ale ve své současné podobě neumí sám přistupovat k rozhraní *DOM*. Dokáže pouze volat *JavaScript*, kterému může předávat primitivní datové typy. Emscripten proto ještě musí k modulu přidat *JavaScriptový* spojovací kód a *HTML*.



Obrázek 2.2: Princip překladu kódu do prostředí *WebAssembly* [16]

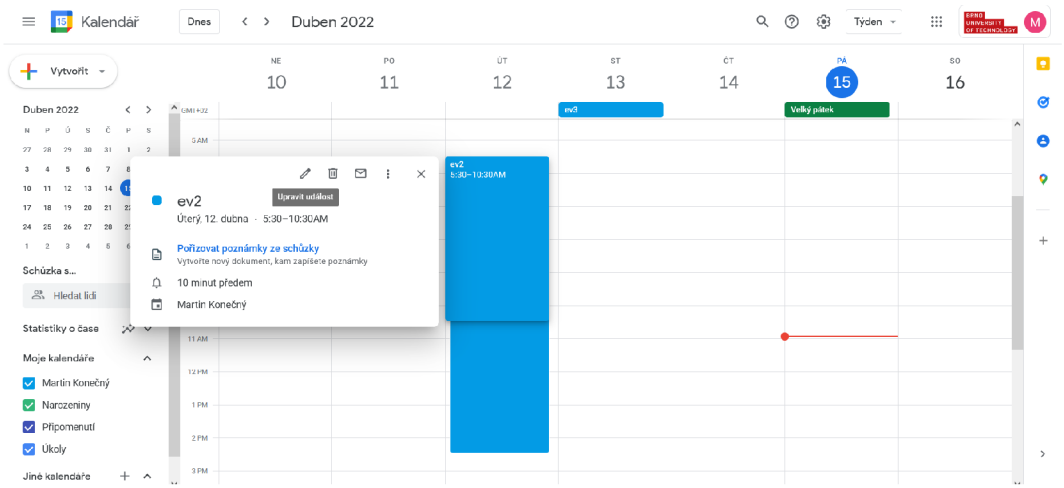
Kromě zprovoznění *DOM* spojovací kód ještě implementuje některé *C/C++* knihovny jako *SDL*, *OpenGL* nebo *OpenAL*. Tyto knihovny jsou implementovány v rámci webových *API*, a proto potřebují *JavaScriptový* kód k propojení s *WebAssembly*.
Vygenerovaný dokument *HTML* načte spojovací kód *JavaScriptu* a vypíše výstup kódu do prvku *<textarea>*. Pokud aplikace používá *OpenGL*, obsahuje *HTML* také prvek *<canvas>*, který se použije jako cíl vykreslování. Výstup *Emscripten* lze velmi snadno upravit a vytvořit z něj libovolnou webovou aplikaci [16].

2.3 Průzkum vybraných aplikací

Za účelem zjištění principů používaných při tvorbě uživatelského rozhraní bylo prozkoumáno několik běžně používaných kalendářních aplikací. Jelikož tyto aplikace většinou všechny řeší zobrazování kalendářních událostí velmi podobným způsobem, byly vybrány následující čtyři z nich, které sloužily jako hlavní inspirace pro samotnou aplikaci.

2.3.1 Kalendář *Google*

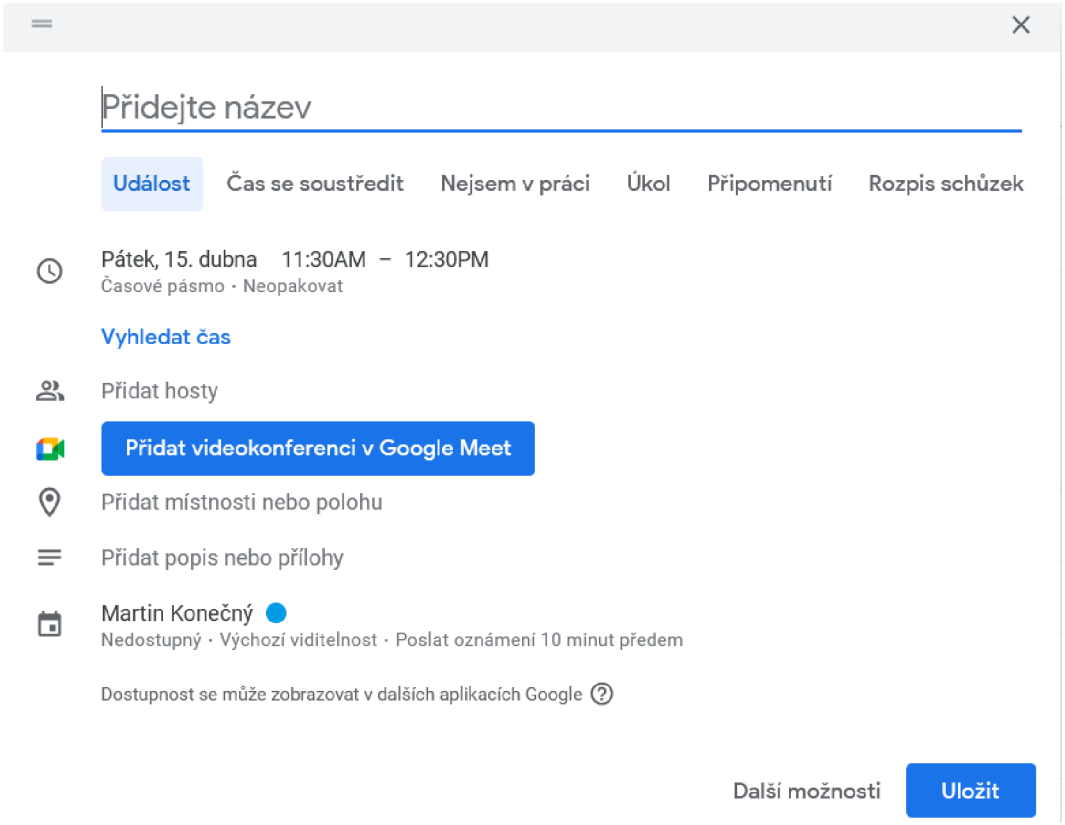
Kalendář *Google* [4] je online plánovací kalendář vyvinutý stejnojmennou společností *Google*. Jeho první beta verze byla zprovozněna 13. dubna 2006, a pro veřejnost byla zpřístupněna v roce 2009. Aplikace existuje jak ve webové podobě, tak jako aplikace pro systémy *IOS* a *Android*.
Kromě klasického plánování událostí, jejich úpravy a nastavování upozornění na ně, podporuje aplikace také další funkce, jako například zobrazování státních svátků nebo narozenin (ke kterým se aplikace dostane díky jejímu propojení s dalšími službami poskytovanými firmou *Google*), nebo zvaní ostatních uživatelů na události.



Obrázek 2.3: Týdenní zobrazení Kalendáře *Google*

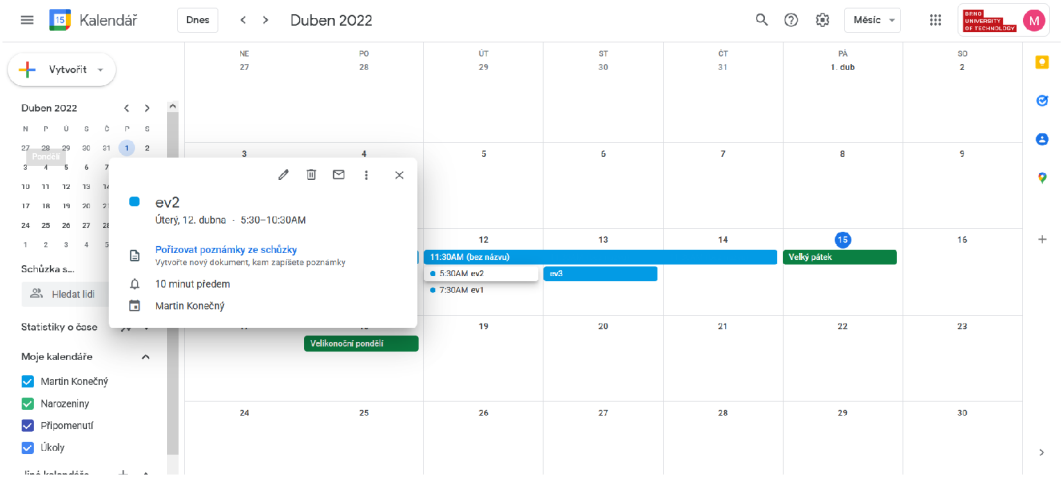
Ve výchozím zobrazení zobrazuje kalendář *Google* události v aktuálním týdnu. V horní liště nad samotnou tabulkou událostí je vyznačen aktuální rok a měsíc. Lišta také obsahuje možnost přepínat mezi týdny nebo se přepnout na aktuální týden. Dále lišta obsahuje nápovědu, odkaz na nastavení, možnost vytisknout aktuální kalendář a možnost přepínat mezi různými režimy zobrazení. Kromě klasického denního, týdenního, měsíčního a ročního zobrazení obsahuje aplikace také možnost čtyřdenního zobrazení a prostého seznamu událostí bez jejich umístění do kalendáře. Mezi těmito zobrazeními je také možné se přesouvat pomocí klávesových zkratek.
Levá lišta obsahuje tlačítko pro vytvoření nové události. Pod tímto tlačítkem se nachází zmenšené zobrazení měsíce, které funguje jako alternativa k přepínání přes horní lištu: kliknutím na den v zobrazení měsíce se přepneme do zobrazení týdne, ve kterém se daný den nachází, a dvojklikem na něj se přepneme do denního zobrazení. Kromě toho lišta umožňuje filtrovat události pomocí povolování a zakazování jednotlivých kategorií, a nabízí možnost

v kalendáři zobrazit státní svátky. Celou lištu je pro lepší přehlednost možné schovat kliknutím na tlačítko v levé části horní lišty.
Samotné události jsou zobrazovány v tabulce, kde sloupce představují dny v týdnu a řádky představují čas. V horní části tabulky jsou vypsány dny v týdnu a jejich datumy, a je vyznačen aktuální den. Ve výchozím nastavení týden začíná nedělí. Je však možné změnit nastavení tak, aby začínal v libovolný den. V samotné tabulce je pak červenou čarou vyznačen aktuální čas. Jednotlivé události jsou zobrazovány v tabulce pomocí barevných obdélníků, jejichž délka v tabulce odpovídá jejich délce trvání. Události zabírající celý den jsou místo toho umístěny v horním rámečku. Po kliknutí na událost se zobrazí okno s podrobnějšími informacemi o ní, včetně možnosti událost upravit nebo odstranit.



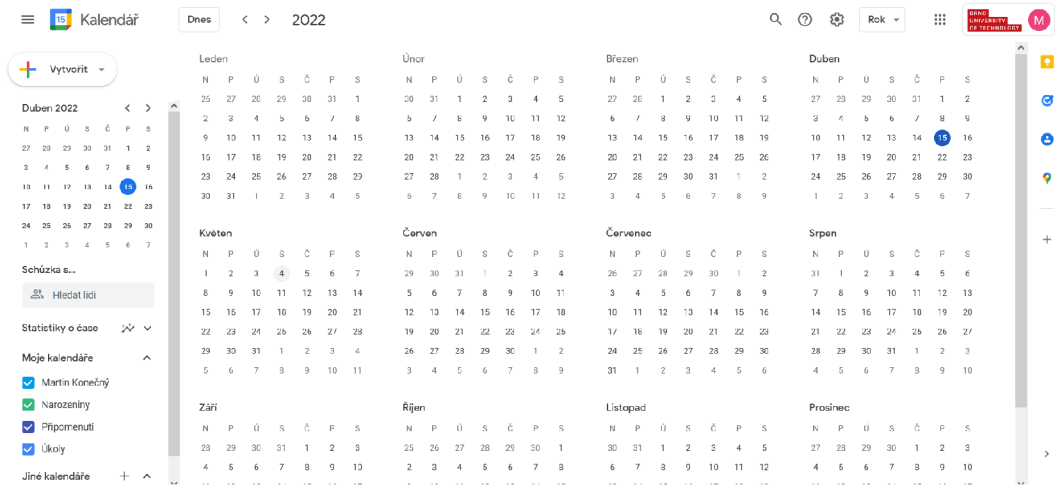
Obrázek 2.4: Vytváření nové události v Kalendáři *Google*

Po kliknutí na políčko v tabulce nebo na tlačítko v levém horním rohu se zobrazí okno pro vytvoření nové události. Na výběr je zde několik možných typů událostí, které však vesměs pouze mění to, jaké informace k nim můžeme přidat. Kromě názvu, popisu a času události k nim například můžeme přidávat místo konání, odkaz na videokonferenci, opakování, nebo čas, kdy chceme být na událost upozorněni. Denní a čtyřdenní zobrazení jsou řešeny stejným způsobem jako zobrazení týdne, liší se pouze počtem zobrazených dní.



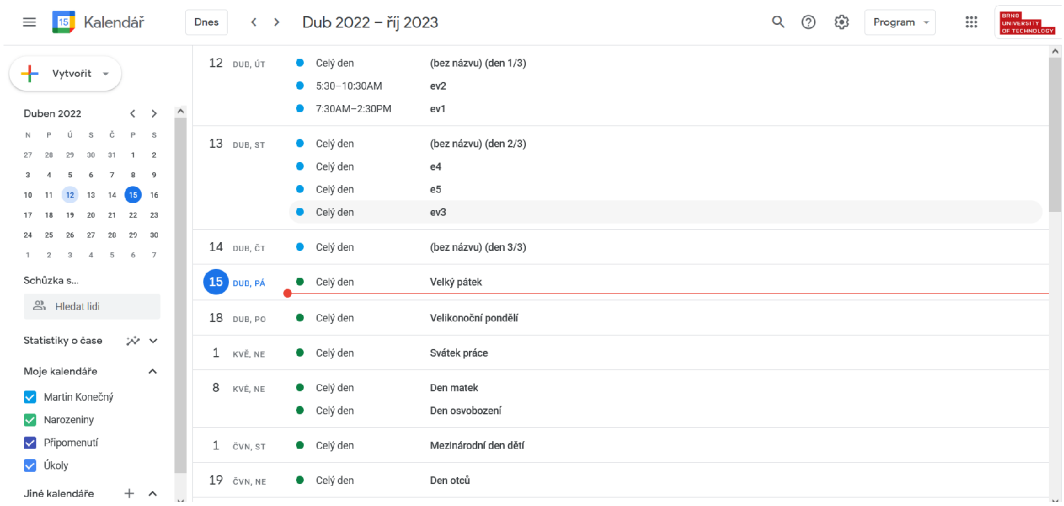
Obrázek 2.5: Zobrazení měsíce v kalendáři *Google*

Zobrazení měsíce má opět podobu tabulky. Horní a levá lišta zůstávají ve stejné podobě, a sloupce jsou opět členěny podle jednotlivých dní v týdnu. V tabulce se již nevyskytuje čas, pouze čísla jednotlivých dní. Aktuální den je vybarvený modře, a dny z předchozího a následujícího měsíce jsou zobrazeny šedě. V políčkách pro jednotlivé dny jsou zobrazované události pro daný den uspořádané do sloupce. V případě, že se v jeden den vyskytuje tolik událostí, že by se do políčka nevešly, zobrazí se pouze první 3 a poté položka "X dalších", po jejímž rozkliknutí se v novém okně zobrazí kompletní seznam událostí pro daný den. Stejně jako v týdenním a denním pohledu je možné kliknutím na místo v kalendáři otevřít dialog pro vytvoření události pro daný den.



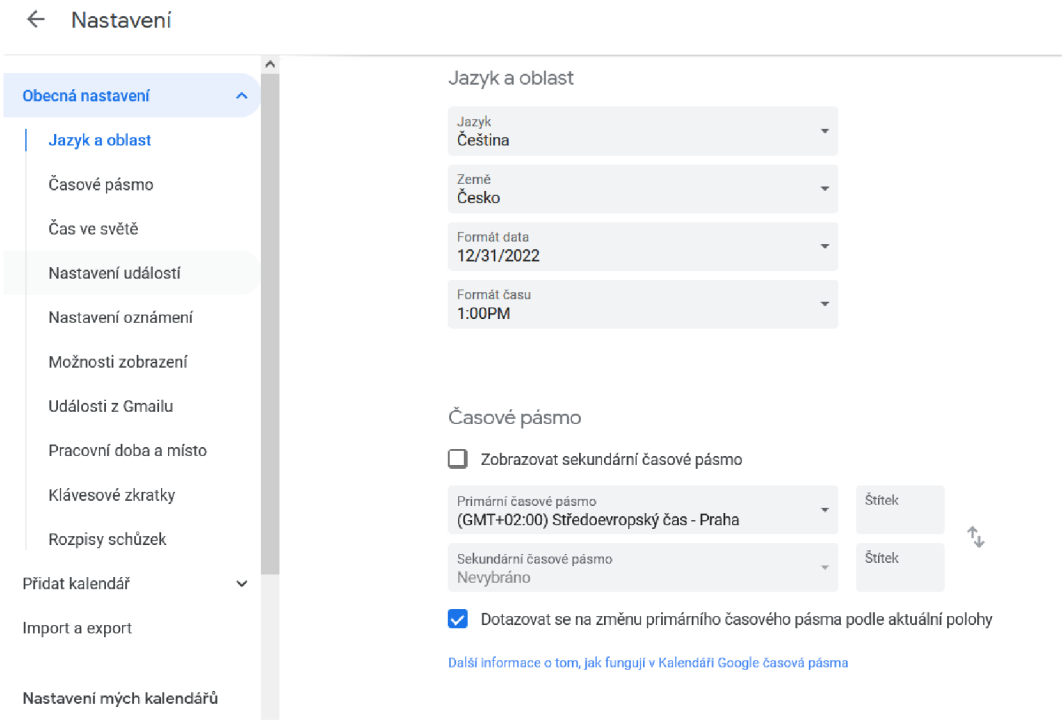
Obrázek 2.6: Roční zobrazení kalendáře *Google*

Roční pohled v této aplikaci plní pouze roli přecházení mezi různými datumy. Události se v něm zobrazují pouze tehdy, když klikneme na konkrétní den. Dvojklikem na některý den se pohled přepne do toho režimu, který byl před vybráním ročního pohledu zvolen naposledy, a pohled se přesune na týden/měsíc, ve kterém se daný den nachází.



Obrázek 2.7: Seznam událostí v kalendáři *Google*

Pohled s názvem *Program* zobrazuje seznam událostí seřazených podle času. V seznamu je vyznačen dnešní den pomocí červené linky a modře vybarveného čísla dne. Seznam začíná jeden den před dneškem a končí po 6 měsících, je ale možné kliknutím na tlačítko načíst další. Události jsou barevně označeny podle toho, do které kategorie patří (uživatelem vytvořené události vs. státní svátky nebo narozeniny, atp.).



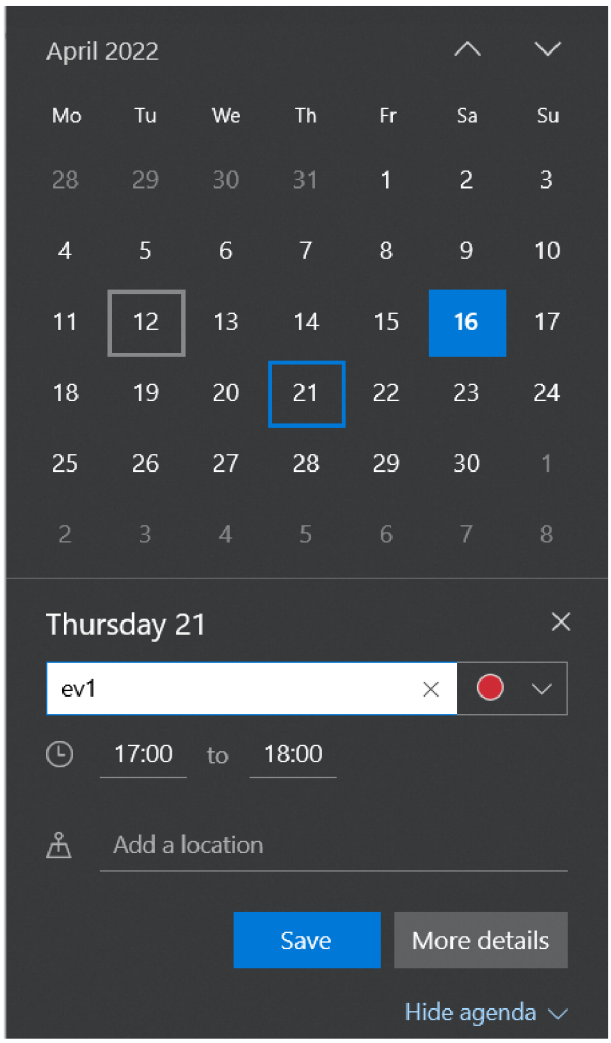
Obrázek 2.8: Nastavení kalendáře *Google*

Kliknutím na nastavení se dostaneme do nového okna s velkým množstvím konfiguračních možností, seřazených do několika kategorií. Je zde například možné nastavit, kterým dnem začíná týden, v jakém formátu se zobrazuje datum, nebo možnost zapnout/vypnout

zvýrazňování víkendů. Je zde také možnost kalendář exportovat a importovat.
Celkově kalendář *Google* obsahuje velké množství funkcí, a dokáže si poradit s celou řadou komplexních problémů (zvaní uživatelů do videokonferencí, synchronizace mezi uživateli, automatické zobrazování narozenin, atd.). Má zároveň relativně přehledné a intuitivní rozhraní, se kterým se dá jednoduše pracovat.

2.3.2 *Windows Calendar*

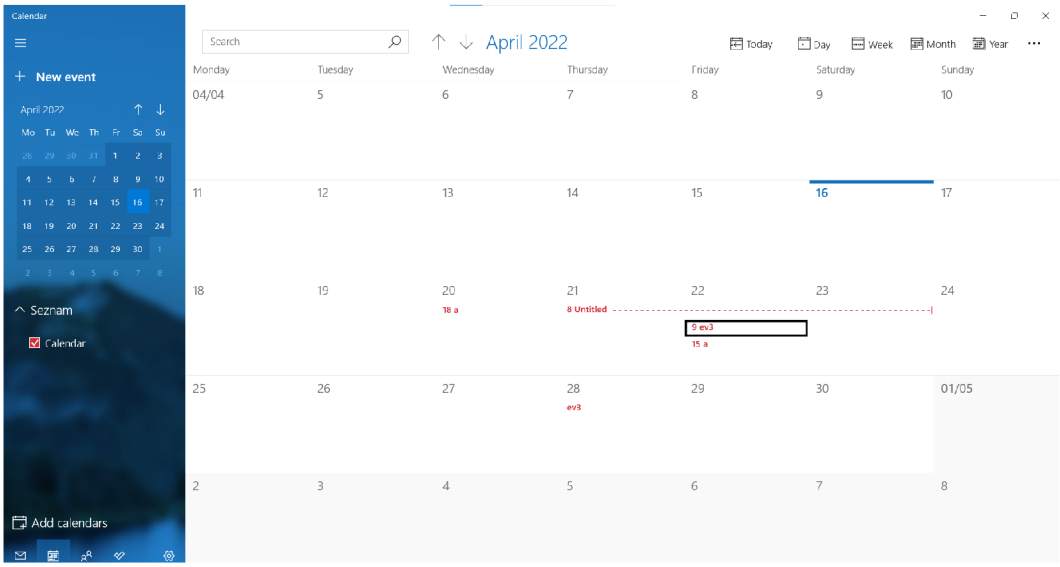
Windows Calendar [18] je aplikace vyvinutá společností *Microsoft* pro systém *Windows*, a je automaticky nainstalována při instalaci tohoto systému. Aplikace umožňuje synchronizaci dat s ostatními často používanými kalendáři, jako je například výše zmíněný kalendář *Google*.



Obrázek 2.9: Zobrazení kalendáře *Windows Calendar* ve spodní liště počítače

Zjednodušená verze kalendáře se nachází ve spodní liště počítače. Ta má podobu zmenšené verze zobrazení měsíce, kde je také možné vytvářet nové události. Pohled však v této

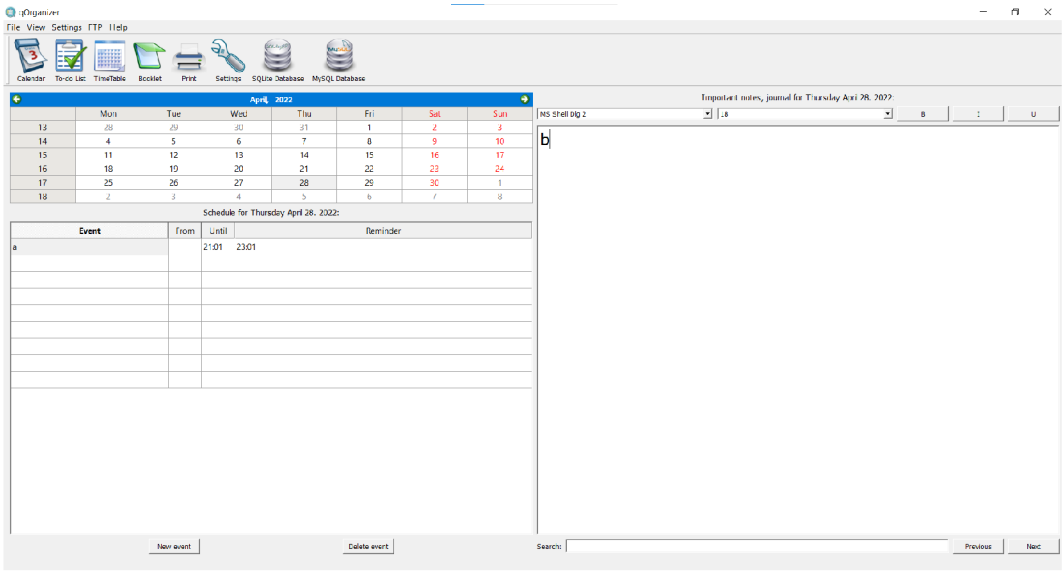
verzi umožňuje zobrazovat pouze události toho dne, který je právě vybraný, a možnosti vytváření nových událostí jsou velmi omezené. Pro plnou funkcionalitu je už potřeba aplikaci řádně otevřít buď kliknutím na její ikonu, nebo kliknutím tlačítko *More Details* při vytváření události ve zjednodušeném pohledu.



Obrázek 2.10: Zobrazení měsíce v kalendáři *Windows Calendar*

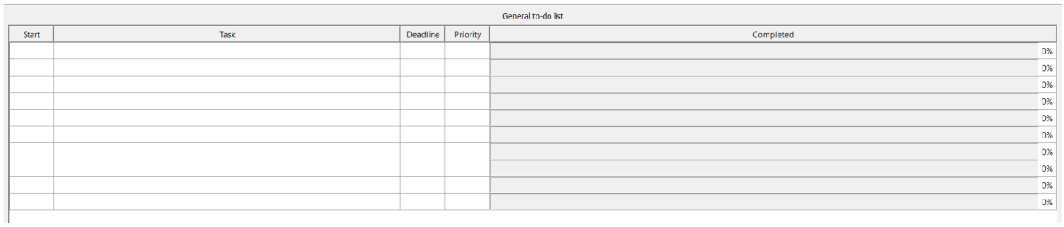
Zobrazení událostí se zde mírně liší od předchozího kalendáře. Místo nutnosti proklikávat mezi jednotlivými měsíci v zobrazení měsíce (i když tato možnost je pořád k dispozici) jsou všechny měsíce součástí jediné tabulky, kterou je možné listovat. Jednotlivé měsíce jsou pak od sebe odlišeny tím, že mají střídavě bílé a šedé pozadí. Jednotlivé události, které netrvají celý den, mají vedle sebe číslo hodiny, ve které začínají. Detaily událostí zde nelze zobrazovat přímo. Je potřeba nejdříve kliknout na políčko dne, kde se zobrazí seznam událostí, a poté se kliknutím na konkrétní událost přesuneme do okna pro úpravu události, kde teprve vidíme všechny detaily. Aplikace také na rozdíl od kalendáře *Google* nepodporuje klávesové zkratky.

2.3.3 *QOrganiser*

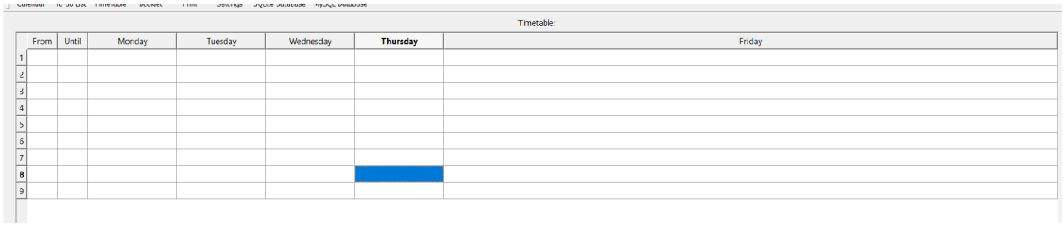


Obrázek 2.11: Základní zobrazení plánovače událostí *QOrganiser*

QOrganiser [8] je desktopový plánovač událostí, který se spíše než na intuitivní rozhraní zaměřuje na větší flexibilitu pro vytváření a organizování událostí. Kromě klasického plánování událostí obsahuje kalendář zápisník s poznámkami k jednotlivým dnům, do kterého je možné vkládat odkazy a obrázky. Kalendář je také možné synchronizovat s databází sql na vzdáleném serveru, a tu sdílet s více uživateli. Zároveň poskytuje další formy zápisu dat: seznam úkolů, týdenní rozvrh, nebo brožuru.



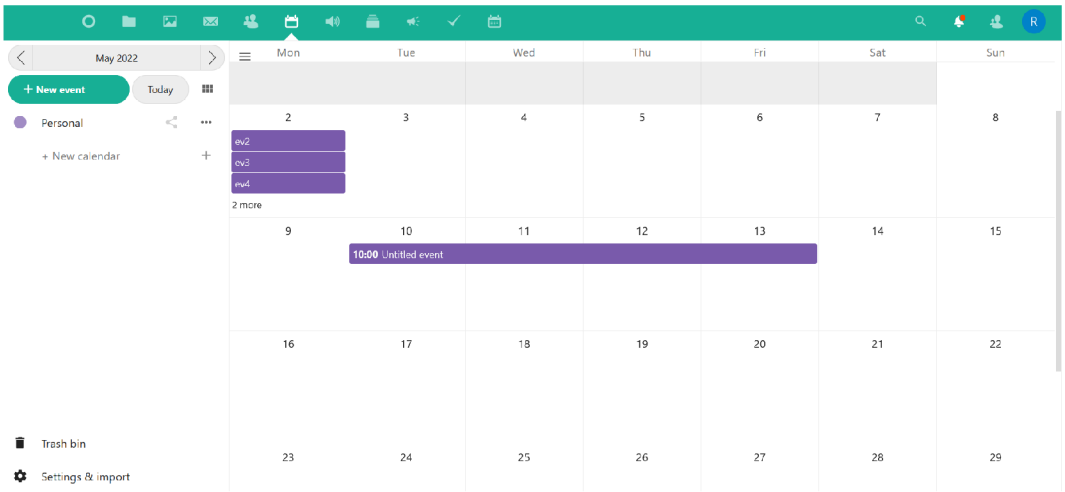
Obrázek 2.12: Seznam úkolů v aplikaci *QOrganiser*



Obrázek 2.13: Týdenní rozvrh v kalendáři *QOrganiser*

2.3.4 *Nextcloud Calendar*

Nextcloud [7] je software typu klient-server pro vytváření a používání služeb hostingu souborů. Jelikož se jedná o bezplatný software s otevřeným zdrojovým kódem, může si jej kdokoli nainstalovat a provozovat na svých soukromých serverech.
Nextcloud je funkčně podobný systémům jako *Dropbox*, *Office 365* nebo *Google Drive*, pokud se používá s integrovanými systémy *Collabora Online* nebo *OnlyOffice*. Může být hostován v cloudu nebo lokálně [5].
Tento systém se vyvinul ze systému *ownCloud*, který byl poté pozměněn, a byl z něj vytvořen *Nextcloud*, který je nadále aktivně vyvíjen. [6]
Jedná se sice primárně o úložiště souborů, systém je ale velmi modulární a obsahuje celou řadu pluginů, které si uživatelé mohou nainstalovat. Jedním z nich je právě kalendář.



Obrázek 2.14: Zobrazení měsíce v kalendářní aplikace *Nextcloud*

Nextcloud Calendar má opět podobné rozhraní jako kalendář *Google*, má však o něco méně tlačítek na lištách, a panely s ovládacími prvky jsou výrazněji graficky oddělené. Políčka v kalendáři, znázorňující dny z předchozího a následujícího měsíce, jsou také lépe odlišena od ostatních tím, že je šedě vybarveno nejenom číslo dne, ale i jejich pozadí. *Nextcloud Calendar* také umožňuje aplikaci spouštět z desktopového klienta, a lze ho zprovoznit i na systémech *Linux*.

Samotná data o kalendářních událostech v kalendáři *Nextcloud* nejsou uložena na počítači klienta, nýbrž na vzdáleném serveru v databázi vlastněné společností *Nextcloud*. K těmto datům je možné přistupovat pomocí speciálních požadavků (tzv. *queries*). Ty musí mít vždy specifikovány konkrétního uživatele, jehož data požadují, a mohou mít specifikována další kritéria jako datový rozsah nebo filtr podle textových řetězců. Komunikaci mezi klientem a serverem dále definuje standard *CalDAV* [2], což je rozšíření protokolu *RFC4918* [13], definujícího komunikaci se vzdáleným serverem.

```php
<?php

use OCP\Calendar\IManager;

class MyService {

    /** @var IManager */
    private $calendarManager;

    public function __construct(IManager $calendarManager) {
        $this->calendarManager = $calendarManager;
    }

    public function searchInUserCalendar(string $uid,
                                          string $calendarUri,
                                          DateTimeImmutable $from,
                                          DateTimeImmutable $to): void {
        $principal = 'principals/users/' . $uid;

        // Prepare the query
        $query = $this->calendarManager->newQuery($principal);
        $query->addSearchCalendar($uri);
        $query->setTimerangeStart($from);
        $query->setTimerangeEnd($to);

        // Execute the query
        $objects = $this->calendarManager->searchForPrincipal($query);
    }

}
```

Obrázek 2.15: Příklad použití *query* v kalendáři *Nextcloud*. Databázi je zaslán požadavek pro události z kalendáře konkrétního uživatele a je v něm specifikován časový úsek, ze kterého má databáze události vracet [11]

Kapitola 3

Návrh aplikace a rozhraní

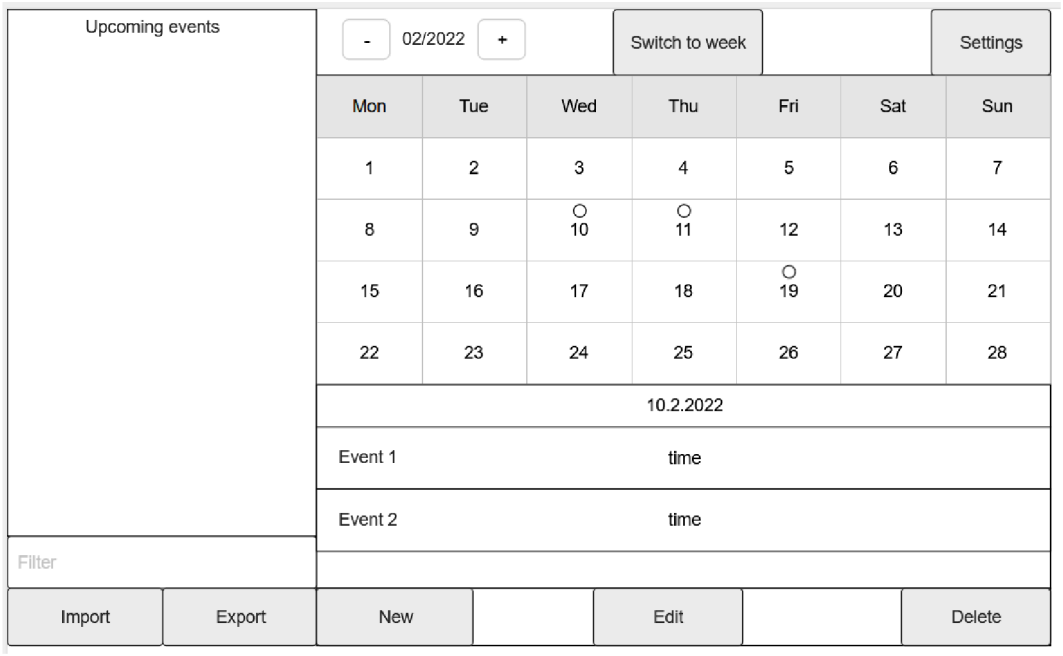
Tato kapitola se zabývá návrhem uživatelského rozhraní vlastní kalendářní aplikace a datových struktur pro práci s ní. Kalendářní aplikace prozkoumané v předchozí kapitole slouží jako inspirace pro tuto kapitolu.
Aplikace bude vytvořena v jazyce *C++17* s pomocí frameworku *Qt* [9], použitého pro tvorbu uživatelského rozhraní. Aplikace má fungovat jako desktopová i jako webová aplikace. K tomu je požit nástroj *Emscripten*, který z *C++* kódu vytvoří ekvivalentní kód v prostředí *WebAssembly*, který je poté možné uložit na web.

3.1 Návrh uživatelského rozhraní

Uživatelské rozhraní by mělo být v první řadě přehledné a snadno ovladatelné. Uživateli by měly být všechny ovládací prvky k dispozici přímo v daném pohledu, místo toho, aby byly schované v dalších oknech. Ovládací prvky by měly být dobře viditelné a intuitivně pojmenované. Kalendář by měl v daném pohledu zobrazovat obecné informace o všech událostech, které probíhají v daném časovém rámci, a nabízet možnost zobrazení podrobnějších informací o událostech, které si vybereme. Samozřejmě musí také umožňovat vytváření, upravování a odstraňování událostí. Kalendář by měl mít dva typy pohledu: měsíční a týdenní, mezi nimiž by mělo být možné jednoduše přepínat. Kalendářní aplikace, prozkoumané v předchozí kapitole, navíc také umožňovaly roční a denní pohled. Denní pohled byl ovšem zpravidla identický se zobrazením týdne, pouze zobrazoval informace o menším počtu dní, a roční pohled sloužil pouze k navigaci do jiných typů pohledu, což by tato aplikace měla být schopna efektivně zajistit i bez něj. Rozhodl jsem se je proto do této aplikace nezahrnout. V obou typech pohledu by měly být události zobrazeny určitou formou tabulky, kde je možné klikat na jednotlivé události a zobrazovat informace o nich.

3.1.1 Zobrazení měsíce

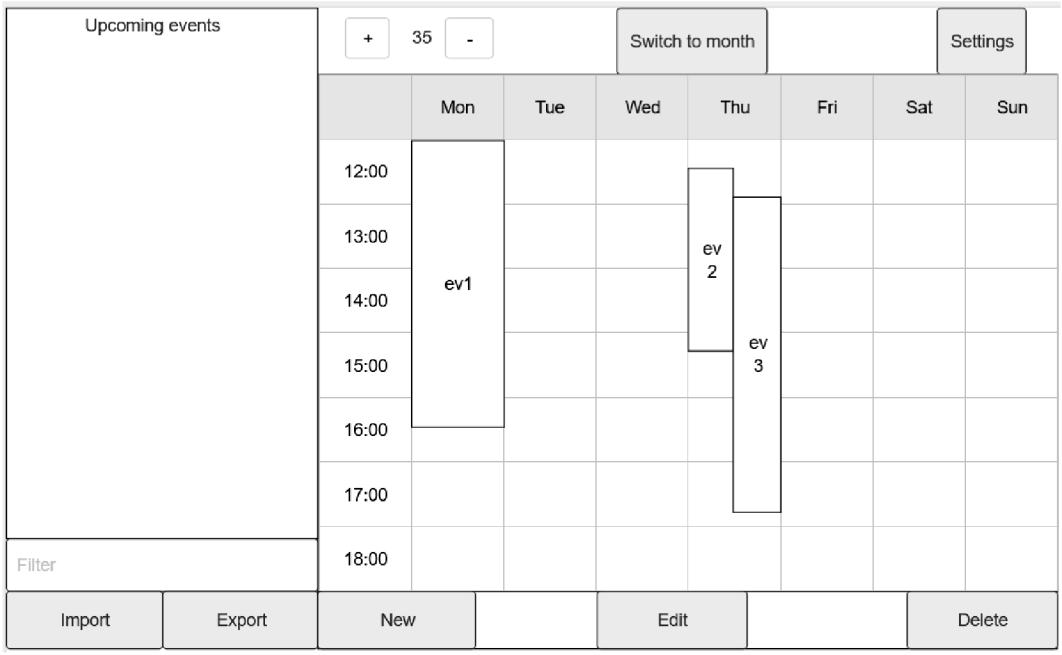
Na obrázku 3.1 je vidět předběžná podoba zobrazení měsíce v aplikaci. V horní liště je zobrazen aktuální měsíc a rok, a je zde možnost přesunout se na jiné datum. Lišta také umožňuje zobrazit okno s nastavením a přesunout se do zobrazení týdne.
Pod lištou se nachází tabulka s aktuálně vybraným měsícem. Dny jsou uspořádány tak, aby stejné dny v týdnu byly ve stejném sloupci. Dny, ve kterých se vyskytují nějaké události, budou určitým způsobem označeny.
Pod tabulkou se nachází seznam s událostmi vybraného dne (pokud se v daném dni nějaké vyskytují). Bezprostředně po spuštění kalendáře bude vybrán dnešní den. U událostí bude zobrazen jejich název a čas, a po kliknutí na danou událost se zobrazí další informace. Pod seznamem budou také k dispozici tlačítka pro úpravu a odstranění vybrané události a pro vytvoření nové události v daném dni.
Podrobné informace o vybrané události budou umístěny v levé části pohledu. Při prvním zobrazení pohledu zde budou zobrazeny informace o první události vyskytující se na seznamu událostí pro dnešní den (pokud taková událost existuje). Po kliknutí na konkrétní den se automaticky vybere první událost v seznamu, a po kliknutí na jinou událost v seznamu se zobrazí informace o ní. Pod informacemi o události je také umístěn filtr, s jehož pomocí můžeme zobrazit pouze události patřící do určité kategorie, a pod filtrem se nachází tlačítka pro import a export kalendáře.



Obrázek 3.1: Návrh zobrazení měsíce vlastní kalendářní aplikace

3.1.2 Zobrazení týdne

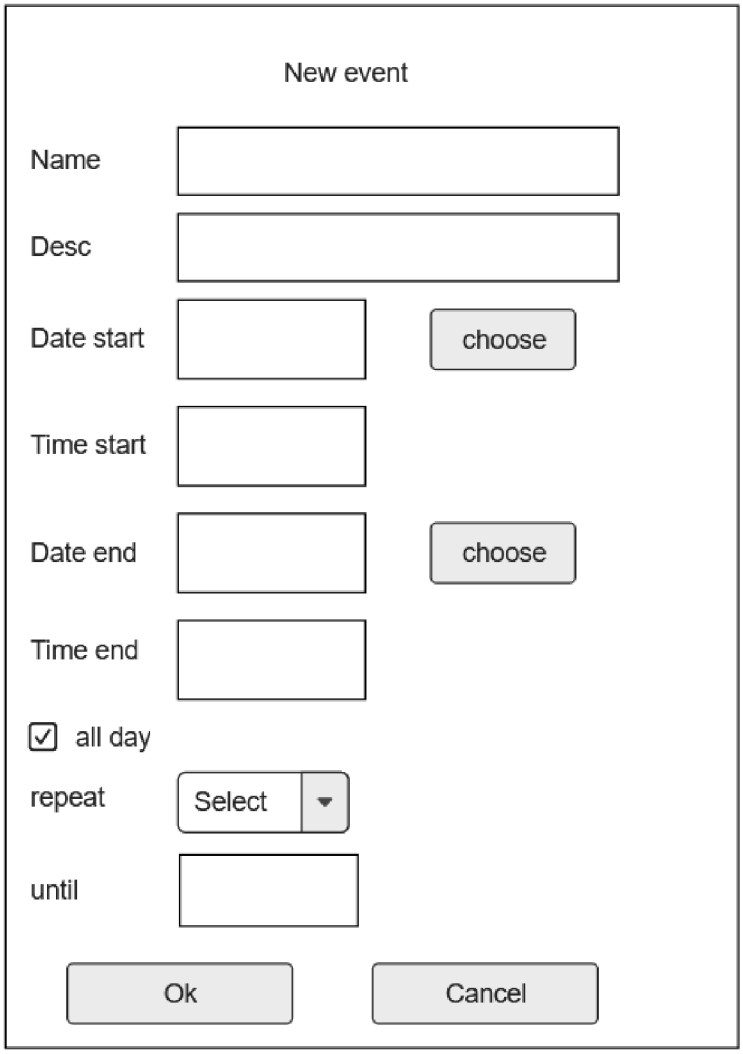
Obrázek 3.2 zobrazuje předběžnou podobu zobrazení týdne v aplikaci. Levá část s detailním popisem události, filtrem a exportem/importem událostí zůstává stejná. V horní části se místo datumu nachází číslo aktuálního týdne (při spuštění aplikace by se měl časový rámec nastavit na současný týden). V tabulce událostí jsou jednotlivé dny reprezentovány sloupci v tabulce, zatímco řádky reprezentují jednotlivé hodiny. Pravděpodobně nebude praktické natěsnat na sebe řádky tak, aby se do pohledu vešlo všech 24 hodin. Proto bude zobrazen pouze určitý časový úsek, s tím, že bude možné tabulkou posouvat nahoru a dolů. Události jsou zobrazovány v podobě barevných obdélníků, umístěných do tabulky podle jejich doby trvání. Jelikož doba trvání vyplývá přímo z umístění dané události v tabulce, není již třeba ji explicitně psát na událost samotnou, takže je u ní uveden pouze název. Po kliknutí na obdélník zobrazující danou událost se o ní vypíší detailní informace v levé části pohledu. Vzhledem k tomu, že jsou všechny události vypsány v samotném pohledu, není už potřeba samostatného seznamu událostí jako v zobrazení měsíce. Místo toho je uvolněné místo zabráno tabulkou, aby bylo možné zobrazit větší časový úsek.



Obrázek 3.2: Návrh zobrazení týdne vlastní kalendářní aplikace

3.1.3 Vytvoření/úprava události

Obrázek 3.3 ukazuje, jak by mělo vypadat okno pro vytvoření nebo úpravu události. Kromě klasického výčtu všech možných atributů aplikace se v okně nachází i tlačítko pro výběr počátečního a koncového datumu přímo na kalendáři. Výběr datumu probíhá tím způsobem, že se aplikace přepne do zobrazení měsíce, kde klikneme na odpovídající den. Políčka pro výběr počátečního a koncového času budou viditelná pouze v případě, že políčko *all day* není zaškrtnuto (výběr času by v tomto případě nedával smysl). Stejně tak políčko pro datum ukončení opakování by mělo být viditelné pouze v případě, že se událost vůbec bude opakovat. V případě, že upravujeme již existující událost, se její hodnoty předem vypíší do odpovídajících položek. Okno také obsahuje tlačítko *cancel* pro anulování veškerých provedených změn.



Obrázek 3.3: Návrh okna pro vytváření událostí ve vlastní kalendářní aplikaci

3.2 Návrh knihovny datových struktur

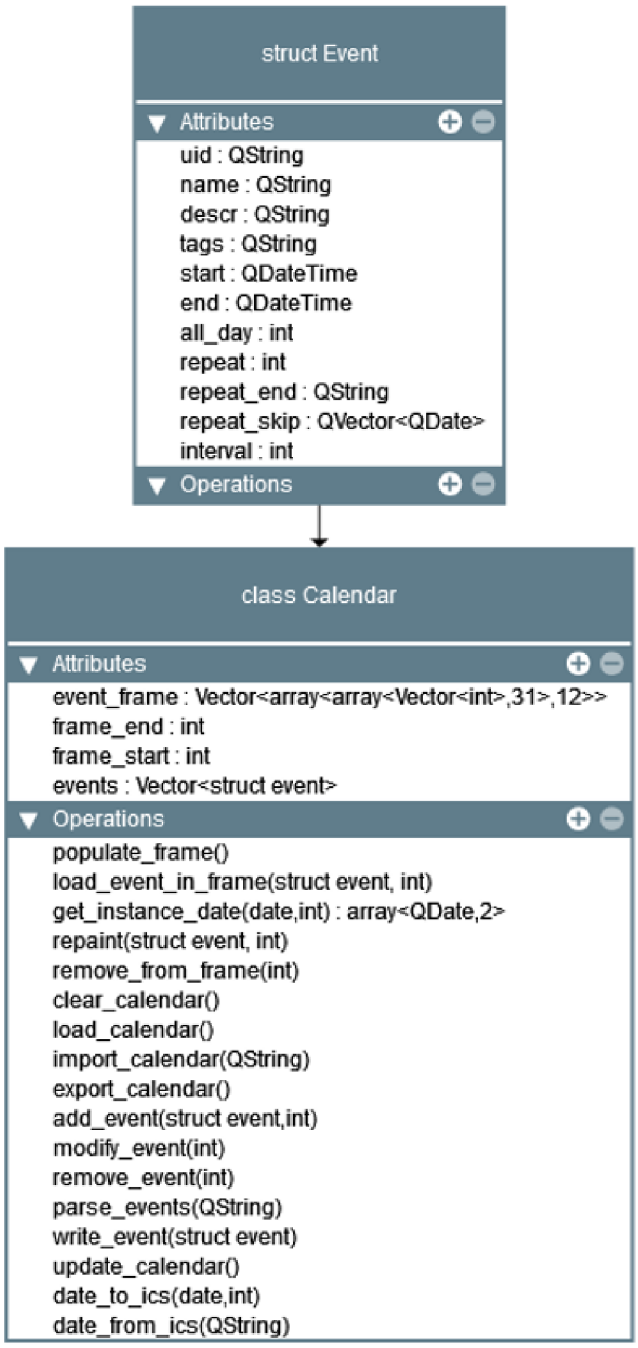
Kalendářní knihovna zobrazená na obrázku 3.4 má za úkol zpracovávat a ukládat data z nahraného kalendářního souboru. Pro tento účel obsahuje třídu *Calendar*, která má za úkol zpracovat data z kalendářního souboru takovým způsobem, aby s nimi zbytek aplikace mohl pracovat. Knihovna bude také definovat strukturu *event*, v níž budou obsaženy informace k jednotlivým událostem.

Třída *Calendar*

Tato třída bude sloužit ke zpracovávání dat o kalendářních událostech. Bude mít na starosti načítání dat o událostech ze souborů ve formátu *ICalendar*, jejich ukládání do paměti v podobě struktury *event*, a přepisování souborů s událostmi v případě modifikace dat v nich. Události obsažené v těchto strukturách bude schopna pomocí dalších metod modifikovat. Dále je potřeba, aby obsahovala funkce pro export a import kalendářů, a pro překlad dat z podoby textového řetězce do podoby kalendářní struktury a naopak. Jelikož jsou datumy v kalendářních souborech zapsány v podobě textového řetězce, musí je také umět přeměnit na formát *DateTime*.
Třída bude také zjišťovat, ve které dny je potřeba zobrazovat dané události, aby tato informace nemusela být vypočítávána zbytkem aplikace. K tomu slouží konstrukce *event frame*, která napodobuje rozložení dní v kalendáři (pole roků obsahující pole měsíců, jež obsahuje pole dní). V této konstrukci (jedná se ve skutečnosti o vektor dvojrozměrného pole vektorů) se v každém políčku reprezentujícím konkrétní den bude vyskytovat vektor s odkazy na události, které jsou na daný den naplánovány. K vyplňování této konstrukce bude sloužit metoda *populate frame*. Instance této třídy bude obsažena ve třídě *MainWindow*.

Struktura *event*

Tato struktura slouží pro uložení všech informací o konkrétní události zapsané v souboru ve formátu *ICalendar*. Bude obsahovat název a popis události, identifikační řetězec, datum a čas začátku a konce události, formu a interval opakování události, a datum konce opakování. Kromě toho bude také obsahovat informaci, jestli událost zabírá celý den, a vektor datumů, ve kterých platí výjimka ze standardního opakování události. Pokud tato událost vznikla úpravou jedné instance opakující se události, je zde také přítomen odkaz na původní událost. Vzhledem k tomu, že se ve formátu *ICalendar* mohou vyskytovat data, se kterými tato knihovna nedokáže pracovat, budou ve struktuře tato data obsažena v textové podobě z důvodu zachování dat při exportu a importu.



Obrázek 3.4: Diagram třídy *Calendar* a datových struktur knihovny

3.3 Návrh aplikace

Aplikace, jejíž diagram je zobrazen na obrázku 3.5, má za úkol správně zobrazovat data získaná z knihovny. Dále musí umět reagovat na podněty uživatele, zajišťovat navigaci mezi jednotlivými okny a umožňovat uživateli upravovat data uložená v knihovně. Pro tento účel bude obsahovat třídu *MainWindow* pro zobrazování událostí, a pro celkové ovládání aplikace, a struktury *config*, *week view* a *week view event* pro uchovávání dodatečných informací potřebných pro správné zobrazování kalendářních událostí.

Struktura *config*

V této struktuře bude uložena konfigurace kalendáře. Bude se jednat o interpretaci dat, uložených v konfiguračním souboru a interpretovaných třídou *MainWindow*.

Struktura *week view*

Tato struktura slouží k uchovávání doplňujících informací při zobrazení týdne. Bude obsahovat informace o aktuálním týdnu (začátek, konec, číslo týdne v roce), odkaz na widgety sloužící k zobrazení týdenních událostí, a dodatečné informace potřebné pro správné vykreslení týdenních událostí. Bude také obsahovat vektor dvojrozměrných polí, který bude napodobovat tabulku, ve které se události v zobrazení týdne budou zobrazovat.

Struktura *week view event*

Tato struktura bude obsahovat informace o jednotlivých událostech v zobrazení týdne, které budou obsaženy ve výše zmíněné struktuře *week view*. Struktury se budou nacházet v dvourozměrném poli, kde každá položka pole odpovídá jednomu dni. Budou obsahovat upravený datum začátku a konce události (pro případ, že se jedná o instanci opakované události), odkaz na událost v paměti kalendáře, a informace potřebné ke správnému vykreslení události na tabulku v týdenním zobrazení.

Třída *MainWindow*

Tato třída má za úkol implementovat uživatelské rozhraní: Bude zobrazovat informace, které třída *Calendar* získala, manipulovat s daty podle podnětů uživatele, a volat třídu *Calendar* v případě, že je potřeba přepsat data o uložených událostech (z důvodu zabránění ztráty dat by se data v souborech měla přepisovat bezprostředně po jejich změně v aplikaci).
Třída bude reagovat na podněty uživatele pomocí *signálů* a *slotů*. *Signál* je určitá uživatelova akce definovaná frameworkem (například kliknutí na určité políčko tabulky, změna hodnoty v okně, atd.), *slot* je funkce definovaná jako reakce na tento signál.
Třída musí obsahovat informace o dnešním datu, a o dni, na který je aplikace právě zaměřená. Dále musí mít uloženu událost, u níž právě vypisujeme detailní informace (resp. událost, kterou právě upravujeme). Třída také musí umět přepínat pohled mezi jednotlivými okny aplikace, a volat patřičné metody třídy *Calendar*, pokud je potřeba zpracovat nová data nebo pozměnit existující data.

Obrázek 3.5: Diagram třídy *MainWindow* a jejích struktur

Kapitola 4

Implementace a testování

Aplikace byla implementována v jazyce C++17 s použitím frameworku *Qt 5.15*. Byla vytvořena pomocí nástroje *Qt Creator*. [10] Verze aplikace pro web byla přeložena nástrojem *Emscripten 1.39* do prostředí *WebAssembly*, a desktopová verze byla přeložena nástrojem *MinGW 8.1*. Obě verze vychází ze stejného zdrojového kódu. Desktopová aplikace byla vyvinuta pro systém *Windows*, je však kompatibilní i se systémem *linux*. Webová aplikace funguje ve všech prohlížečích, které podporují *WebAssembly* (*Safari*, *Chrome*, *Firefox*, *Edge*).

4.1 Technická struktura aplikace

Zdrojový kód aplikace obsahuje soubor *Calendar.pro*, který slouží ke generování *Makefile* prostředím *Qt*. Tento soubor definuje věci jako soubory potřebné pro překlad, cílovou složku pro sestavený projekt nebo další příkazy a parametry pro kompilátor. Kromě toho obsahuje soubor *Calendar.pro.user* s uživatelským nastavením pro vývojové prostředí.
Samotný kód kalendářní knihovny je obsažen v souborech *Calendar.cpp* a *MainWindow.cpp* a jejich hlavičkových souborech. Soubor *MainWindow.ui* obsahuje definici grafického rozhraní. Tento soubor je automaticky generován vývojovým prostředím poté, co je v něm dané rozhraní namodelováno.
Výslednou aplikaci lze ze zdrojového kódu přeložit pomocí nástroje *Qt Creator* verze 6.0+. Pro desktopovou verzi je potřeba specifikovat jako nástroj pro překlad *MinGW 64-bit*, a pro webovou verzi překladač *wasm 32* (viz návod k instalaci). K samotnému spuštění výsledné aplikace už *Qt Creator* není potřeba.
V případě desktopové verze je výsledkem sestavení aplikace spustitelný soubor ve složce *release*, v případě webové aplikace čtveřice souborů, které je možné uložit na web.
U webové aplikace bylo ještě potřeba vyřešit problém s persistencí dat: Aplikace vytváří a nahrává soubor *cal.ics* pro uchovávání informací o kalendářních událostech, a soubor *config.cfg*, pro uchovávání informací o konfiguraci aplikace. Prostředí *WebAssembly* má ovšem formu uzavřeného sandboxu, která neumožňuje komunikaci se soubory mimo něj (nemůžeme samozřejmě dovolit webové stránce, aby měla přístup k souborům v počítači). Proto bylo potřeba do projektu zahrnout soubor *presettings.js*. Jedná se o oficiální *Qt* plugin v jazyce *Javascript*, který pro sandbox vytvoří virtuální databázi *IFDBS*, ke které jde přistupovat jako k běžnému souborovému systému v počítači. Tato databáze je uložena v paměti prohlížeče, a přetrvává až do chvíle, kdy se uživatel rozhodne vyprázdnit *cache* daného prohlížeče. Ve webové podobě má výsledná aplikace velikost cca. 16Mb. Desktopová aplikace má sama

o sobě velikost v řádech stovek Kb, potřebuje ale ke správnému fungování několik *Qt* pluginů ve svém adresáři. Celková velikost proto naroste na několik desítek Mb.

4.1.1 Zpracovávání kalendářních souborů

Zpracování dat o událostech má na starosti třída *Calendar*. Při spuštění aplikace je třídou *MainWindow* zavolána metoda *load calendar* třídy *Calendar*, která načte text ze souboru *cal.ics* ve složce */etc* (pokud tento soubor ještě neexistuje, proces se přeskočí a aplikace zobrazí prázdný kalendář bez událostí). Tento text poté rozdělí na podřetězce, kde jako oddělovač použije řetězec *BEGIN:VEVENT* (jelikož ve formátu *ICalendar* je takto označován začátek nové události). Tyto podřetězce jsou poté zaslány metodě *parse events*.
Tato metoda podřetězce rozdělí na jednotlivé řádky. U každého řádku metoda zkontroluje, jestli obsahuje jedno z klíčových slov používaných standardem *ICalendar* (jako *SUMMARY*, *DTSTART*, apod.). Pokud ano, je z řádku získána hodnota a je uložena do příslušného atributu nově vytvořené struktury *event*. V případě datumů je potřeba zavolat metodu *date from ics*, která řetězec konvertuje na datum a čas formátu *DateTime* (kvůli zvláštnímu formátu, ve kterém se podle standardu *ICalendar* ukládají datumy, nelze volat standardní *QDateTime::fromString*, ale je potřeba implementovat vlastní metodu). Pokud řádek neobsahuje žádné klíčové slovo relevantní pro zpracování události, je jeho řetězec přidán do speciálního atributu události. To proto, aby zůstala zachována data vzniklá importem z jiných aplikací. Po průchodu všemi řádky podřetězce metoda zkontroluje, jestli je ve struktuře dostatek informací pro to, aby byla událost validní (musí mít jméno a validní datum začátku a konce). Pokud ano, je událost přidána na konec pole událostí kalendáře.
Obdobným způsobem třída postupuje při načítání nového kalendáře. Metoda *import calendar* opět rozdělí události na podřetězce, které pošle metodě *parse events*. Předtím ještě zkontroluje, jestli chtěl uživatel předchozí kalendář přepsat, a pokud ano, vymaže původní obsah pole struktur.
Třída také obsahuje metodu, díky které může události přetvořit zpět na textové řetězce ve formátu *ICalendar*. Té využívá, pokud chceme kalendář exportovat, a také pokaždé, když dojde k modifikaci existujících dat (přidání/úprava/odstranění existující události). Při modifikaci metoda používá třídu *QSaveFile*, která zajišťuje atomičnost při přepisování souboru s daty. Soubor je normálně otevřen pro zápis, ale k samotnému přepsání souboru dojde až po zadání instrukce *commit*, aby nemohlo dojít k tomu, že se při nějaké chybě programu zapíše jenom část dat. Pokud by byl proces přepisování přerušen, zůstane soubor zachován v předchozí podobě.

4.1.2 Načítání dat pro zobrazení událostí

Kromě samotného uložení dat do paměti potřebuje třída *Calendar* ještě také zjistit, ve které dny se dané události budou zobrazovat. Tento údaj je zapotřebí k tomu, aby při samotném zobrazování událostí nemusely metody u každé události kontrolovat, jestli se vyskytuje v daném časovém rámci. K tomu *Calendar* využívá vektor *event frame*. Tento vektor má jednu položku pro každý rok ve stanoveném rozsahu (na začátku aplikace jsou to 3 roky do minulosti a 3 roky do budoucnosti). Každá z těchto položek má podobu pole o velikosti 12 (1 pro každý měsíc v daném roce). Každá položka tohoto pole obsahuje další pole o velikosti 31 pro jednotlivé dny v daném měsíci. Položkami těchto polí jsou vektory s čísly, které odpovídají pozicím těch událostí v paměti, které se odehrávají v daný den. Pokud máme například stanovený rozsah 2019 až 2025, a chceme zjistit, jaké události se odehrávají 5. března 2020, podíváme na pozici *event frame[1][3][5]* a tam najdeme vektor

ukazující na pozice všech událostí, které se v ten den odehrávají. Tato podoba zápisu sice působí poněkud složitě, ale aplikace se dokáže v této proměnné jednoduše orientovat díky tomu, že pracuje s datumy uloženými ve formě *QDate*, ze kterých si může vytáhnout číslo dne, měsíce a roku. Pokud má například uložený dnešek v proměnné c, může pozici dnešních událostí najít pomocí
$pos = event\_frame[c.year() - frame\_start][c.month() - 1][c.day() - 1][i]$
Do tohoto rámce metoda odkazy umístí tak, že postupně projde všechny události v paměti, a dá odkazy do všech políček mezi datumem začátku a konce události (pokud se tedy událost nachází v daném rámci).

Opakující se události

U opakujících se událostí metoda umístí do rámce všechny instance opakování, které do něj patří. Toho dosáhne tím, že podle daných pravidel opakování události inkrementuje datum pomocí metod *add days*, *add months* nebo *add years*. Díky tomu nemusí řešit čtení mimo rozsah pole (pokud je na konci měsíce a zavolá *add days*, datum se automaticky překlopí do dalšího měsíce).
Je zde ovšem problém s překrývajícími se instancemi událostí: při zobrazování událostí potřebujeme znát datum začátku a konce konkrétní instance, a tímto způsobem ho nemáme jak zjistit v případě, že se několik instancí překrývá. *Calendar* proto obsahuje další metodu pro zjištění správného datumu dané instance. Využívá k tomu toho, že instance s dřívějším začátkem se do pole zapíší jako první. Proto metodě stačí poslat pořadí dané instance ve vektoru, a z něj získá metoda správné datum přičítáním intervalu opakování k datumu začátku a konce instance, a kontrolou, jestli jsme aktuálního datumu dosáhli tolikrát, kolikrát jsme potřebovali. (Příklad: chceme získat datum instance události s týdenním opakováním, která se vyskytuje v daný den na třetím místě. Nejdříve nastavíme datum začátku a konce instance na datum začátku a konce originální události. Poté tento datum postupně inkrementujeme o týden, a kontrolujeme, jestli se aktuální den nachází uvnitř daného datumu. Třetí nalezená instance opakování, pro kterou toto platí, je právě tou instancí, kterou jsme se snažili najít. Metoda tedy vrátí datum jejího začátku a konce).

4.1.3 Zobrazování událostí a interakce s rozhraním

Zobrazování kalendářních událostí a práci s rozhraním má na starosti třída *MainWindow*. Tato třída manipuluje s prvky rozhraní pomocí signálů a slotů. Prvky rozhraní mají podle svého typu definovány signály, které mohou posílat (např. dvojklik na prvek, změna hodnoty prvku, kliknutí na políčko tabulky prvku, atd.). Na tyto signály je ve třídě *MainWindow* poté definována reakce v podobě slotů (v podstatě se jedná o funkce, které se automaticky zavolají při daném signálu). Tyto dvojice signál-slot jsou definovány na začátku běhu programu. Třída poté načte konfiguraci (pokud konfiguraci nenajde, vytvoří novou s výchozími hodnotami), zavolá třídu *Calendar* pro načtení událostí, a podle nastavení zavolá metodu buďto pro měsíční nebo týdenní zobrazení.

Zobrazení měsíce

Metoda napřed zkontroluje, zda se stále nacházíme uvnitř rámce, ve kterém jsou třídou *Calendar* načteny události, a pokud ne, zavolá metodu pro jeho rozšíření. Poté metoda nastaví tabulku zobrazení měsíce (která má formu *QTableWidget*). Metoda nejprve nastaví datum na 1. den současného měsíce, a poté na pondělí týdne, ve kterém se daný den nachází (abychom zachovali tvar tabulky, která má 7 sloupců a 6 řádků). Poté prochází a nastavuje všechna okna tabulky: U každého políčka napíše číslo odpovídajícího dne. V případě, že se jedná o políčko reprezentující den, který se vyskytuje mimo současný měsíc, nastaví barvu písma šedě, jinak ji nastaví černě pro pracovní dny a červeně pro víkendy. Když se jedná o dnešní den, zvýrazní ho tučně. Jestliže se v daném dni vyskytují nějaké události, tak políčko vybarví zeleně (světle-zeleně pro události, které ještě nenastaly, tmavě-zeleně pro události v minulosti). Tento proces se opakuje pokaždé, když se přepneme na jiný měsíc nebo změníme v kalendáři nějakou událost. Pokud se jedná o první vykreslení pohledu po zapnutí aplikace, nebo po kliknutí na tlačítko *Current*, zavolá se také metoda pro zobrazení událostí dnešního dne.

Výpis událostí dne

Metoda pro zobrazení událostí dne se volá při kliknutí na políčko tabulky. Metoda nejprve zkontroluje, jestli dané políčko reprezentuje den patřící do současného měsíce. Pokud ne, místo zobrazení zavolá metodu pro přepnutí pohledu do daného měsíce (pokud tato funkcionalita není zakázána v nastavení). Pokud ano, podívá se do tabulky, jestli vektor pro daný den obsahuje nějaké elementy (pokud je zadán filtr, ještě zkontroluje, jestli dané elementy filtrem projdou). Ty pak zobrazí jako elementy v pravé liště (která má formu *QListWidget*), u kterých napíše jejich název a čas trvání v daném dni. Jestliže událost trvá přes celý vybraný den, napíše u ní místo času trvání "all day".
Po kliknutí na událost v seznamu se zavolá metoda pro vypsání detailních informací o události (při prvním kliknutí na políčko dne je automaticky vybrána první položka v seznamu). Metoda se opět podívá do tabulky, kde zjistí index dané události, a podle něj ji vyhledá v paměti kalendáře. Pak z ní vypíše všechna přítomná data do levé lišty (která má formu *QLabel*). V případě, že se jedná o událost s opakováním, načte místo datumů uložených ve struktuře datumy získané zavoláním metody třídy *Calendar* pro načtení datumů dané instance.

Zobrazení týdne

Metoda pro zobrazení týdne nalezne potřebné události podobným způsobem, jako metoda pro zobrazení měsíce, jenom pracuje s týdenním rámcem místo měsíčního. Samotné vykreslování událostí probíhá do 7 tabulek uspořádaných vedle sebe v horizontálním zobrazení (*QHboxLayout*). Tuto formu jsem zvolil proto, aby bylo možné jednoduše přidávat a odebírat sloupce v tabulce bez toho, aby se měnila velikost samotných tabulek (což je nutné kvůli zajištění možnosti zobrazovat více událostí odehrávajících se ve stejnou dobu). Listování tabulkou bylo umožněno použitím widgetu *Scroll Area*, podporovaného prostředím *Qt*.
Samotné tabulky jsou reprezentovány pomocí pole dvourozměrných vektorů, definovaného ve třídě *MainWindow*. Jak pole, tak tabulka mají jeden řádek pro každý desetiminutový

interval daného dne (celkem tedy 144 řádků), a na počátku má každá z tabulek jeden sloupec. U každé události metoda napřed zjistí, jaký časový úsek událost zabírá v daném dni (pokud začíná už před daným dnem, nastaví se začátek na 0:00, pokud končí až po daném dni, nastaví se konec na 23:59). Poté metoda určí počáteční bod události v tabulce (řádek reprezentující časový interval, ve kterém událost pro ten den začíná), a vypočítá, kolik řádků bude zabírat.
Následně zkontroluje, jestli je pro událost v poli místo: každé políčko pole (reprezentující políčko tabulky v týdenním pohledu) má v sobě zapsáno číslo reprezentující index události, která je v daném místě tabulky vykreslená. Políčka, ve kterých se žádná událost nevyskytuje, mají hodnotu -1. Pokud je v daném sloupci místo, zapíše se index události do všech políček daného sloupce reprezentujících časové úseky, ve kterých se daná událost vyskytuje, a daná políčka tabulky se sloučí do jednoho. Pokud místo není, podívá se metoda do dalšího sloupce. V případě, že v tabulce už žádný další sloupec není, tabulka se rozšíří o nový sloupec, a událost se umístí do něj. Při přepnutí do jiného týdne nebo při změně některé z událostí se tabulka vyčistí, tabulka i pole se opět zmenší na jeden sloupec, do pole se na všechny pozice zapíše hodnota -1, a proces proběhne znovu.
Týdenní pohled vypisuje jednotlivé hodiny vedle tabulek a dny v týdnu společně s jejich datumy nad nimi (nejedná se o přímou součást tabulky, ale o na ní nezávislé elementy). Zobrazování podrobnějších informací o události probíhá pomocí stejných metod jako u zobrazení měsíce. Jelikož týdenní pohled neobsahuje seznam denních událostí jako je tomu u zobrazení měsíce, volá se metoda pro vypsání informací kliknutím na samotnou událost v tabulce.

Vytváření a úprava událostí

Vytváření nové události a úprava existující události využívají stejné dialogové okno. Metodu pro zobrazení okna pro vytvoření nové události může uživatel aktivovat buďto kliknutím na políčko požadovaného dne, a poté kliknutím na tlačítko *New*, nebo dvojklikem na políčko dne (pokud je v zobrazení týdne, tak dvojklikem na prázdné místo v tabulce). Metodu pro úpravu události uživatel aktivuje vybráním konkrétní události a kliknutím na tlačítko *Edit* nebo dvojklikem na konkrétní událost.
Při úpravě se do okna načtou odpovídající hodnoty z právě vybrané události, při vytváření nové události se do něj načtou výchozí hodnoty specifikované v nastavení. Zároveň se nastaví viditelnost jednotlivých políček v okně podle zvolených hodnot (například políčka pro výběr času začátku a konce události jsou skrytá, pokud uživatel zvolí, že má událost trvat celý den, pokud uživatel zvolil, že se událost nemá opakovat, nebude se ho okno ptát, kdy má událost s opakováním přestat, atd.). Tato políčka se poté budou dynamicky zobrazovat a skrývat podle toho, jaké hodnoty jsou v okně právě zadány.
Okno uživateli mimo jiné umožňuje nastavit datum začátku a konce události jeho výběrem přímo v kalendáři. Toho se dosáhne tím, že po kliknutí na tlačítko *Pick date* se znovu zavolá metoda pro vykreslení měsíce, tentokrát ale s pozměněnými atributy v třídě. Tyto atributy způsobí, že metoda nebude vyhledávat události, pouze zobrazí kalendář, ve kterém je možné se dále přesunovat stejně jako v normálním zobrazení, a zvýrazní druhé vybrané datum z menu tvorby události (vybrané datum konce, pokud právě vybíráme datum začátku, a vybrané datum začátku, pokud právě vybíráme datum konce).
Pokud uživatel upravuje událost, která má nastavené opakování, umožní mu okno také specifikovat, jestli chce upravovat celou sérii opakujících se událostí, nebo pouze danou konkrétní instanci opakování. V prvním případě se datum a čas začátku a konce automa-

ticky nastaví na datum a čas původní události, ve druhém případě se automaticky nastaví na datum a čas dané instance opakování.

Ukládání změn do paměti

Po kliknutí na tlačítko *Ok* se zavolá odpovídající metoda pro zapsání změn do třídy *calendar*. V případě nové události se událost přidá na konec pole, v případě úpravy existující události se nahradí událost v poli na daném indexu. Poté je zavolána metoda, která vykreslí danou událost do rámce (u nové události jde pouze o vykreslení, při úpravě se napřed musí z rámce odstranit původní vykreslení události). Potom se znovu zavolá metoda pro zobrazení měsíce/týdne, aby se změny projevily v zobrazení.
Pokud uživatel upravoval pouze jednu instanci opakující se události, tak se v kalendáři vytvoří nová událost, která ve své struktuře bude mít odkaz na původní událost, ze které byla vytvořena. Pokud si někdy v budoucnu rozklikne uživatel danou událost, metoda pro její zobrazení uvidí, že se odkazuje na jinou událost, a podle toho načte relevantní informace. V původní události se poté do seznamu výjimek přidá datum, ve kterém by měla daná instance opakování normálně začínat, aby metoda věděla, že byla instance nahrazena.
Při kliknutí na tlačítko *Cancel* se místo toho struktura zahodí bez toho, aby byly v kalendáři provedeny nějaké změny. V obou případech se poté okno skryje, a uživatel se vrátí do původního pohledu.
Odstranění události probíhá tak, že se událost vymaže z pole událostí v kalendáři (jak v poli struktur, tak v rámci). Pokud chce uživatel odstranit opakující se událost, zeptá se ho aplikace, jestli chce odstranit celou sérii opakujících se událostí, nebo pouze jednu instanci. Odstranění jedné instance se provede tím, že se její datum přidá do seznamu výjimek z opakování dané události. Při odstranění celé série opakování musí metoda kromě odstranění události samotné ještě projít seznam událostí a vymazat všechny události s odkazem na ni (jedná se totiž o upravené instance dané série opakování).

4.2 Grafická podoba aplikace

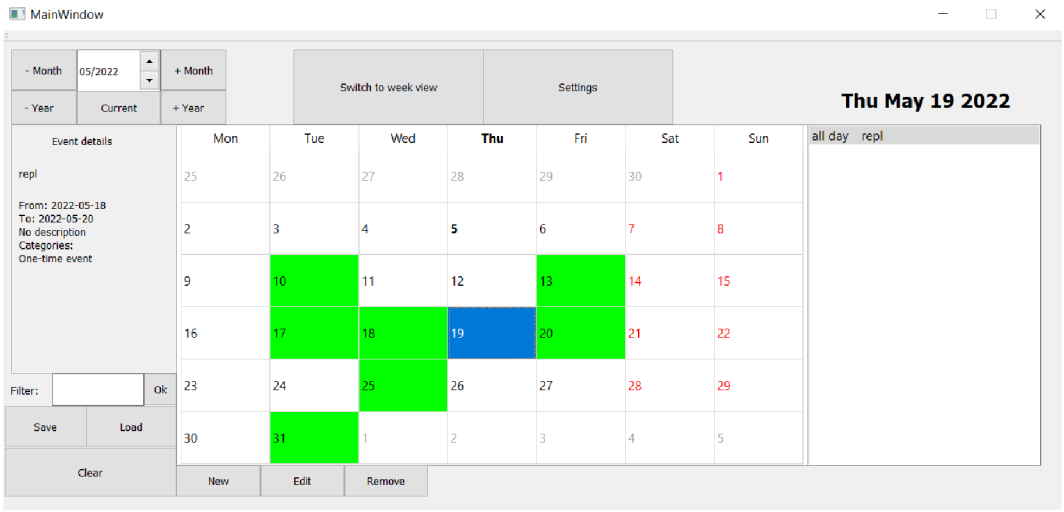
Aplikace má dva režimy zobrazení: zobrazení měsíce a zobrazení týdne. Kromě toho obsahuje okno s dialogem pro vytvoření nové události, případně úpravu existující události, a okno s nastavením aplikace. Aplikace také obsahuje jednoduchý dialog pro nahrání nového kalendářního souboru.

4.2.1 Zobrazení měsíce

Zobrazení měsíce se ve výchozí funkcionalitě zobrazí jako první po zapnutí aplikace. Skládá se ze 4 hlavních částí:

Horní lišta obsahuje tlačítka pro přechod mezi jednotlivými měsíci v kalendáři. Jsou zde tlačítka pro přesun jak o měsíc, tak o rok. Kromě toho aplikace nabízí více možností, jak se přesunovat mezi jednotlivými měsíci, takže si uživatel může vybrat tu, která mu nejvíce vyhovuje: na výběr je přepínání šipkami vedle zobrazení datumu, lze kliknout buďto na měsíc nebo rok a přesunovat se pomocí kolečka myši nebo šipek na klávesnici, nebo je možné buď měsíc nebo rok přímo přepsat. Má také možnost se stisknutím tlačítka *Current*

přesunout zpátky do současného měsíce. Horní lišta kromě toho obsahuje možnost zobrazit okno s nastavením, nebo se přepnout do zobrazení týdne.



Obrázek 4.1: Zobrazení událostí vybraného měsíce ve výsledné kalendářní aplikaci

Uprostřed se nachází tabulka se samotným kalendářem. V tabulce jsou zobrazena čísla jednotlivých dní, a nad nimi jsou zobrazeny dny v týdnu. Víkendové dny jsou vybarveny červeně. Dny, které patří do předchozího nebo následujícího měsíce, jsou vybarveny šedě. Dnešní den je zvýrazněn tučným písmem.
Dny, na které je naplánována nějaká událost, jsou zeleně zvýrazněny. Události, které se udály v minulosti, jsou zvýrazněny tmavě zelenou barvou, ostatní světle zelenou. Po kliknutí na políčko konkrétního dne se toto políčko vybarví modře a vypíše se seznam událostí naplánovaných na daný den. Při spuštění aplikace je automaticky vybrán dnešní den. Po kliknutí na políčko dne, který se v daném měsíci nevyskytuje, se kalendář přepne na zobrazení daného měsíce. Tuto funkcionalitu je možné v nastavení vypnout. Pod samotnou tabulkou se nacházejí tlačítka pro vytvoření nové události, pro úpravu existující události, a pro odstranění události. Tlačítka pro úpravu a odstranění události jsou viditelná pouze tehdy, když je nějaká konkrétní událost vybrána.

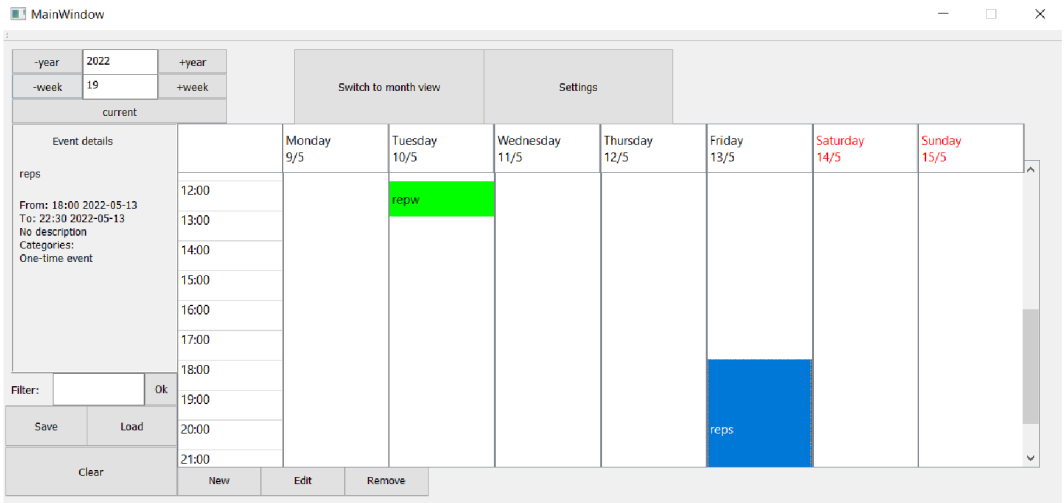
V pravé části aplikace se nachází seznam událostí, vyskytujících se ve vybraném dni. Oproti návrhu byl tento seznam přesunut ze spodní části doprava, aby bylo možné zobrazit více událostí a také proto, že jinak by tabulka byla zbytečně široká. Jednotlivé události jsou zde vypsány ve sloupci pod sebou. Je zde uveden název události a doba jejího trvání. Jestliže událost trvá celý den, zobrazí se poznámka "all day", jinak je uveden čas začátku a konce události. V případě, že se jedná o událost trvající déle než jeden den, je brána v potaz pouze ta část události, která se děje v daný den. Nad seznamem je také zobrazeno aktuální datum.

V levé části aplikace jsou vypsány podrobné informace o aktuálně vybrané události, kterou uživatel vybral kliknutím na položku seznamu událostí v pravé části. Bezprostředně po vybrání dne je automaticky vybrána první položka seznamu. Je zde uveden její název, podrobnější popis, přesný čas začátku a konce události, kategorie, do kterých událost patří, a pravidla, podle kterých se událost opakuje.

Pod popisem události se nachází filtr, který události filtruje podle kategorií, které k nim byly přiřazeny. Poté, co uživatel napíše buďto jednu kategorii, nebo několik kategorií oddělených čárkou, a klikne na tlačítko "Ok", se v kalendáři zobrazí pouze ty události, které obsahují alespoň jednu z uvedených kategorií.
Dále lišta také obsahuje tlačítko pro uložení kalendáře, které zobrazí okno s výběrem lokace, kde se má kalendář ve formátu *ICalendar* uložit, a možností soubor pojmenovat (u webové aplikace se místo toho soubor, automaticky pojmenovaný *cal.ics*, stáhne), a tlačítko pro načtení kalendáře, které zobrazí obdobné okno pro výběr souboru k nahrání.
Pod těmito tlačítky se ještě vyskytuje tlačítko *Clear*, které umožňuje vymazat právě nahraný kalendář z aplikace.

4.2.2 Zobrazení týdne

V zobrazení týdne zůstává levá lišta stejná jako v zobrazení měsíce. V horní liště se místo sady tlačítek pro možnost přepínání mezi měsíci zobrazí sada tlačítek pro přepínání mezi týdny. Opět je zde možnost se přesunovat mezi týdny pomocí kolečka myši, nebo číslo týdne zadat manuálně. Zůstává tu také možnost přepínat rok.
Události jsou opět zobrazeny pomocí tabulky. V horní části každého sloupce jsou vypsány názvy dní v týdnu a jejich datumy. Víkendové dny jsou vyznačeny červeně.
Jednotlivé řádky odpovídají jednotlivým hodinám dne. Jelikož by se do pohledu všech 24 hodin nevešlo, je potřeba v pohledu listovat nahoru a dolů. Pohled ve výchozím nastavení začíná 10. hodinou.



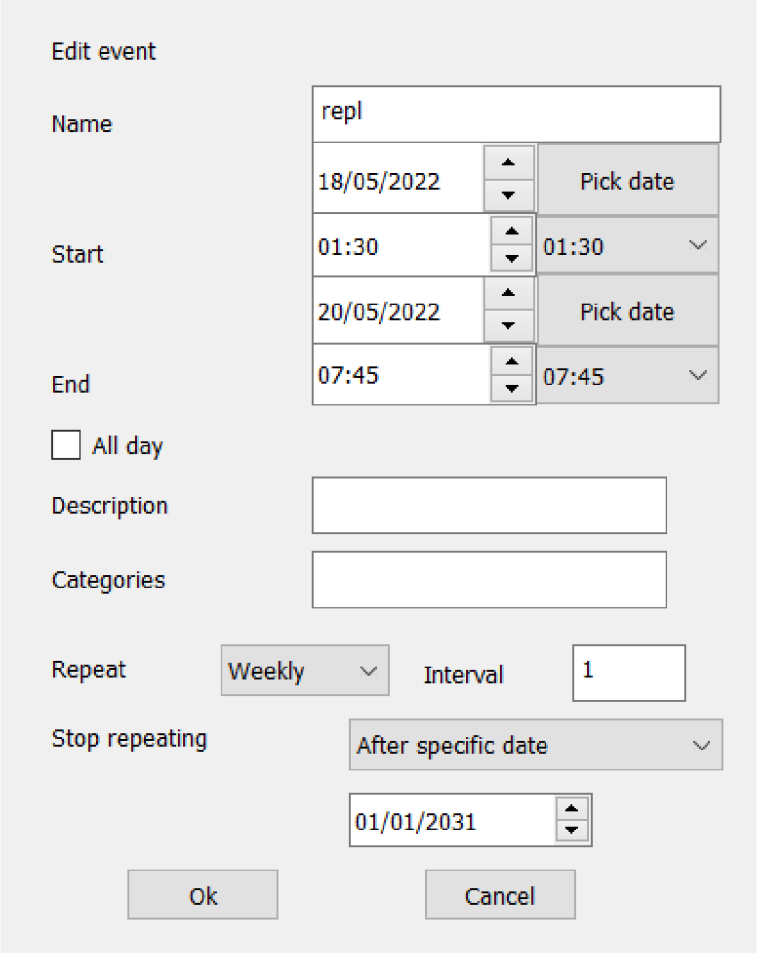
Obrázek 4.2: Týdenní zobrazení výsledné kalendářní aplikace

Události jsou v jednotlivých dnech zobrazeny v tabulce podle času, kdy začínají a končí, a je u nich vyznačen jejich název. Pokud se události v nějakém časovém úseku překrývají, jsou na tabulce automaticky zúženy tak, aby je bylo možné zobrazit v jednom sloupci vedle sebe. Protože takovýto způsob zobrazení přestává být praktický ve chvíli, kdy se překrývá velké množství událostí, je možné dvojklikem na název dne příslušný sloupec rozšířit tak, aby zabíral celou tabulku. Jedná se v podstatě o obdobu zobrazení dne, pro které však díky jeho identické struktuře se zobrazením týdne nebylo potřeba implementovat speciální

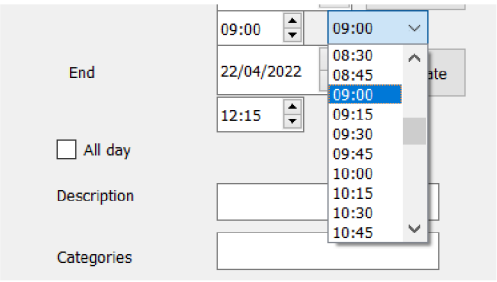
režim. Dalším dvojklikem na název dne se zobrazení vrátí do původního stavu. Po kliknutí na událost v tabulce se daná událost zvýrazní, a v levé liště se zobrazí podrobnosti o ní. Pod tabulkou se opět nachází tlačítka pro vytvoření, úpravu a odebrání události. Tento režim již neobsahuje pravou lištu s výčtem událostí v daném dni, jelikož jsou už všechny události i s jejich názvy přítomny v tabulce.

4.2.3 Okno pro vytváření a úpravu událostí

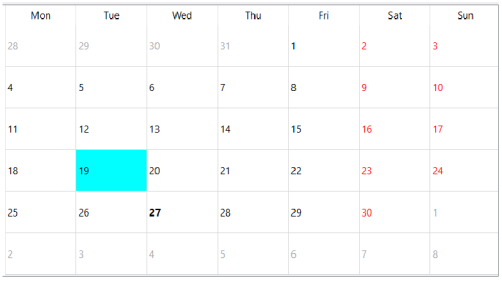
Do okna pro vytvoření/úpravu události se může uživatel dostat dvojím způsobem: Buďto vybere konkrétní událost/den a klikne na příslušné tlačítko, nebo okno otevře dvojklikem na příslušnou položku v tabulce. V případě úpravy existující události se do okna načtou odpovídající údaje o dané události. Při vytváření nové události se u obou způsobů otevření okna předvyplní datum začátku události na vybraný den. Podle vybrané konfigurace v nastavení se předvyplní také čas začátku události, a datum a čas konce události. Položka *all day* a volba opakování se nastaví na předem vybrané hodnoty (ve výchozím nastavení se zaškrtne kolonka *all day*, datum konce události se nastaví na stejný datum jako datum začátku události, čas začátku události se nastaví na současný čas, čas konce události na 1 hodinu od současného času, a opakování se zakáže).



Obrázek 4.3: Vytváření a úprava událostí ve výsledné kalendářní aplikaci



Obrázek 4.4: Výběr času ve výsledné kalendářní aplikaci

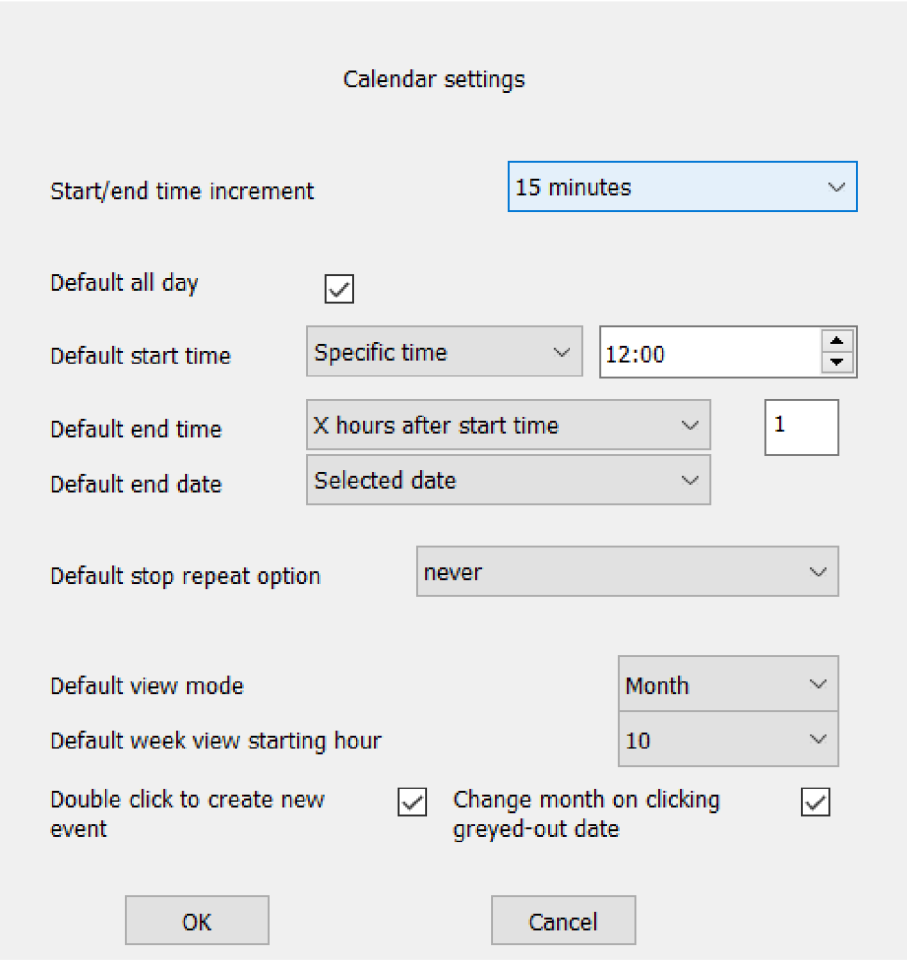


Obrázek 4.5: Výběr datumu ve výsledné kalendářní aplikaci

Datum lze vybrat stejnými způsoby, jako se vybírá datum v zobrazení měsíce. Kromě toho existuje i možnost vybrat potřebný datum přímo v kalendáři: Tlačítko *Pick date* přenese uživatele na zobrazení měsíce, kde může datum nastavit kliknutím na políčko v kalendářní tabulce. Při tomto pohledu nejsou zobrazovány ostatní události, místo toho je zvýrazněno druhé zvolené datum (konec při výběru začátku a začátek při výběru konce).
Čas se dá kromě klikání na šipky vybrat také kliknutím na seznam po pravé straně. Po kliknutí na něj se zobrazí seznam časových hodnot v patnáctiminutových intervalech (interval je možné změnit v nastavení), ve kterém je možné listovat. Po otevření okna nebo po manuální změně času se seznam nastaví na nejbližší vyšší hodnotu v seznamu (např. pokud je čas nastavený na 17:50 a v seznamu je interval 15 minut, tak se seznam nastaví na hodnotu 18:00, samozřejmě bez toho, aby v tuto chvíli došlo k přepsání vybraného času). Položky pro výběr času jsou viditelné pouze tehdy, když není zaškrtnutá volba *All day*.
V okně můžeme nastavit různé způsoby opakování aplikace: denní, týdenní, měsíční a roční. Pokud je některá z nich vybraná, objeví se i volba výběru toho, kdy se má daná událost přestat opakovat. Na výběr je několik základních možností (jako třeba příští měsíc nebo příští rok), popřípadě může uživatel nastavit konkrétní datum.
Okno také obsahuje tlačítko "Cancel", které zahodí provedené změny v události.

4.2.4 Okno nastavení

Okno s nastavením jsem se snažil nezahltit příliš velkým množstvím konfiguračních možností zejména proto, aby se v nich uživatel neztrácel, a aby jej nebylo potřeba členit do podoken. Většina z nabízených voleb upravuje výchozí hodnoty při vytváření nových událostí. Uživatel má možnost nastavit interval časových hodnot pro výběr času při vytváření/úpravě události, může určit, jestli mají ve výchozím nastavení události trvat celý den, a kdy se mají přestat opakovat. Dá se také nastavit výchozí začátek a konec událostí: Čas začátku lze nastavit buď na současný čas, nebo na konkrétní hodinu. Čas konce se dá nastavit buď na konkrétní čas, nebo na X hodin po čase začátku. Jde také nastavit, jestli mají události končit ve stejný den jako začínají, nebo X dní po začátku. Je zde také možnost nastavit, kterou hodinou začíná týdenní zobrazení (stále je možné pohled manuálně posunout na dřívější hodinu).
Kromě konfigurace okna pro vytváření událostí zde může uživatel také nastavit výchozí režim pohledu (buď zobrazení měsíce, nebo zobrazení týdne). Jde rovněž aktivovat a deaktivovat funkcionalitu, při které po kliknutí na den, který se ve vybraném měsíci nevyskytuje, dojde k přesunu do daného měsíce. Stejně tak lze aktivovat a deaktivovat funkcionalitu, která umožňuje otevření okna pro vytvoření události dvojklikem na dané místo v tabulce.



Obrázek 4.6: Okno s nastavením výsledné kalendářní aplikace

4.3 Testování aplikace

Testování probíhalo na notebooku s operačním systémem Windows 10, operační pamětí 16GB a procesorem AMD Ryzen 7 4800H s frekvencí 2.90 GHz. Webová verze aplikace byla testována na lokálním serveru (za použití stejného notebooku).

Nejprve bylo provedeno zátěžové testování. Pro kalendář byly náhodně vygenerovány události ve formátu *Icalendar* náhodně rozmístěné mezi roky 2020 a 2024. Kalendář byl spouštěn s postupně vzrůstajícím množstvím událostí a byla měřena jak doba prvního spuštění, tak doba potřebná pro načtení nového pohledu kalendáře (při změně datumu, modifikaci události apod.). K měření času byla využita *Qt* knihovna *<QElapsedTimer>*.

počet událostí	doba spuštění	odezva - měsíc	odezva - týden
100	74 ms	0 ms	2 ms
1000	99 ms	1 ms	3 ms
10 000	392 ms	1 ms	22 ms
50 000	1727 ms	2ms	182 ms
100 000	3398 ms	3 ms	456 ms

Tabulka 4.1: Výsledky zátěžového testování desktopové aplikace

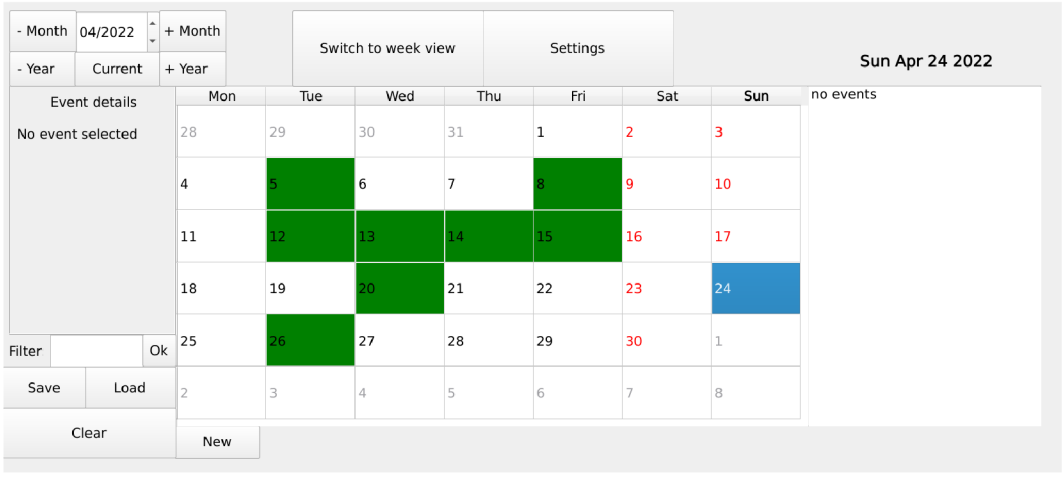
počet událostí	doba spuštění	odezva - měsíc	odezva - týden
100	27 ms	1 ms	3 ms
1000	118 ms	2 ms	4 ms
10 000	1438 ms	3 ms	47 ms
50 000	2928 ms	5 ms	442 ms
100 000	5232 ms	8 ms	1171 ms

Tabulka 4.2: Výsledky zátěžového testování webové aplikace

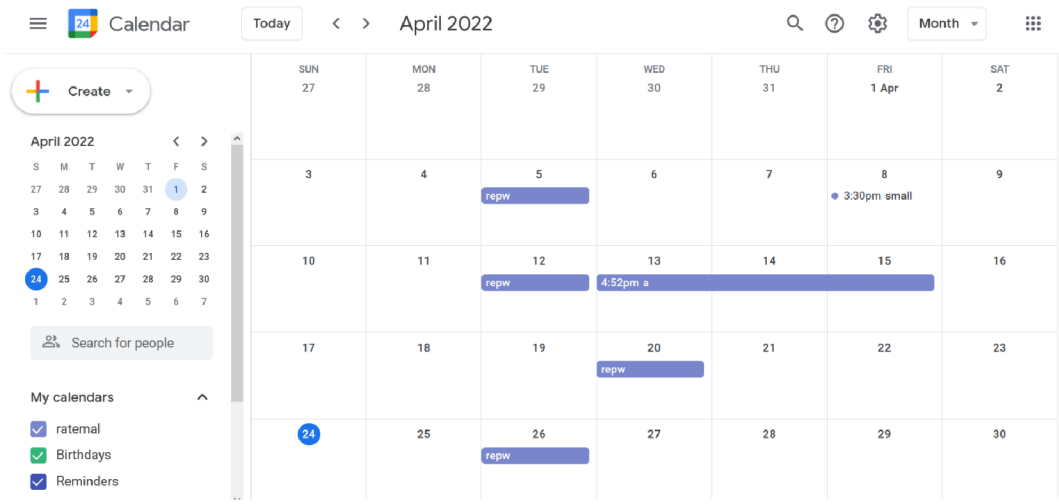
Aplikace pracuje velmi rychle a dokáže si poradit s mnohem větším množstvím dat, než kolik může uživatel reálně využít. I s obrovským množstvím dat se aplikace nenačítá déle, než několik sekund. Prodleva při přepínání mezi jednotlivými měsíci v režimu zobrazení měsíce je zanedbatelná bez ohledu na množství dat. U zobrazení týdne to už ovšem neplatí a prodleva u něj roste nelineární rychlostí. To je dáno poměrně složitým způsobem, jakým se události vykreslují do tabulky. Toto se však v praxi projeví až ve chvíli, kdy se každý den v týdnu děje několik desítek událostí zároveň, a kdy už týdenní zobrazení dávno přestává být praktické (u 100 000 záznamů v 5 letech je to cca. 380 událostí v jediném týdnu, což je cca. 54 událostí v jediném dni). Pokud bude mít uživatel o trochu méně náročný denní rozvrh, bude si moci plánovat unikátní události na desítky let dopředu bez znatelného zpomalení aplikace v libovolném režimu zobrazení.
Import nového kalendáře zabere přibližně stejnou dobu, jako načtení ekvivalentního seznamu událostí při spuštění aplikace. Vytvoření/úprava/odstranění události trvá při 100 000 záznamech necelou sekundu, při menším množství záznamů v kalendáři je prodleva zanedbatelná. Jedinou objevenou limitací aplikace je to, že databázový systém *IFDBS*, používaný pro ukládání událostí ve webové aplikaci, má omezenou velikost, která pojme jenom něco málo přes 100 000 událostí. Další události je stále možné nahrát, pouze se po zavření prohlížeče neuloží a nelze je exportovat. U desktopové aplikace žádné takové omezení není a nepodařilo se mi najít takové množství dat, které by aplikace nedokázala zpracovat.

Původně jsem měl v plánu provést přibližné porovnání s aplikací *Google Calendar*, jenomže tato aplikace už při 10 000 událostech nedokázala data vůbec zpracovat, a i soubor s 1000 událostmi aplikace zpracovávala téměř celou minutu.

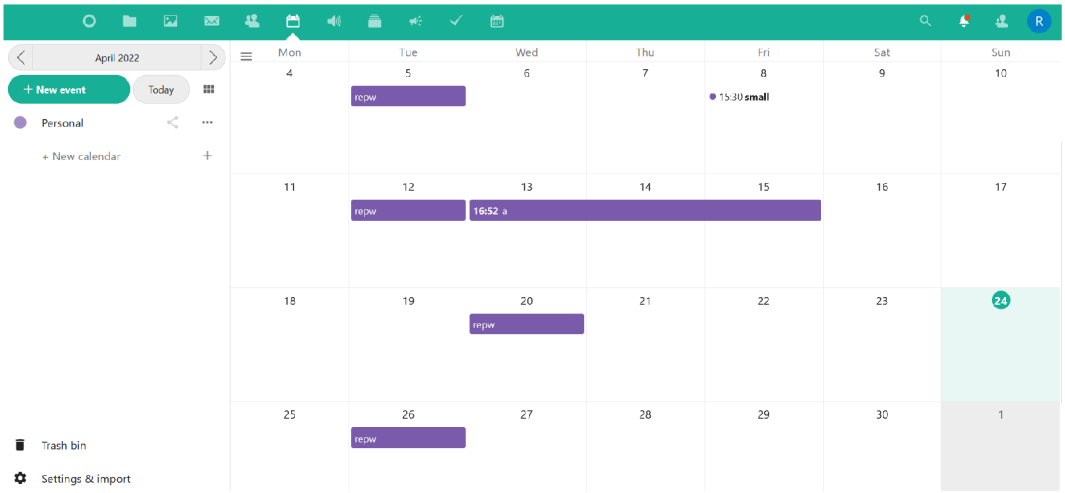
Další test byl zaměřen na kontrolu validity kalendářních dat a jejich kompatibility s ostatními kalendářními aplikacemi, používajícími standard *ICalendar*. Pro kontrolu byly využity aplikace Kalendář *Google* a *Nextcloud Calendar*, zmíněné ve 2. kapitole.
Nejprve bylo ve výsledné aplikaci manuálně vytvořeno několik událostí tak, aby využily co nejvíce možných funkcionalit kalendáře: Události trvající celý den, události, které trvají několik dní a přitom začínají v konkrétní čas, události s různými způsoby opakování, apod. Poté byly identické události vytvořeny v kalendáři *Google*. Z výsledné aplikace i z kalendáře *Google* byly následně exportovány jejich seznamy událostí a nahrány druhou aplikací (takže výsledná aplikace nahrála data z kalendáře *Google* a kalendář *Google* nahrál data z výsledné aplikace). Na obou aplikacích bylo poté porovnáno zobrazení událostí z jejich vlastních dat a z dat druhé aplikace. Tento proces byl poté zopakován s aplikací *Nextcloud Calendar*, a byl proveden jak pro desktopovou, tak pro webovou verzi aplikace.



Obrázek 4.7: Testovací události exportované z kalendáře *Google* a následně zobrazené ve webové verzi výsledného kalendáře



Obrázek 4.8: Testovací události exportované z výsledné aplikace a následně zobrazené v kalendáři *Google*



Obrázek 4.9: Testovací události exportované z výsledné aplikace a následně zobrazené v kalendáři *Nextcloud*

Z výsledku testu vyplývá, že je výsledná aplikace kompatibilní se standardem *ICalendar*. Kalendář dokáže sdílet validní data s ostatními kalendářními aplikacemi a dokáže zpracovávat kalendářní soubory jimi vytvořené, a to jak ve webové, tak v desktopové verzi.

Kapitola 5

Závěr

Cílem bakalářské práce bylo prozkoumat existující kalendářní aplikace a standardy v nich používané, a na základě tohoto průzkumu vytvořit vlastní kalendářní aplikaci, kterou lze použít jak v podobě desktopové aplikace, tak v podobě webové aplikace využívající prostředí *WebAssembly*, a kterou lze snadno ovládat pomocí efektivního uživatelského rozhraní.
Za tímto účelem bylo prozkoumáno několik běžně používaných kalendářních aplikací, přičemž jako hlavní inspirace posloužil kalendář *Google*. Zároveň byl prostudován standard *ICalendar*, který udává formu zápisu kalendářních dat, a zajišťuje kompatibilitu při sdílení dat mezi různými kalendářními aplikacemi. Bylo analyzováno také prostředí *WebAssembly*, ve kterém pracuje webová verze aplikace.
Na základě získaných poznatků bylo navrženo snadno ovladatelné grafické uživatelské rozhraní vlastní aplikace, a definována její grafická podoba. Poté byla navržena knihovna zprostředkující funkce nutné k ovládání této aplikace.
Aplikace byla následně implementována v jazyce *C++17* s pomocí frameworku *Qt* verze 5.15 pro tvorbu grafického rozhraní, a pro webovou verzi také pomocí nástroje *Emscripten* pro přeložení zdrojového kódu do prostředí *WebAssembly*. Při implementaci bylo využito několika knihoven podporovaných frameworkem *Qt*, například na práci s datumem a časem, nebo ke generování *UID*, a plugin sloužící k vytvoření virtuálního souborového systému u webové aplikace. Poté byla aplikace patřičně otestována.
Výsledkem je rychlá a snadno použitelná kalendářní aplikace s intuitivním uživatelským rozhraním, která je kompatibilní se současným kalendářním standardem *ICal*, a dokáže snadno pracovat i s velkým množstvím kalendářních dat.
Další rozšíření aplikace by mohlo vést k vytvoření klient—server architektury a automatickému sdílení dat mezi uživateli, nebo k možnosti přidávat k událostem zeměpisné souřadnice pro vyznačení místa konání na mapě. Přitom by však stále měl být kladen důraz na jednoduchost používání aplikace. Další možností by také mohlo být přepsání knihovny pomocí jazyka *C++20*, aby samotná knihovna nebyla závislá na frameworku *Qt*.

Literatura

[1] *ICalendar.org - iCalendar Resources, Specifications and Tools* [online]. Z Content [cit. 2022-04-25]. Dostupné z: https://icalendar.org/.

[2] C. Daboo, L. D. *CalDAV Access (RFC 4791) | RFC Specifications* [online]. Z Content, 2007 [cit. 2022-04-25]. Dostupné z: https://icalendar.org/RFC-Specifications/CalDAV-Access-RFC-4791/.

[3] *Emscripten 3.1.9-git (dev) documentation* [online]. [cit. 2022-04-25]. Dostupné z: https://emscripten.org/.

[4] *About: Google Calendar* [online]. [cit. 2022-04-25]. Dostupné z: https://dbpedia.org/page/Google_Calendar.

[5] Nestor, M. *Canonical, Collabora, and Nextcloud Deliver Work From Home Solution to Raspberry Pi Users - 9to5Linux* [online]. 9to5Linux, 2021 [cit. 2022-04-25]. Dostupné z: https://9to5linux.com/canonical-collabora-and-nextcloud-deliver-work-from-home-solution-to-raspberry-pi-users.

[6] PVince81. *Nextcloud/server: Nextcloud server, a safe home for all your data* [online]. GitHub, Inc., 2022 [cit. 2022-04-25]. Dostupné z: https://github.com/nextcloud/server.

[7] GmbH, N. *Nextcloud - Regain control over your data* [online]. [cit. 2022-04-25]. Dostupné z: https://nextcloud.com/.

[8] Béla, B. *QOrganizer official website* [online]. [cit. 2022-04-25]. Dostupné z: http://qorganizer.sourceforge.net/.

[9] *Qt | Cross-platform software development for embedded and desktop* [online]. Qt Group [cit. 2022-04-25]. Dostupné z: https://www.qt.io/.

[10] *Embedded Software Development Tools | Cross Platform IDE | Qt Creator* [online]. The Qt Company [cit. 2022-04-25]. Dostupné z: https://www.qt.io/product/development-tools.

[11] *Calendar integration - Nextcloud latest Developer Manual latest documentation* [online]. Nextcloud GmbH, 2022 [cit. 2022-04-25]. Dostupné z: https://docs.nextcloud.com/server/latest/developer_manual/digging_deeper/groupware/calendar.html.

[12] Frank Dawson, A. G. *RFC 2445 - Internet Calendaring and Scheduling Core Object Specification (iCalendar)* [online]. Internet Engineering Task Force, 1998 [cit. 2022-04-25]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc2445.

[13] Dusseault, L. *RFC 4918 - HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)* [online]. Internet Engineering Task Force, 2007 [cit. 2022-04-25]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc4918.

[14] Desruisseaux, B. *RFC 5545 - Internet Calendaring and Scheduling Core Object Specification (iCalendar)* [online]. Internet Engineering Task Force, 2009 [cit. 2022-04-25]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc5545.

[15] Daboo, C. *RFC 7986 - New Properties for iCalendar* [online]. Internet Engineering Task Force, 2016 [cit. 2022-04-25]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc7986.

[16] *WebAssembly Concepts - WebAssembly | MDN* [online]. Mozilla Foundation, 2022 [cit. 2022-04-25]. Dostupné z: https://developer.mozilla.org/en-US/docs/WebAssembly/Concepts.

[17] *WebAssembly* [online]. [cit. 2022-04-25]. Dostupné z: https://webassembly.org/.

[18] *Calendar* [online]. Microsoft, 2022 [cit. 2022-04-25]. Dostupné z: https://support.microsoft.com/en-us/office/calendar-d8ad42a0-c0a7-4d18-ae31-a2b86ec64a92.