Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky

Konfiguračné nástroje pre Raspberry Pi

Bakalárska práca

2020 **Gabriel Adamčák**

Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky

Konfiguračné nástroje pre Raspberry Pi

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	9.2.1. Informatika
Školiace pracovisko:	Katedra počítačov a informatiky (KPI)
Školiteľ:	Ing. Miroslav Biňas PhD.
Konzultant:	Ing. Miroslav Biňas PhD.

Košice 2020 **Gabriel Adamčák**

Názov práce:	Konfiguračné nástroje pre Raspberry Pi
Pracovisko:	Katedra počítačov a informatiky, Technická univerzita v Košiciach
Autor:	Gabriel Adamčák
Školiteľ:	Ing. Miroslav Biňas PhD.
Konzultant:	Ing. Miroslav Biňas PhD.
Dátum:	13. 5. 2020
Kľúčové slová:	LaTeX, programovanie, sadzba textu

Abstrakt: Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Thesis title:	Configuration tools for Raspberry Pi
Department:	Department of Computers and Informatics, Technical University of Košice
Author:	Gabriel Adamčák
Supervisor:	Ing. Miroslav Biňas PhD.
Tutor:	Ing. Miroslav Biňas PhD.
Date:	13. 5. 2020
Keywords:	LaTeX, programming, typesetting

Abstract: Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Tu vložte zadávací list pomocou príkazu
`\thesisspec{cesta/k/suboru/so/zadavacim.listom}`
v preambule dokumentu.

Čestné vyhlásenie

Vyhlasujem, že som záverečnú prácu vypracoval(a) samostatne s použitím uvedenej odbornej literatúry.

Košice, 13.5.2020 .....

Vlastnoručný podpis

Poďakovanie

Na tomto mieste by som rád poďakoval svojmu vedúcemu práce za jeho čas a odborné vedenie počas riešenia mojej záverečnej práce.

Rovnako by som sa rád poďakoval svojim rodičom a priateľom za ich podporu a povzbudzovanie počas celého môjho štúdia.

V neposlednom rade by som sa rád poďakoval pánom *Donaldovi E. Knuthovi* a *Leslie Lamportovi* za typografický systém LaTeX, s ktorým som strávil množstvo nezabudnuteľných večerov.

Obsah

Zoznam obrázkov

Motivácia

Cieľom bakalárskej práce je vytvorenie konfiguračných nástrojov pre Raspberry Pi, t.j. sprievodcu po prvom spustení a nastavenia systému, ktoré sú dostupné kedykoľvek počas behu operačného systému.

Sprievodca po prvom spustení sa už v systéme nachádza, bohužiaľ je nepriehľadný a jeho rozšírenie si vyžaduje zásah do zdrojového kódu. Preto výsledkom tejto práce bude priehľadný a jednoducho rozšíriteľný sprievodca.

Nastavenia v systéme sami o sebe implementované niesú, sú porozhadzované a väčšina z nich sa musí vykonať v termináli. Výsledkom bude jednotná aplikácia, v ktorej tieto nastavenia budú združené a jednoducho rozšíriteľné.

Tieto nástroje budú napísané v jazyku Python verzie 3. Pre grafické zobrazenie som sa rozhodol použiť knižnicu GTK3 keďže ju podporuje väčšina linuxových distribúcií.

1 Formulácia úlohy

Text záverečnej práce musí obsahovať kapitolu s formuláciou úlohy resp. úloh riešených v rámci záverečnej práce. V tejto časti autor rozvedie spôsob, akým budú riešené úlohy a tézy formulované v zadaní práce. Taktiež uvedie prehľad podmienok riešenia.

2 Raspberry Pi

2.1 História

S príchodom automatizácie a každoročného zmenšovania komponentov sme dos peli k tomu, že plnohodnotný počítač dokážeme chytiť do dlane. Jedným z takých je aj počítač o veľkosti platobnej karty, Raspberry Pi.

Prvá verzia pomenovaná jednoducho Raspberry Pi prišla na tento svet 29. Februára 2012 a každým rokom vznikajú vylepšené verzie pri nezmenených rozmeroch. Vo svete existuje už jeho štvrtá verzia, kde sme sa posunuli zas o kúsok ďalej. Ako uvádzajú na oficiálnych stránkach *„Prvýkrát sme vytvorili kompletný zážitok z využívania počítača. Či už upravujete dokumenty, prehliadate web, vytvárate tabuľky alebo pripravujete prezentáciu, zážitok bude plynulý a veľmi rozpoznateľný - ale na menšom, energeticky úspornejšom a oveľa nákladovo efektívnejšom stroji“*[9]. Hlavnou výhodou je aj jeho cena, ktorá sa pohybuje na úrovni 35tich dolárov.

2.1.1 Raspberry Pi v školách

Hlavným cieľom tvorby Raspberry Pi bolo priniesť späť počítačovú výučbu na školách pre vzdelávanie mladých ľudí, ako píše web raspberrytips.com[1]. Aj na Technickej Univerzite v Košiciach je tomu daný priestor a pre tretiakov je pripravený predmet Základy internetu vecí, kde majú študenti možnosť oboznámiť sa s touto platformou, skúšať a prototypovať si nové veci v novom laboratóriu pre oblasť internetu vecí[11]. Taktiež sa s tým môžeme stretnúť na stredných, poprípade základných školách, kde sa pre mladších žiakov dajú vytvárať zaujímavé veci pomocou programovacieho jazyka Scratch. No a pre tých viacej zdatných je tu programovací jazyk Python, Java alebo C.

2.1.2 Raspberry Pi vo firmách

Vo firmách je najčastejšie využitie pre automatizovanie vecí, či už je to zber dát, práca s nimi a následné vyhodnocovanie alebo spracovávanie obrazu pomocou kamery, napríklad ako elektronický vrátnik. Pomocou strojového učenia by zariadenie vedelo na základe údajov rozhodnúť, kto má prístup do firmy a kto nie. Taktiež je ho možné použiť ako jednoduchý server v lokálnej sieti, napríklad ako zálohovací. Pre zamestnancov by sa potrebné dáta pri pokazení alebo odsudzení počítača zachovali.

2.1.3 Iné riešenia

Raspberry Pi sa taktiež dá používať ako stolný počítač pre menej náročných ľudí, na ovládanie robotov alebo aj ako kontrolný prvok inteligentnej domácnosti. Inteligentná domácnosť, tento pojem sa do povedomia ľudí dostáva čoraz viac a viac a s Raspberry Pi si ju užívateľ môže vyskladať podľa seba a za nízku cenu. Použitia Raspbeery Pi sú neobmedzené a jediné na čom závisia, je kreativita užívateľa.

2.1.4 Rozhranie systému

Rozhranie systému, tak ako aj v iných operačných systémoch, môžme rozdeliť na dva typy - textové a grafické.

Grafické rozhranie

Rozhranie systému je jednoduché, podobá sa na väčšinu linuxových distribúcii a je postavené na operačnom systéme Debian. Je navrhnuté pre stabilitu, aj keď neustále dochádza k zlepšovaniu výkonu softvéru aj prostredia, stále však zostáva veľa pre to, aby sa dalo použiť pre výkonného používateľa, ako uvádza Abishek Muthian vo svojom blogu[8]. Vstupom pre toto rozhranie môže byť klávesnica, myš poprípade dotyk alebo zvuk, ak máme pripojené adekvátne zariadenia alebo senzory.

Textové rozhranie

Je textovo založené, slúži na ovládanie softwéru a operačného systému a zároveň umožňuje užívateľovi reagovať na textové výzvy zadaním jednotlivých príkazov

do rozhrania a prijatím odpovede rovnakým spôsobom[2]. Vstupom pre toto rozhranie je iba klávesnica. V počiatkoch to bolá jediná možnosť ako komunikovať s počítačom.

2.2 Nástroje

Pi v názve Raspberry Pi je referencia na časť Python z prvých vytvorených zariadení, ako uvádza Patrick Fromaget [1]. Taktiež uvádza, že tieto zariadenia začínali iba s príkazovým riadkom, kde užívateľ musel zadať Python kód, ktorý chcel využiť.

Väčšina nástrojov je ale aj tak napísaných v jazyku C alebo v jeho rozšírení C++. Jazyk C a C++ sú kompilačné, to znamená, že pre vygenerovanie zdrojového spustiteľného súboru potrebujú kompilátor[1]. Medzi ich hlavné výhody patrí rýchlosť a účinnosť. Medzi nevýhody patria statické premenné a správa pamäte keďže pre každú premennú je potrebné pri použití samostatne alokovať miesto v pamäti. Taktiež je potrebné pri ukončení práce s premennou uvoľniť miesto v pamäti.

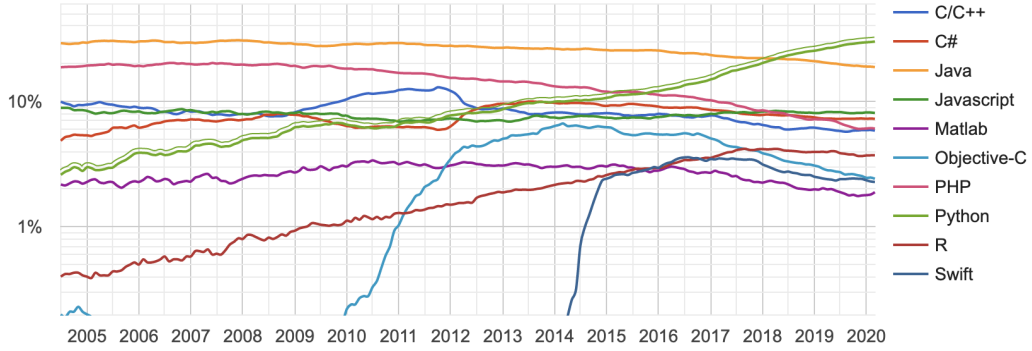
Python, ako vyšší programovací jazyk, potrebuje pre svoju funkčnosť interpreter[2]. Medzi jeho výhody patrí jednoduchosť a interaktívnosť. Oproti jazyku C/C++ je pomalší, nakoľko premenné sú dynamicky alokované a správu pamäte si riadi sám.

2.2.1 Popularita programovacích jazykov

Ako vyplýva z grafu a následne aj tabuľky programovací jazyk python aktuálne patrí medzi najviac populárne programovacie jazyky. Vďačí tomu najme svojej jednoduchosti a čitateľnosti kódu. Strojové učenie, umelá inteligencia (AI), veľké dáta a robotika sa spoliehajú vo veľkej miere na jazyk Python. Pri jazyku python nie ste viazaní na jednu platformu.

Prekvapivo si však mnohí vývojári nevyberajú Python ako svoj primárny jazyk, pretože je také ľahké sa ho naučiť a používať, vybrali si ho ako druhý alebo až tretí jazyk. To môže byť ďalší dôvod, prečo je medzi vývojármi tak populárny.

[1] Program, ktorý preloží vysoko úrovňový zdrojový kód na binárny kód[3].
[2] Aplikácia, na ktorej priamo beží skript napísaný v Pythone[12].



Obr. 2.1: Popularita programovacích jazykov[5]

Celosvetová tabuľka popularity programovacích jazykov (Mar. 2020)			
Poradie	**Programovací jazyk**	**Popularita**	**Nárast**
1	Python	30.09 %	+3.9 %
2	Java	18.84 %	-1.7 %
3	Javascript	8.1 %	-0.1 %
4	C#	7.27 %	-0.0 %
5	PHP	6.08 %	-1.1 %
6	C/C++	5.86 %	-0.2 %
7	R	3.73 %	-0.2 %
8	Objective-C	2.42 %	-0.5 %
9	Swift	2.28 %	-0.1 %
10	Matlab	1.89 %	-0.1 %

2.2.2 Sprievodca po prvom spustení

Sprievodca po prvom spustení sa už v systéme nachádza,ale z nášho pohľadu nie je používateľsky veľmi prívetivý. Sprievodcu je možné nechtiac zavrieť, čo môže mať následky. Ak ho užívateľ neukončí korektne, sprievodca sa spustí znova až po reštarte systému. Taktiež je ho možné opakovane spustiť z príkazoveho riadku čo si myslíme že je pre neskúseného užívateľa vcelku náročné.

Riešením bymohlo byť vytvorenie prívetivého jednoduchého prostredia, ktorým by si užívateľ pri prvom spustení musel prejsť, ktoré by pôsobilo dobrým dojmom hneď na začiatku a ktoré by bolo modulárne a jednoducho rozšíriteľné, nakoľko existujúci sprievodca nieje veľmi užívateľsky prívetivý je napísaný v jednom jedinom zdrojovom súbore a nedá sa jednoducho rozšíriť.

Obr. 2.2: Existujúci sprievodca po prvom spustení

Sprievodca po prvom spustení v operačnom systéme Windows

Ak prvýkrát spustíme systém Windows na celej obrazovke sa nám zobrazí sprievodca po prvom spustení bez možnosti ukončenia alebo preskočenia. Používateľ musí nastaviť základné nastavenia systému ako napríklad meno, heslo, klávesnicu región alebo čas. Systém sa na základe týchto nastavení prednastaví aby bolo jednoduchšie s ním pracovať.

V novších verziách operačného systému je tento sprievodca ladený do modrej a nieje možné ho zavrieť aleb preskočiť.

Sprievodca po prvom spustení v operačnom systéme OSX

Sprievodca je ladený do sivej farby a je používateľsky prívetivý. Taktiež je v sprievodcovi možné migrovať nastavenia zo staršieho zariadenia. Na výber máme z týchto troch možností:

- z lokálneho disku
- z počítača s operačným systémom Windows
- z počítača s operačným systémom OSX

Je to zaujímavý nastroj keďže väčšinou sa stáva že máme starý počítač a potrebujeme presunúť potrebné veci do nového počítača. Tento nástroj nám tento postup vie rapídne urýchliť. Taktiež ako v operačnom systéme Windows ani tento sprevodca po prvom spustení sa nedá preskočiť.

Sprievodca po prvom spustení v operačnom systéme Fedora

Tento sprievodca ponúka možnosť pripojiť online konto k vytváranému kontu. Na výber je možnosť pridať Google, Nextcloud, Microsoft alebo Facebook konto. Taktiež je v ňom možné zapnutie alebo vypnutie lokalizačných služieb.

Sprievodca po prvom spustení v operačnom systéme Ubuntu

Sprievodca po prvom spustení sa v operačnom systéme Ubuntu nenachádza. Všetky nastavenia sa nastavujú počas inštalácie operačného systému. Je to úplne iný spôsob ako v ostatných operačných systémoch. Používateľ musi pri inštalácií ostať pri počítači.

2.2.3 Nastavenia systému

V operačnom systéme sa bohužiaľ nenachádzajú jednotné nastavenia systému, takže pre bežného používateľa je veľmi náročné niečo nastaviť. Snahou je vytvoriť jednu aplikáciu ktorá by združovala nastavenia, tak ako je to aj v iných operačných systémoch. Používateľ by bol aj bez väčších zručností schopný nastaviť si čo potrebuje, kedykoľvek chce. Týmto krokom by sa rozšírila platforma medzi viacej užívateľov, ktorí pri používaní nemajú čas, nerozumejú a zároveň sa im nechce hľadať v dokumentácií ktorým príkazom by v termináli nastavili potrebné veci v systéme.

Príkladom ako by to správne malo vyzerať sú nastavenia systému v operačných systémoch Windows, OSX, Fedora alebo Ubuntu. Nastavenia sú jednoduché, priehľadné, prívetivé a umožňujú používateľovi prispôsobiť a nakonfigurovať operačný systém.

Nastavenia systému v operačnom systéme Windows

V novších verziách operačného systému Windows boli nastavenia systému úplne prerobené. Ponúkajú jednoduché menu rozdelené do viacerých kategórií. Po klik-

nutí na danú kategóriu sa presunieme na zvolenú stránku. Ak chceme ísť na inú stránku s nastaveniami musíme sa vrátiť späť a vybrať inú.

Ani v týchto nastaveniach nenájdeme všetko potrebné. V operačnom systéme sa momentálne nachádzajú aj staršie nastavenia systému, v ktorých je možné nastaviť všetko potrebné. V budúcich verziách operačneho systému sa ale tieto nastavenia plánujú úplne spojiť do jednej aplikácie.

Nové nastavenia sú oproti starším oveľa priehľadnejšie a intuitívnejšie.

Nastavenia systému v operačnom systéme OSX

V operačnom systéme OSX sú nastavenia systému podobné ako v operačnom systéme Windows a oproti nastaveniam v operačnom systéme Windows sú rozdelené do viacero menších kategórii.

Pri väčšom zásahu do operačného systému vyžadujú zadanie hesla. Je to ďalší faktor zabezpečenia a neumožnuje to neoprávnenému používateľovi vykonávať zmeny v nastaveniach.

Nastavenia systému v operačnom systéme Fedora

Oproti operačným systémom Windows a OSX nastavenia systému v operačnom systéme Fedora vyzerajú úplne inak. Na ľavej strane je zoznam s nastaveniami a v druhej časti okna už nachádzajú nastavenia konkrétneho výberu. Podľa toho čo sa vyberie z ľavej strany menu sa prispôsobí pravá strana. Takže ak vyberieme možnosť nastavenie pozadia tak na pravej strane sa nám objavia obrázky, ktoré môžme použiť ako pozadie pracovnej plochy.

Nastavenia systému v operačnom systéme Ubuntu

V novších verziách operačného systému Ubuntu sa nastavenia systému podobajú na tie z operačného systému Fedora. Taktiež tu nájdeme postranný menu panel s výberom možností nastavení a na pravej strane sa nám zobrazí stránka s nastavením, ktoré sme si zvolili.

V starších verziách sa nastavenia podobali na tie z operačného systému OSX.

2.3 Používateľské rozhranie

„Navrhovanie jednoduchého a jasného objektu trvá minimálne dvakrát tak dlho ako obvyklým spôsobom"[15].

Používateľské rozhranie je bod medzi ľudskou alebo počítačovou interakciou a komunikáciou v zariadení. Môže to zahŕňať obrazovku, klávesnicu, myš a vzhľad pracovnej plochy[3] . Je to zároveň cesta, ktorou užívateľ interaguje s aplikáciou[10]. Je to najdôležitejšia časť každého operačného systému. Dobre navrhnuté používateľské rozhranie a obrazovka sú pre používateľov nesmierne dôležité. Taktiež je oknom na zobrazenie schopností systému, mostom k schopnostiam softvéru. Pre mnohých používateľov je používateľské rozhranie systém, pretože je to jedným z mála viditeľných komponentov produktu, ktorý vývojári vytvárajú.

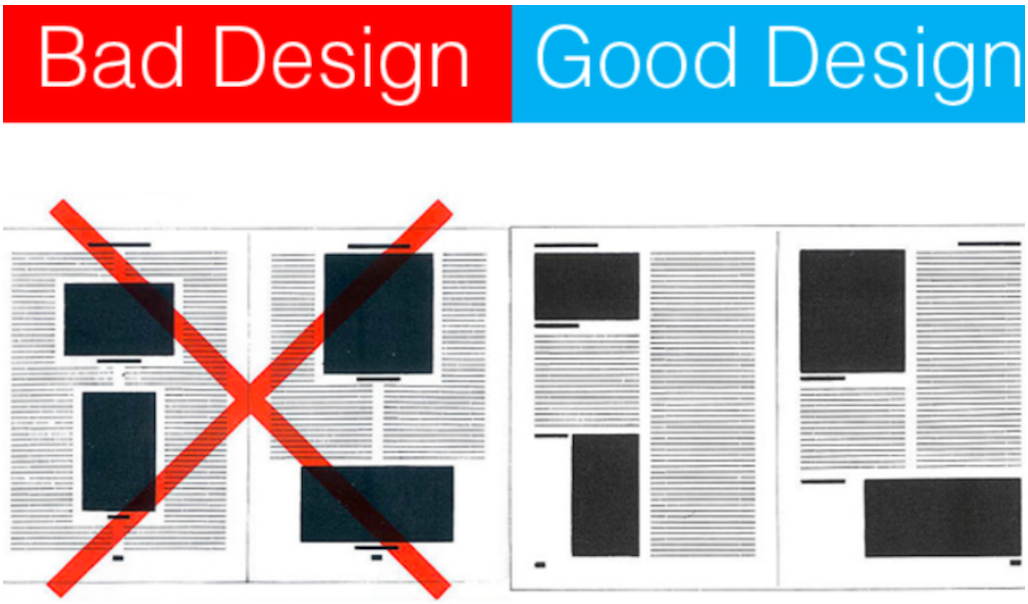
Rozloženie, vzhľad a navigácia ovplyvňujú používateľov viacerými spôsobmi. Pokiaľ sú zmätený, majú väčší problém so spravením úlohy ktorú chcú vykonať a spôsobuje to oveľa viac zbytočných chýb [13].

2.3.1 Najčastejšie chyby pri tvorbe používateľského rozhrania

- **nekonzistentný štýl aplikácie** - vynikajúci dizajn používateľského rozhrania by mal zodpovedať štýlu, aby používatelia jasne porozumeli danému obsahu a odpovedali naň.
- **ne-responzívny dizajn** - dizajn aplikácie by mal byť vytvorený tak, aby sa vedel prispôsobiť na rôzne rozlíšenia obrazovky.
- **dizajn neorientovaný na užívateľa** - pri vytváraní dizajnu je treba brať do úvahy to, aby naše nápady boli prijaté a pochopené užívateľom.
- **frustujúce chyby** - chyby by mali jednoducho vysvetliť čo sa stalo a ako ich môže užívateľ opraviť. Taktiež je ich dobré ukázať blízko miesta kde chyba vznikla.

Zmysel používateľského rozhranie je doručiť produkt včasným a elegantným spôsobom[7].

[3]Priestor na obrazovke displeja, ktorý reprezentuje veci, ktoré by mohli byť na fyzickom stole.



Obr. 2.3: Príklad dobrého a zlého dizajnu[7]

2.3.2 Grafické aplikácie v jazyku Python

Pri tvorbe grafickej aplikácie v jazyku Python máme na výber z viacero grafických knižníc.

- **Kiwi** - je OpenGL ES 2[4] akcelerovaný rámec na vytváranie nových používateľských rozhraní , ktorý umožňuje ľahko napísať kód a spustiť ho na rôznych platformách alebo operačných systémoch (Windows, MacOSX, Linux, Android iOS a Raspberry Pi). Je voľne dostupný a obsahuje viac ako 20 widgetov vo svojej súprave nástrojov. Tento rámec je profesionálne vyvinutý, podporovaný a používaný a je ho možné používať aj v komerčnej sfére. Rámec je stabilný a má dobrú dokumentáciu. Obsahuje sprievodcu programovaním, ktorý vám pomôže začať.

- **PyQT** - je väzba jazyka Python pre Qt[5], čo je sada knižníc a vývojových nástrojov jazyka C++, ktoré zahŕňajú abstrakcie pre grafické užívateľské rozhrania nezávislé od platformy. Je voľne dostupný ale niektoré funkcie v ňom nie sú k dispozícii zadarmo. Qt je dobre zavedená v komunite vývojárov a má nástroje, ktoré to odrážajú. Písanie Python aplikácií v Qt znamená, že

[4]https://www.khronos.org/opengles/
[5]https://www.qt.io/why-qt

máte prístup k QtCreatoru[6], ktorý obsahuje návrhársky režim na generovanie kódu pre rozloženie vašej aplikácie.

- **Tkinter** - je bežne dodávaný s jazykom Python a je štandardným grafickým rozhraním pre Python. Je obľúbený pre svoju jednoduchosť a grafické užívateľské rozhranie. Je to voľne dostupný a je k dispozícii na základe Python licencie.

- **WxPython** - je multiplatformová sada grafického používateľského rozhrania pre programovací jazyk Python. Je implementovaný ako sada rozširujúcich modulov jazyka Python, ktoré obaľujú grafické komponenty populárnej medziplatformovej knižnice wxWidgets. Umožňuje jednoducho a ľahko vytvárať programy s vysoko funkčným grafickým používateľským rozhraním. Keďže programovacím jazykom je Python, programy so sadou wxPython sú jednoduché, ľahko zapisovateľné a ľahko pochopiteľné.

  Rovnako ako Python a wxWidgets, wxPython je voľne dostupný, čo znamená, že ho môže používať každý a zdrojový kód je k dispozícii pre kohokoľvek. Ktokoľvek si ho môže pozerieť a upraviť ho taktiež na opravy a vylepšenia môže prispieť ktokoľvek.

- **PyGtk** - PyGtk je sada väzieb jazyka Python na populárnu súpravu nástrojov GTK[7]. V kombinácií s Glade grafickým konštrukérom poskytuje efektívny spôsob zrýchleného vývoja aplikácií.

  GTK je vhodný pre malé aplikácie až po kompletné aplikačné balíčky. Projekty postavené pomocou sady GTK a jej závislostí fungujú bezproblémovo na známych operačných systémoch. GTK je voľne dostupný projekt s otvoreným zdrojovým kódom, ktorý spravuje GNOME[8] a aktívna komunita prispievateľov.

  GTK poskytuje mnoho funkcií, ako je natívny vzhľad, podpora tém, objektovo orientovaný prístup, ktorý dnešní vývojári hľadajú v súprave nástrojov.

[6] https://doc.qt.io/qtcreator/index.html
[7] https://www.gtk.org/
[8] https://www.gnome.org/

2.3.3 Návrhové vzory pri tvorbe aplikácie

Návrhový vzor je zdokumentované najlepšie riešenie alebo časť riešenia, ktoré bolo úspešne aplikované vo viacerých prostrediach na vyriešenie problému, ktorý sa opakuje v špecifickom súbore situácií.

Návrhový vzor je efektívnym prostriedkom na sprostredkovanie / komunikáciu toho, čo sme sa dozvedeli o kvalitných dizajnoch. Výsledkom je:

- Spoločný jazyk na sprostredkovanie skúseností získaných pri jednaní s týmito opakujúcimi sa problémami a ich riešeniami.
- Bežná slovná zásoba prvkov návrhu systému na riešenie problémov diskusií. Prostriedok na opätovné použitie a nadviazanie na získané poznatkyčo vedie k zlepšeniu kvality softvéru, pokiaľ ide o jeho udržiavateľnosť a opätovné použitie.

Návrhové vzory nie sú teoretické konštrukty. Návrhový vzor môže byť reprezentovaný ako zapuzdrenie znovu použiteľného riešenia, ktoré bolo úspešne použité na vyriešenie bežného problému.[14]

Rozdelenie návrhových vzorov:

- **Vytváracie vzory** - umožňujú nám vytvárať objekty na základe požadovaných kritérií a kontrolovaným spôsobom
- **Štrukturálne vzory** - týkajú sa usporiadania rôznych tried a objektov, aby sa vytvorili väčšie štruktúry a poskytli nové funkcie
- **Vzory správania** - týkajú sa identifikácie bežných vzorov komunikácie medzi objektmi a realizácie týchto vzorov

3 Metodológia

Pri tvorbe grafickej aplikácie máme dve možnosti:

- - načítať všetky obrazovky naraz pri spustení aplikácie
- - Načítať danú obrazovku a pri zmene načítať dalšiu

Každá z týchto možností má svoje výhody ale aj nevýhody.

Pri prvej možnosti sa stánky načítavajú na začiatku takže čas spustenia aplikácie je razantne dlhší. Taktiež sú zvýšené nároky na výkon a pamäť zariadenia čo v prípade menej výkonných zariadení môže spôsobiť problém.

Pri druhej možnosti sa aplikácia spúšťa rýchlo kedže vykresľujeme iba jednú aktuálnu stránku. Nároky na výkon a pamäť sú razantne menšie kedže stále držíme referenciu iba na jednú stránku. Pri zmene stránky sa stará zmaže a vytvorí sa ďalšia. Trvá to o trošku dlhšie ako keď máme načítané všetky ale rozdiel je minimálny.

Pre druhú možnosť je najlepšie využiť návrhový vzor Stav.

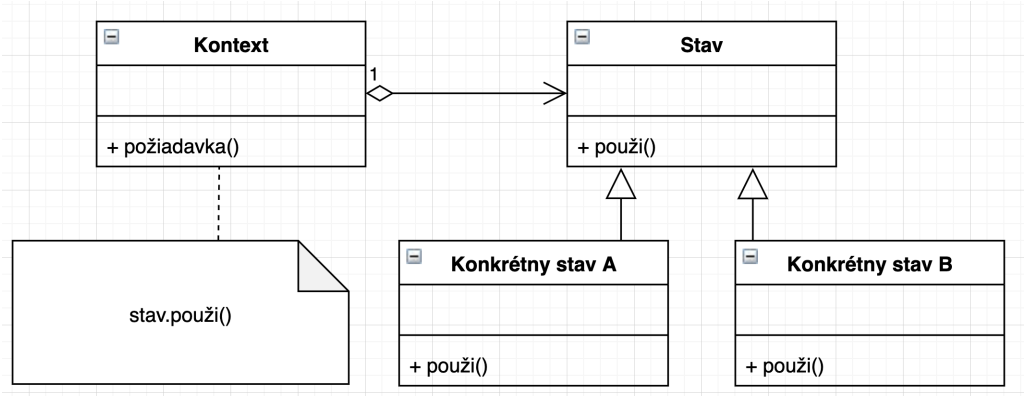
3.1 Návrhový vzor Stav

Pri stvorbe aplikácií alebo pri vytváraní menu v hrách sa najčastejšie používa návrhový vzor Stav. Návrhový vzor Stav povoľuje objektu zmeniť svoje správanie keď sa jeho vnútorný stav zmení. Zdá sa, že objekt zmení svoju triedu.

Výhodou je logika špecifická pre daný stav je lokalizovaná v triedach, ktoré reprezentujú tento stav[4].

Tento návrhový vzor sa teda hodí pre tvorbu aplikácie pre systémové nastavenia ako aj pre aplikáciu sprievodcu po prvom spustení. Okno aplikácie je objekt ktorého spravanie sa bude meniť. Každá stránka v aplikácií bude teda aktuálny

stav v objekte okno. Keď sa presunieme na dalšiu stránku v aplikáciií objekt okno jednoducho zmení svoj stav z aktuálnej stránky na nasledujúcu novú stránku.



Obr. 3.1: UML diagram návrhového vzoru stav[6]

Kontext

- definuje rozhranie záujmu pre klientov
- udržuje inštanciu podtriedy konkrétny stav, ktorá definuje aktuálny stav

Stav

Definuje rozhranie na zapuzdrenie pridruženého správania s konkrétnym stavom kontextu.

Konktétny stav

Každá podtrieda implementuje správanie spojené so stavom kontextu

3.2 Návrh používateľského rozhrania

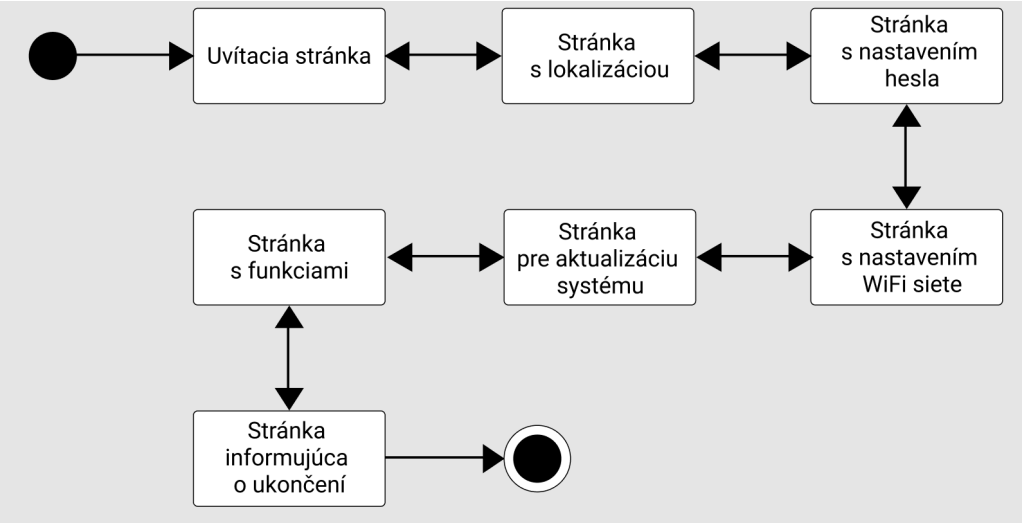
Používateľské rozhranie bolo vytvárané iteratívne, formou prototypovania vo viacerých iteráciách. V každej iterácií bolo vykonané overenie formou používateľského testovania. Prvé prototypy boli vytvorené v prototypovacom nástroji Figma[1] a následne otestované a vyhodnotené používateľským dotazníkom. Zmeny, ktoré užívatelia navrhli boli zrealizované v návrhu používateľského rozhrania.

[1]https://www.figma.com

3.2.1 Návrh postupnosti obrazoviek

Postupnosť obrazoviek sprievodcu po prvom spustení

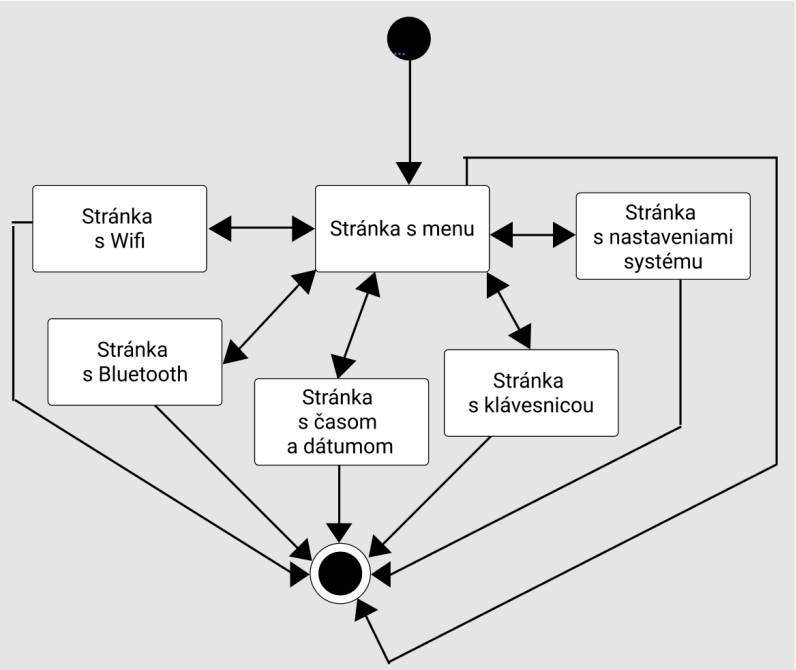
Ako vidíme z obrázka postupnosť obrazoviek je lineárna. Používateľ si musí prejsť každým krokom. Používateľ sa môže aj vrátiť späť avšak ukončenie programu je možné až na konci sprevodcu. Týmto chceme prinútiť používateľa prejsť sprievodcom po prvom spustení hneď po spustení operačného systému. Keďže riešenie bude modulárne postupnosť obrazoviek sa môže meniť v závislosti od použitia. Stále ale bude platiť že stránky budú nasledovať za sebou.



Obr. 3.2: Postupnosť obrazoviek sprievodcu po prvom spustení

Postupnosť obrazoviek nastavení systému

Postupnosť obrazoviek v nastaveniach systému je navrhnutá tak aby sa z každej stránky dalo dostať na každú stránku bez nutnosti sa vraciať spať. Nastavenia systému budú riešené podobne ako v operačnom systéme Fedora. Cez postrannú navigáciu sa používateľ môže dostať na akúkoľvek stránku nastavení systému. Pridaním nových stránok do aplikácie sa postupnosť obrazoviek nemení akurát do menu sa nám pridá viac stránok.



Obr. 3.3: Postupnosť obrazoviek nastavení systému

3.3 Grafická knižnica

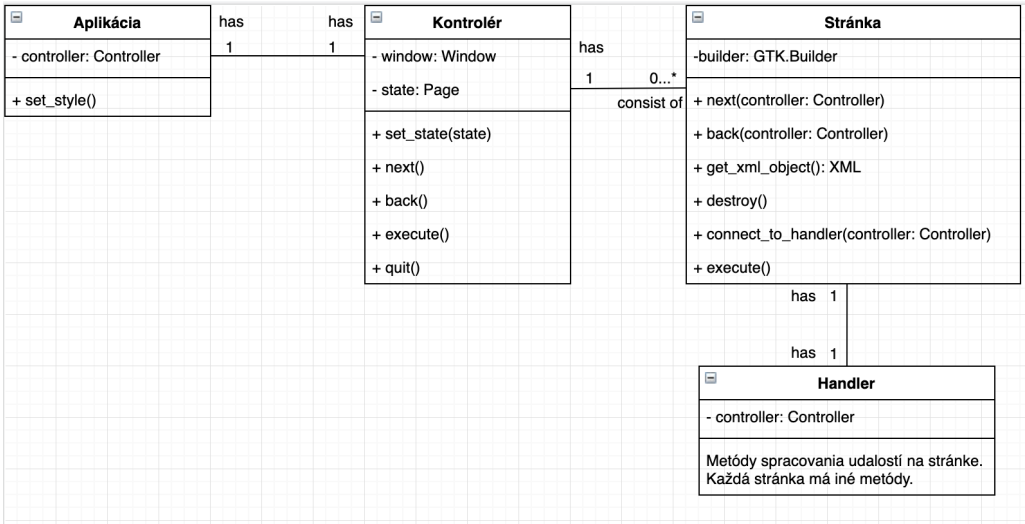
Pri výbere grafickej knižne máme na výber z viacerých možností ale najlepšou možnosťou bude použitie knižnice PyGTK. Pri knižnici PyGTK netreba do operačného systému Raspbian sťahovať ani inštalovať žiadne moduly navyše keďže ju používa aj samotný operačný systém. Snažíme sa využívať iba moduly čo obsahuje samotný operačný systém pretože sprievodca po prvom spustení ako aj nastavenia systému sa budú nachádzať už čistom operačnom systéme. S každým novým nainštalovaným modulom navyše sa zväčšuje aj veľkosť celého operačného systému a ak by sme pri každej novej aplikácií uvažovali týmto smerom, veľkosť celého operačného systému by sa rapídne zvýšila.

S knižnicou sa jednoducho pracuje a dokumentácia k nej je dostatočná.

4 Implementácia

4.1 Sprievodca po prvom spustení

4.1.1 Konceptuálny model



Obr. 4.1: Konceptuálny model

Aplikácia

Ako vidíme z konceptuálneho modelu, objekt Aplikácia má len tieto povinnosti: vytvoriť okno aplikácie a prenechať riadenie triede Kontrolér, ktorá všetko riadi. Pre nastavenie štýlu aplikácie slúži metóda:

- **set_style**() - táto metóda nastaví grafické vlastnosti aplikácie z css[1] súboru kde je zadefinované ako má každý element aplikácie vyzerať

[1] https://www.w3schools.com/whatis/whatis_css.asp

Kontrolér

Kontrolér sa stará o riadenie programu, to znamená: presúvanie sa z jednej stránky na druhú, vykonávanie danej stránky(nastavení, ktoré užívateľ zadal a aj jej vykreslenie, vypnutie aplikácie. Kontrolér je jeden a drží v sebe stále referenciu iba na jednu stránku, pri prepnutí na inú sa táto stránka zmaže. Kontrolér sa skladá z týchto metód:

- **set_state(state)** - metóda slúži na vytvorenie nového stavu. Parameter state je stránka na ktorú chceme prejsť.
- **next()** - spustí metódu next() na aktuálnej stránke
- **back()** - táto metóda spustí metódu back() na aktuálnej stránke
- **execute()** - tak isto ako metóda next() a metóda back() metóda execute() spustí metódu execute() na aktuálnej stránke
- **quit()** - vykonaním tejto metódy ukončíme aplikáciu

Stránka

Stránka je objekt, v ktorom je zadefinované čo sa má vykresliť pri vytváraní stránky, aká stránka je po nej a aká pred ňou, a čo sa má stať ak sa spustí vykonávanie stránky. Každá stránka musí obsahovať minimálne tieto metódy:

- **next(controller)** - vykonaním tejto metódy sa posunieme na dalšiu stránku. Metóda má jeden parameter, inštanciu kontroléra, ktorý všetko riadi. V metóde je definované aká stránka nasleduje po aktuálnej.
- **back(controller)** - vykonaním tejto metódy sa posunieme na predchádzajúcu stránku.Tak isto ako aj metóda next aj táto metóda má jeden parameter, inštanciu kontroléra, ktorý všetko riadi. V metóde je definované na aká stránka je pred aktuálnou.
- **get_xml_object()** - táto metóda vracia XML objekt danej triedy, v ktorom je zadefinované čo sa na danej stránke nachádza a ako má stránka vyzerať.
- **destroy()** - je metóda na zničenie stránky. Používa ju objekt kontrolér pri zmene stránky.

- **connect_to_handler(controller)** - je metóda pre pripojenie správcu udalostí k danej triede. Každá stránka má práve jedného správcu udalostí.
- **execute()** - v tejto metóde definujeme čo sa vykoná so stránkou. Či už je to pripojenie k WiFi sieti alebo nastavenie lokalizácie.

Týmto riešením vieme povedať z aktuálnej stránky aká sa nachádza pred ňou a aká následuje po nej. Je to navrhnuté modulárne takže ak chceme pridať novú stránku do sprievodcu po prvom spustení tak len jednoducho vytvoríme stránku a pozmeníme poradie stránok v **back()** a **next()** metódach.

Správca udalostí

Správca udalostí je objekt, ktorý sa stará o objekty na stránke a zabezpečuje čo sa vykoná ak daný objekt zmeníme. Príkladom môže byť napríklad formulár na zadanie hesla kde je treba zabezpečiť aby pri zadaní hesla sa ukazovateľ sily hesla zmenil. Každý správca má svoje metódy iné v závislosti na stránke ku ktorej patrí.

4.2 Nastavenia systému

Jednou z hlavných častí operačného systému sú aj systémové nastavenia. Sú vytvorené modulárne takže nieje problém s pridaním nových stránok alebo ich odobratím.

4.2.1 Konceptuálny model

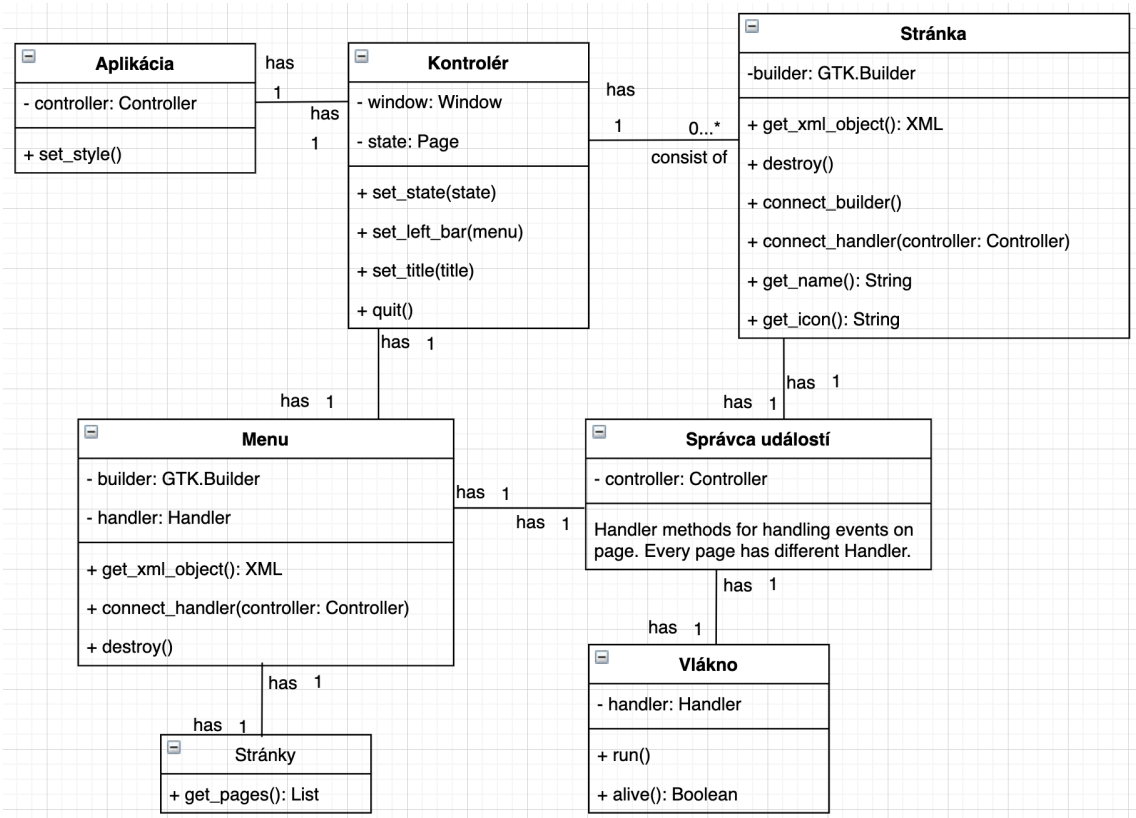
Aplikácia

Objekt aplikácia je podobná ako v sprievodcovi po prvom spustení. Vytvorí okno aplikácie, nastaví ho, nastaví ikonku aplikácie, pripojí css[2] súbor a prenechá kontrolu aplikacie objektu kontrolér.

Kontrolér

Kontrolér má rovnaké povinnosti ako aj v sprevodcovi po prvom spustení, to znamená že sa stará o ovládanie aplikácie. Kontrolér sa skladá z týchto metód:

[2]https://www.w3schools.com/whatis/whatis_css.asp



Obr. 4.2: Konceptuálny model

- **set_left_bar(menu)** - táto metóda slúži na nastavenie postranného ovladácieho panelu (menu)
- **set_state(state)** - metóda slúži na vytvorenie novej stránky. Parameter state je stránka. Zavolaním tejto metódy sa presunieme na požadovanú stránku.
- **set_title(title)** - táto metóda nastaví názov hlavičky aplikácie a názov v aktuálne otvorených aplikáciách na názov stránky na ktorej sa aktuálne nachádzame.
- **quit()** - vykonaním tejto metódy ukončíme aplikáciu

Menu

Tento objekt má na starosti nastavenie postranného ovládacieho panela (menu). To znamená zobrazenie v ňom list so všetkými dostupnými stránkami spolu s ich ikonkami. Pokiaľ daná stránka ikonku nemá zobrazí sa v menu čierne mínus na

indikáciu toho že daná stránka momentálne ikonku nemá. Názov stránky a aj jej ikonka je získavaná pomocou metód stránky **get_name**() a **get_icon**()

Stránky

V tomto objekte sú zadefinované všetky stránky, ktoré chceme zobraziť v aplikácii. Tento objekt má jedinú metódu **get_pages**(), ktorá vracia zoznam stránok pre zobrazenie. Zmenením poradia tohto zoznamu sa nám upraví poradie stránok v aplikácií. Pridaním stránky do tohto zoznamu sa pridá stránka aj do aplikácie.

Stránka

Stránka je objekt, kde je zadefinované čo sa nachádza na danej stránke. Každá nová vytvorená stránka musí byť potomkom tohto objektu. Tento objekt má tieto metódy, ktoré musia byť implementované v každej novej stránke:

- **connect__builder**() - je metóda pre vytvorenie GtkBuilder objektu, ktorý prečíta textové popisy používateľského rozhrania a vytvorí inštanciu opísaných objektov[3]
- **get_xml_object**() - táto metóda vracia XML objekt danej stránky, kde je zadefinované kde je aký objekt umiestnený a aké má vlastnosti.
- **destroy**() - je metóda na zničenie stránky. Používa ju objekt kontrolér pri zmene stránky.
- **connect_handler**(**controler**) - táto metóda zabezpečuje pripojenie správcu udalostí k aktuálnej stránke.
- **get_name**() - metóda vracia názov stránky, ktorý sa potom využíva v objekte **Menu** alebo v objekte **Kontrolér**
- **get_icon**() - je jediná metóda, ktorá nemusí byť implementovaná. Ak nieje imlementovaná automaticky vracia hodnotu **None**[4]. Metóda vracia cestu k ikonke stránky.

V projekte je vytvorená prázdna stránka aby vytváranie stránok bolo ešte jednoduchšie.

[3] https://developer.gnome.org/gtk3/stable/GtkBuilder.html
[4] https://www.w3schools.com/python/ref_keyword_none.asp

Správca udalostí

Správca udalostí má rovnakú úlohu ako v sprievodcovi po prvom spustení, taktiež sa stará o udalosti ktoré nastanu na stránke. Pre prípad ak by vykonávanie kódu trvalo dlhšie je vytvorená trieda **Vlákno**.

Vlákno

Tento objekt slúži na vykonávanie úloh na pozadí. Ak máme kód, ktorý chceme aby sa vykonával na pozadí vytvoríme inštanciu[5] tohto objektu a zadáme akú úlohu na pozadí má vykonať. Tento objekt má tieto metódy:

- **run**() - metóda spusti vykonávanie kódu na pozadí
- **alive**() - táto metóda slúži na overenie či sa nami vybraný kód stále vykonáva na pozadí.

4.3 Výzor aplikácií

Rozloženie aplikácie

Každá stránka v sprievodcovi po prvom spustení alebo aj v nastaveniach systému obsahuje XML súbor, v ktorom sú zadefinované objekty na stránkach ako aj ich rozloženie na stránke. Pre jednoduchšie navrhovanie a spravovanie stránok je vhodné použiť nástroj **Glade** [6]. Glade je nástroj, ktorý umožnuje navrhovať používateľské rozhrania bez nutnosti písať kód. Všetky vlastnosti sa uložia do .glade súboru a následne ich je možné načítať aplikáciou.

Vzhľad aplikácie

Výzor aplikácií a objektov na stránkach je zadefinovaný v .css súbore. Primárne GTK aplikácia berie výzor objektov z operačného systému. Ak však chceme výzor upraviť, v každej aplikácií sa nachádza .css súbor kde je možné možné zmeniť výzor objektov. Následne či už pomocou nástroja Glade alebo manuálne je možné túto zmenu aplikovať na objekt v našej aplikácií.

[5] https://docs.python.org/3/tutorial/classes.html
[6] https://glade.gnome.org/

5 Dosiahnuté výsledky

6 Interpretácia výsledkov

7 Ďalšia práca

8 Záver

Literatúra

1. FROMAGET, Patrick. The awesome story of Raspberry Pi. In: dostupné tiež z: https://raspberrytips.com/raspberry-pi-history/.
2. TECHOPEDIA. Command Line Interface (CLI). In: dostupné tiež z: https://www.techopedia.com/definition/3337/command-line-interface-cli.
3. TECHOPEDIA. Compiler. In: dostupné tiež z: https://www.techopedia.com/definition/3912/compiler.
4. TONI SELLARÈS, Universitat de Girona. The State Pattern.
5. CARBONNELLE, Pierre. PYPL PopularitY of Programming Language. 2020.
6. KHOSRAVI, Khashayar; GUÉHÉNEUC, Yann-Gaël. A Quality Model for Design Patterns. 2020.
7. HACKERNOON. Common Errors of UI Designers. In: 2019. Dostupné tiež z: https://hackernoon.com/6-bad-ui-design-examples-common-errors-of-ui-designers-e498e657b0c4.
8. MUTHIAN, Abishek. Getting smoother desktop experience on Raspberry Pi. In: 2019. Dostupné tiež z: https://abishekmuthian.com/getting-smoother-desktop-experience-on-raspberry-pi/.
9. Raspberry foundation. Your new desktop computer. In: 2019. Dostupné tiež z: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/.
10. TECHTARGET. User interface (UI). In: 2019. Dostupné tiež z: https://searchapparchitecture.techtarget.com/definition/user-interface-UI.
11. Katedra počítačov a informatiky. IoT lab oficiálne otvorený! In: 2018. Dostupné tiež z: https://kpi.fei.tuke.sk/sk/iot-lab-oficialne-otvoreny.

12. TECHTERMS. Interpreter. In: 2010. Dostupné tiež z: `https://techterms.com/definition/interpreter`.

13. WILEY, John; SONS. *The Essential Guide to User Interface Design*. WILEY, 2007.

14. KUCHANA, Partha. Software Architecture Design Patterns in Java. In: AUERBACH PUBLICATIONS, 2004, s. 481.

15. NELSON, T. H. *The Home Computer Revolution*. The Distributors, 1977.