FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Bezdrátově ovladatelné USB zařízení
Student:	Tomáš Kuchař
Vedoucí:	Ing. Radomír Polách
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2019/20

Pokyny pro vypracování

Nastudujte princip komunikace pomocí technologie Bluetooth v prostředí platforem Raspberry Pi a operačního systému Android. Nastudujte USB Gadget kernel moduly funkční na Raspberry Pi Zero. Analyzujte, navrhněte a implementujte platformu pro bezdrátové předávání dat mezi Raspberry Pi a operačním systémem Android. V úvahu berte bezpečnost a možnost propojení většího množství Raspberry Pi k jednomu zařízení s operačním systémem Android. Implementujte aplikaci pro ovládání USB Gadget kernel modulů na Raspberry Pi Zero pomocí mobilního telefonu s operačním systémem Android. Implementovanou aplikaci důkladně otestujte a zdokumentujte.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 7. ledna 2019

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Bezdrátově ovladatelné USB zařízení

Tomáš Kuchař

Katedra softwarového inženýrství
Vedoucí práce: Ing. Radomír Polách

13. května 2019

Poděkování

V první řadě bych chtěl poděkovat svému vedoucímu práce za podporu, rady, nápady a čas při konzultacích, za skvělé nápady při společném vymýšlení zadání pro tuto práci a také za vypůjčení jednoho zařízení pro lepší testování. Dále bych chtěl moc poděkovat své rodině za čas a prostředky, které mi dali nejen při psaní této práce, ale v průběhu celého studia, protože bez nich bych se sem nikdy nedostal. Děkuji.

Prohlášení

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

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. Dále prohlašuji, že jsem s Českým vysokým učením technickým v Praze uzavřel dohodu, na základě níž se ČVUT vzdalo práva na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona. Tato skutečnost nemá vliv na ust. § 47b zákona č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů.

V Praze dne 13. května 2019 ……

Odkaz na tuto práci

Kuchař, Tomáš. *Bezdrátově ovladatelné USB zařízení.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2019.

Abstrakt

Bakalářská práce si dává za cíl vytvořit chytré USB zařízení z Raspberry Pi, které bude po připojení k počítači simulovat různé USB periferie podle výběru uživatele. Zařízení bude bezdrátově ovladatelné pomocí aplikace na mobilní zařízení s operačním systémem Android. Obsahem práce je teoretický rozbor problému, analýza použitých technologií a popis implementace. Výstupem pak je software pro zařízení Raspberry Pi a mobilní aplikace pro Android.

Klíčová slova: Raspberry Pi Zero W, Android, Bluetooth Low Energy, USB kernel moduly, klávesnice, myš, flash disk.

Abstract

Main goal of this bachelor's thesis is to create a smart USB device from Raspberry Pi that will be able to simulate various USB peripherals by user selection. Device will be wirelessly controlled from Android mobile application. The content of this thesis is theoretical analysis of the problem, analysis of used technologies and description of implementation. The end result is a software for Raspberry Pi and Android mobile application.

Keywords: Raspberry Pi Zero W, Android, Bluetooth Low Energy, USB kernel modules, keyboard, mouse, flash drive.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V minulosti používaly počítače k připojení různých periférií různé konektory. S postupem času ale začal převládat konektor USB, který se stal univerzálním konektorem pro připojení většiny zařízení, která běžně k počítačům připojujeme. V dnešní době má nejeden USB konektor téměř každý stolní počítač i notebook. Pomocí USB konektoru se k počítačům připojují klávesnice, myši, joysticky, modemy, flash disky, externí pevné disky, čtečky paměťových karet, externí optické mechaniky a spousta dalších zařízení. Představme si, že bychom pomocí tohoto univerzálního konektoru připojili k počítači jedno zařízení, které by se podle potřeby dokázalo změnit na jakékoliv z těchto běžně připojovaných zařízení.

V této práci Vám představím své řešení tohoto problému pomocí zařízení Raspberry Pi Zero. Toto zařízení je dnes poměrně dost rozšířené díky poměru ceny a možností, které nabízí. Za pár stokorun si tak bude moci takovéto zařízení pořídit každý. Zároveň je zařízení vhodné i svými rozměry – výsledný produkt bude velký přibližně jako větší flash disk.

Po připojení zařízení k počítači se s ním uživatel spojí pomocí technologie Bluetooth svým mobilním telefonem. V mobilní aplikaci si pak jednoduše zvolí typ zařízení, které by chtěl používat, a následně bude zařízení z aplikace ovládat. V této práci se pokusím demonstrovat příklad použití na třech nejpoužívanějších USB zařízeních – klávesnici, myši a externím paměťovém úložišti.

Kapitola 1

Cíle práce

Cílem teoretické části práce je nastudovat princip komunikace pomocí technologie Bluetooth a jednotlivá prostředí platforem Raspberry Pi a operačního systému Android. Dále je potřeba nastudovat USB kernel moduly funkční na Raspberry Pi Zero. Po seznámení se s technologiemi bude následovat výběr správných technologií pro vytvoření bezdrátově ovladatelného zařízení z Raspberry Pi Zero přes technologii Bluetooth. Raspberry Pi Zero bude pomocí USB kernel modulů simulovat různé USB periferie pro hostitelský počítač. Toto zařízení bude bezdrátově ovladatelné pomocí mobilního telefonu s operačním systémem Android.

Cílem praktické části práce bude implementovat obě strany popsané v teoretické části. Zaprvé vytvořit program na Rapberry Pi Zero, který bude přes technologii Bluetooth přijímat požadavky a podle toho bude ovládat USB kernel moduly. Jako druhou část je třeba vytvořit uživatelsky přívětivou aplikaci pro operační systém Android, která bude tyto požadavky vysílat podle přání uživatele. Je nutné brát v úvahu bezpečnost a možnost propojení většího množství Raspberry Pi Zero k jednomu zařízení s operačním systémem Android. Po vytvoření aplikace bude následovat důkladné testování.

Kapitola 2

Analýza a popis technologií

2.1 Zařízení Raspberry Pi

Název Raspberry Pi je označení pro sérii jednodeskových počítačů vyvinutých charitativní společností Raspberry Pi Foundation. Původně byla zařízení vytvořena za účelem výuky práce s počítači, následně se však díky své ceně masově rozšířila i pro jiné účely. V dnešní době se Raspberry Pi používá nejčastěji jako alternativa k drahým počítačům v případě, že není potřeba vysoký výkon, jako ovládací jednotka v chytrých domácnostech nebo jako jednoduchý domácí server. [1]

Oproti klasickým počítačům má mnoho výhod. Hlavní výhoda je například ve spotřebě elektrické energie. V klasickém režimu bez připojení energeticky náročných periferií je spotřeba minipočítače okolo 3 wattů. Pro představu se podle [2] elektrická spotřeba jedné klasické žárovky vyrovná spotřebě přibližně 30 zařízení Raspberry Pi. Počítač je napájen pomocí micro USB konektoru, může být tedy napájen například z notebooku či powerbanky. Další dobrou vlastností těchto zařízení je jejich velikost. Celý jednodeskový počítač je velký jen pár centimetrů. [2]

Další výhodou Raspberry Pi je, že kromě potřebného hardwaru je vše zdarma. Pro tento malý počítač je k dispozici řada open source operačních systémů. Nejpoužívanější a nejpodporovanější z nich je Raspbian, který obsahuje i množství open source softwaru. [1] Díky širokému rozšíření těchto počítačů se vybudovala po světě i velká komunita lidí, kteří Raspberry Pi používají a sdílí své příklady použití s ostatními. Na internetu tak najdete velké množství podrobných návodů, knihoven a užitečných rad a zkušeností. [2]

Jak již bylo zmíněno, Raspberry Pi je jednodeskový počítač. Na této desce se nachází ARM procesor, podle verze buď 512 MB nebo 1 GB RAM, grafický procesor a řada dalších komponent včetně konektorů na připojení periferií. Vše záleží na konkrétní verzi Raspberry Pi. Operační systém běží na vložené MicroSD kartě. Tato karta je zároveň použita jako paměťové úložiště pro

soubory a programy. [2]

V tabulce 2.1 najdete srovnání vybraných dostupných modelů Raspberry Pi včetně parametrů a ceny udávané výrobcem.

Tabulka 2.1: Porovnání jednotlivých modelů Raspberry Pi [3]

Název modelu	**Procesor**	**RAM**	**Bezdrátová síť**	**Cena**
R. Pi Model B+	700 MHz	512 MB	není	$25
R. Pi 2 Model B	900 MHz	1 GB	není	$35
R. Pi 3 Model B	1200 MHz	1 GB	Wi-Fi, BT 4.1	$35
R. Pi 3 Model B+	1400 MHz	1 GB	Wi-Fi, BT 4.2	$35
Raspberry Pi Zero	1000 MHz	512 MB	není	$5
Raspberry Pi Zero W	1000 MHz	512 MB	Wi-Fi, BT 4.1	$10

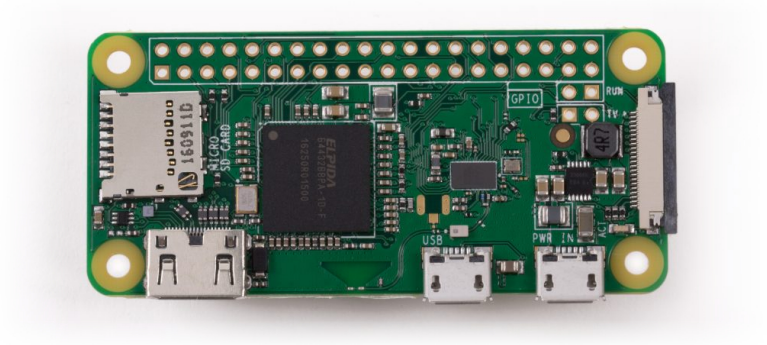
2.1.1 Raspberry Pi Zero W

V roce 2017 přišla společnost Raspberry Pi Foundation na trh s novou verzí svého počítače s názvem Raspberry Pi Zero W. Jednalo se o vylepšení původní verze Raspberry Pi Zero, která měla po svém vydání velký úspěch. V této nové verzi byla do zařízení oproti předchozí verzi přidána možnost připojení přes Wi-Fi a Bluetooth. Toto zařízení obsahuje:

- jednojádrový 1 GHZ procesor,
- 512 MB RAM,
- mini HDMI port,
- jeden micro USB datový port,
- jeden micro USB port určený k napájení zařízení,
- CSI konektor,
- slot na micro SD kartu,
- 802.11 b/g/n wireless LAN,
- Bluetooth 4.1,
- Bluetooth Low Energy. [4]

Po zvážení všech dostupných verzí Raspberry Pi padla jasná volba na verzi Raspberry Pi Zero W. Jako jedna z nejdůležitějších věcí, které nabízí, je možnost připojení přes Bluetooth, což je jeden z požadavků práce. Očekávaný

výkon pro simulaci základních USB zařízení také splňuje. K dispozici je jeden potřebný datový USB konektor a zároveň je zařízení pomocí USB i napájeno. Rozměry zařízení jsou 65 mm na délku a 30 mm na šířku, takže nebude problém pomocí redukce z micro USB na klasické USB připojit zařízení do USB konektoru počítače. Zároveň je zařízení levné a dostupné. Je tedy ideální volbou.



Obrázek 2.1: Zařízení Raspberry Pi Zero W. Převzato z [4]

2.1.2 USB kernel moduly

Abychom lépe pochopili, jak fungují USB kernel moduly, je nejprve potřeba podívat se na samotnou technologii USB. V USB existují 2 různé role pro koncová zařízení – master (host) zařízení a slave (peripheral) zařízení. Každé USB může mít v jednu chvíli pouze jednu z těchto rolí. Master zařízení je nadřazené a stará se o připojení slave zařízení. Úkolem slave zařízení je přidat nějakou funkcionalitu masterovi. K jednomu masterovi může přitom být na jedné sběrnici připojeno až 127 různých slave zařízení. Jako příklad si můžeme uvést počítač, který funguje jako master, a k němu připojujeme slave zařízení, jako např. klávesnici, externí hard disk, tiskárnu a další zařízení. [5]

Úkolem této práce je vytvořit USB slave zařízení. K tomu je potřeba, aby použité zařízení obsahovalo dvě věci – speciální hardwarovou část, která se jmenuje UDC (USB Device Controller) a potřebný software, který poběží na zařízení. USB slave zařízení většinou nelze vytvořit z klasického počítače, protože počítače obsahují USB rozbočovač, díky kterému máme v počítači více USB vstupů pro slave zařízení, ovšem omezuje to počítač pouze na roli master zařízení. Jde to ovšem vyřešit speciální rozšiřující kartou, ale není jich mnoho. Použití počítače jako USB slave zařízení totiž není moc běžná činnost. [5]

V našem případě však UDC je v Raspberry Pi Zero W zabudované a je připojené k jedinému datovému micro USB konektoru na desce. I to je jeden z hlavních důvodů, proč si vybrat právě tuto verzi Raspberry Pi. Další možné dostupné modely podporující UDC jsou Raspberry Pi Zero a Raspberry

Pi A+, ovšem ty zároveň nepodporují Bluetooth. Ostatní modely UDC nepodporují kvůli USB rozbočovači, který jim umožňuje mít více konektorů na připojení periferií. [3]

Samotné UDC však pro zprovoznění nestačí. Ještě je nutné nastavit USB slave zařízení softwarově. Linuxové jádro už v sobě obsahuje ovladače pro ovládání UDC. Je však potřeba nastavit funkcionality pro jednotlivé USB periferie (jako například paměťové úložiště, sériové zařízení, atd.). Těmto periferiím simulovaným linuxovým jádrem se říká „USB gadgets“. Standard USB umožňuje jednomu zařízení poskytovat více funkcionalit. Této množině funkcionalit se říká konfigurace. [5]

Implementace USB gadgets pomocí linuxového jádra obsahuje vrstvu s názvem „composite layer“, která obsahuje kód společný pro všechny funkcionality a umožňuje skládat gadgets z několika funkcionalit. Tato vrstva pak komunikuje s ovladačem UDC. Ten se pak postará o namapování funkcionalit (kterým se na této vrstvě říká USB funkce) na UDC. [5]

2.2 USB periferie

Ještě před 20 lety bylo připojování různých periferií jako klávesnice a myš k počítači celkem oříšek. Každé zařízení používalo jiný konektor, podle toho, který mu vyhovoval nejvíce. Tiskárny používaly buď paralelní nebo sériový port, externí paměťová úložiště používaly kvůli rychlosti paralelní port, modemy a digitální kamery sériový port. Přitom paralelní port byl v počítači většinou jen jeden a sériových také nebylo mnoho a byly pomalé. Pokud zařízení potřebovalo pracovat s počítačem rychle, obvykle se musela připevnit na základní desku rozšiřující karta. [6]

Všechny tyto problémy se pokusila vyřešit technologie USB. Pomocí této jednoduché standardizované technologie můžete stejným konektorem připojit k jednomu počítači až 127 různých periferií. [6]

USB bylo vytvořeno zatím ve třech verzích. První verze USB 1.0 se v tehdejší době moc nerozšířila. Druhá verze s názvem USB 2.0 oproti první verzi zvýšila přenosovou rychlost na 480 Mb/s a donedávna jsme se s ní mohli setkat na téměř všech zařízeních. Tuto verzi USB obsahuje i Raspberry Pi Zero W [3]. Další verze 3.0 zvýšila rychlost přenosu dat oproti předchozí verzi přibližně desetinásobně a přidala řadu dalších výhod. Tato nová verze je zpětně kompatibilní se starší verzí. [6]

USB verze 2.0, které obsahuje naše zařízení, má uvnitř čtyři piny. Dva z nich slouží k napájení zařízení. Při napětí 5 voltů dává maximální proud 500 mA. Já je využiji k napájení samotného Raspberry Pi, protože to je také napájeno pomocí USB a poskytovaný proud je pro napájení dostatečný. [3] Zbylé 2 vodiče z USB jsou tvořeny kroucenou dvojlinkou a využity pro přenos dat. [6]

Ve svém projektu budu pomocí Raspberry Pi simulovat tři nejpoužívanější USB periferie – klávesnici, myš a flash disk.

2.2.1 Klávesnice

Klávesnice je vstupní periferie počítače, pomocí které můžete např. psát text, používat klávesové zkratky nebo ovládat počítačové hry. Každá klávesnice se liší podle výrobce, operačního systému, pro který je navržena, jazykového rozložení nebo podle toho, k jakému zařízení je určena pro připojení. Běžná klávesnice obsahuje 80 až 110 kláves. [7]

Přestože má každá klávesnice trochu odlišné a jinak uspořádané klávesy, způsob, jakým komunikuje se zařízením, ke kterému je připojená, je stále stejný. Klávesnice vyhodnotí, která tlačítka byla kdy stisknuta a kódy těchto tlačítek odešle počítači (případně jinému zařízení, ke kterému je připojena). Ve standardu USB je popsáno 231 základních kláves, která může klávesnice obsahovat a jednotlivé kódy, které se při stisku daného tlačítka odešlou. Počítač pak jednotlivé kódy zpracuje a podle svého nastavení provede příslušnou akci. Pro představu – když v operačním systému změním jazykové rozložení klávesnice, změní se pouze nastavení v počítači. Klávesnice bude stále posílat ty samé kódy pro jednotlivá tlačítka, ale počítač už je bude jinak zpracovávat a vkládat správné znaky podle zvoleného rozložení. [7]

Klávesnice spadá do skupiny zařízení HID (Human Interface Device). Do této skupiny patří všechna zařízení, která mají za úkol umožnit člověku ovládat počítač nebo zobrazovat různé výstupy z počítače. Další taková zařízení jsou například myš nebo joystick. Pro tuto skupinu zařízení má standard USB určeno, jakým způsobem mají komunikovat s počítačem. Celý princip je popsaný v oficiálním dokumentu Device Class Definition for HID 1.11 [8]. V dokumentu HID Usage Tables 1.12 [9] pak nalezneme definované konstanty, které USB HID zařízení posílá počítači tak, aby počítač podle konstanty dokázal porozumět odesílanému požadavku. [8]

Blokům dat, které si navzájem posílají HID zařízení a počítač, se říká reporty. Než však může zařízení začít komunikovat s počítačem, je potřeba se domluvit, v jakém formátu se budou data posílat, neboli jakou strukturu budou mít reporty. Po připojení HID zařízení k počítači odešle toto zařízení tzv. report descriptor. Ten jasně definuje formát reportů, které bude posílat a které očekává. Potom už se mohou začít odesílat data. [8]

Standardní odchozí report klávesnice se skládá z 8 bajtů. První bajt obsahuje informaci o stisknutých modifikátorech – speciálních klávesách jako například CTRL, ALT atd. Seznam těchto kláves a příslušnou pozici bitu najdete v tabulce 2.2. Druhý bajt v reportu je zatím rezervovaný a nemá žádnou funkci. Dalších 6 bajtů postupně označuje klávesy, které byly stisknuty. [8]

Report, který naopak posílá počítač klávesnici, má obvykle velikost 1 bajt. Jednotlivé bity v tomto bajtu reprezentují LED diody na klávesnici pro jed-

Tabulka 2.2: Seznam modifikačních kláves s příslušnou pozicí v prvním bajtu reportu klávesnice [8]

Pozice bitu	**Klávesa**
0	Levý CTRL
1	Levý SHIFT
2	Levý ALT
3	Levý GUI
4	Pravý CTRL
5	Pravý SHIFT
6	Pravý ALT
7	Pravý GUI

notlivá tlačítka. Díky tomu pak poznáme, jestli máme zapnutý CapsLock, NumLock a další podobné klávesy. [8]

Mým úkolem tedy bude po připojení zařízení jako klávesnice odeslat počítači správný report descriptor a poté odesílat binárně 8 bajtů ve správném formátu s hodnotami kláves podle vstupu od uživatele.

2.2.2 Myš

Myš je vstupní periferie počítače, která má za úkol pohybovat s kurzorem po obrazovce, klikat různými tlačítky, nebo posunovat obsah stránek. Běžná počítačová myš obsahuje levé tlačítko pro výběr, pravé tlačítko pro kontextové menu (nebo další funkce v různých programech), kolečko pro vertikální scrollování a kliknutí s různými funkcemi. Některé myši navíc obsahují tlačítka pro krok zpět a vpřed. Všechny tyto funkce bude moje implementace myši umožňovat.

Stejně jako klávesnice se i myš řadí do skupiny HID zařízení. Platí pro ni tedy stejný postup jako pro klávesnici. Jako první akci po připojení k počítači musí myš odeslat příslušný report descriptor, pomocí kterého informuje master zařízení, že se jedná o myš a definuje délku a strukturu jednotlivých reportů, které budou obsahovat data vygenerovaná uživatelem při používání myši. [8]

Klasický myší report se skládá minimálně ze 3 bajtů. V prvním bajtu je pomocí jednotlivých bitů určeno, která tlačítka jsou stisknuta. Běžně jsou z tohoto bajtu využity minimálně 3 bity – první pro levé tlačítko, druhý pro pravé tlačítko a třetí pro klik kolečka. Pokud je příslušný bit nastaven na 1, pak je tlačítko stisknuto, jinak je nastaven na 0. [8] Popis tlačítek, která se odesílají v prvním bajtu v mé implementaci myši naleznete v tabulce 2.3.

Druhý bajt reportu slouží k odeslání hodnoty posunutí kurzoru myši v horizontálním směru, třetí bajt k posunutí ve vertikálním směru. Další bajty

Tabulka 2.3: Seznam tlačítek s příslušnou pozicí v prvním bajtu reportu myši v mojí implementaci [10]

Pozice bitu	**Tlačítko myši**
0	Levé tlačítko
1	Pravé tlačítko
2	Stisknutí kolečka
3	Tlačítko zpět
4	Tlačítko vpřed
5	Nevyužito (0)
6	Nevyužito (0)
7	Nevyužito (0)

reportu nejsou v USB specifikaci definovány a záleží tak na výrobci konkrétního zařízení, zda je využije a jak s nimi naloží. [8] Běžně se však čtvrtý bajt používá pro scrollování kolečkem ve vertikálním směru, případně pátý bajt pro scrollování v horizontálním směru, pokud to myš umožňuje [10]. Délka reportu není omezena, v případě potřeby více funkčností myši si může výrobce zvolit délku reportu podle potřeby [8]. Moje implementace myši bude používat uvedené první 4 bajty reportu.

2.2.3 Flash disk

Při práci s počítači obvykle chceme své výsledky někam uložit, abychom se na ně mohli podívat později, někomu tyto informace předat nebo si něco stáhnout z internetu. K ukládání datových souborů se používají úložiště, která si uložená data budou pamatovat dlouhou dobu i po vypnutí počítače či odpojení od něj. Většina počítačů v dnešní době používá pevný disk. Ovšem když chceme data uložit mimo pevný disk zabudovaný v počítači, používají se k tomu kromě optických médií a externích pevných disků i flash paměti. [11]

Flash paměť se oproti pevnému disku liší tím, že nemá žádné mechanické části a všechno je uvnitř řešeno elektronicky. Běžně se s touto pamětí setkáváme uvnitř flash disků, které se připojují k počítači pomocí USB konektoru, v paměťových kartách nebo v BIOSu počítače. [11]

V této práci nasimuluji flash disk, který se bude pomocí USB konektoru připojovat k počítači. Jediný úložný prostor, který použité zařízení Raspberry Pi obsahuje, je na připojené micro SD kartě. Na této kartě je nainstalovaný operační systém, ale zároveň slouží k ukládání souborů. Použiji proto tuto kartu jako flash paměť pro ukládání dat na virtuální flash disky.

Vytvoření nového virtuálního flash disku se skládá z následujících kroků:

1. Vytvoření nového prázdného binárního souboru na micro SD kartě. Velikost souboru bude nastavena podle požadované kapacity flash disku.

2. Naformátování vytvořeného binárního souboru nějakým souborovým systémem.

3. Namapování virtuálního disku na USB port pomocí UDC. [12]

Po tomto procesu se bude zařízení tvářit jako klasický flash disk. V praxi si uživatel bude moci vytvořit na zařízení několik různých flash disků různé velikosti (do vyčerpání kapacity vložené micro SD karty) a tyto flash disky potom jednotlivě aktivovat a libovolně mezi nimi přepínat.

2.3 Android

Android je operační systém určený převážně pro dotykové mobilní telefony a tablety. Jeho první verze se datuje do roku 2007, kdy se tato zařízení začínala objevovat na trhu. V tom samém roce i společnost Apple představila svou první verzi iPhonu. Do dnešní doby prošel Android několika změnami a vylepšeními s každou jeho novou verzí. V letošním roce by měla společnost Google, která stojí za systémem Android, představit jeho již desátou verzi. V poslední době se tento operační systém dostává i do spousty dalších zařízení, např. do televizí, automobilů nebo chytrých hodinek a náramků. V květnu roku 2017 společnost Google uvedla, že celosvětově existuje přes 2 miliardy zařízení používající Android. [13]

V loňském roce byl operační systém Android nainstalován na přibližně 85–86% smartphonů. Zbytek tvořily produkty od firmy Apple s operačním systémem iOS. Smartphony s jinými operačním systémy tvořily pouze zanedbatelnou část trhu (méně než 0,1%). [13] I díky této skutečnosti byl pro tuto práci vybrán tento operační systém.

Základ tohoto operačního systému tvoří linuxové jádro. Aplikace pro Android je možné psát v několika různých jazycích. Nejpoužívanější a oficiální jazyk pro tento systém je Java. Dále se postupně dostává do popředí jazyk Kotlin, který je navržený tak, aby běžel na Java Virtual Machine. [14] Psát aplikace pro Android v jiných jazycích není tolik rozšířené, proto si ve svém projektu z těchto dvou jazyků vyberu Javu.

Po vytvoření aplikace se zkompiluje kód včetně všech závislostí do souboru s příponou APK. Pomocí tohoto souboru je pak možné nainstalovat aplikaci do mobilního telefonu nebo tabletu. Ve výchozím nastavení je však zakázané instalovat aplikace z jiných zdrojů než z oficiálního android obchodu s názvem Google Play. Jsou 2 možnosti – buď nahrát aplikaci do tohoto obchodu a umožnit tak všem uživatelům Androidu její instalaci, nebo povolit instalaci APK souborů v nastavení systému na konkrétním zařízení.

2.3.1 Obecná struktura aplikace

Aplikace pro Android se skládá z několika komponent. Informace o těchto komponentách jsou brány převážně z [15]. Každá z komponent aplikace má nějakou funkci. V konfiguračním souboru aplikace s názvem „manifest“ pak najdeme seznam těchto komponent a informace o tom, jak jsou spolu propojeny. Dále se v manifestu deklaruje seznam všech povolení, která aplikace potřebuje dostat od systému pro svoji činnost.

Zjednodušeně popíši základní komponenty aplikace:

- **Aktivity**
  Každá aktivita v Android aplikaci reprezentuje jednu obrazovku. Na této obrazovce se zobrazuje obsah a obsahuje prvky pro interakci s uživatelem. Na začátku se aktivitě nastaví základní layout, který určuje, jak a kde budou prvky na obrazovce zobrazené. Každá aktivita má svůj životní cyklus – od vytvoření až po její smazání. Během tohoto životního cyklu aktivity jsou vyvolávány různé metody při různých situacích nebo při interakci uživatele. Ukázku zjednodušeného životního cyklu aktivity můžete vidět na obrázku 2.2.

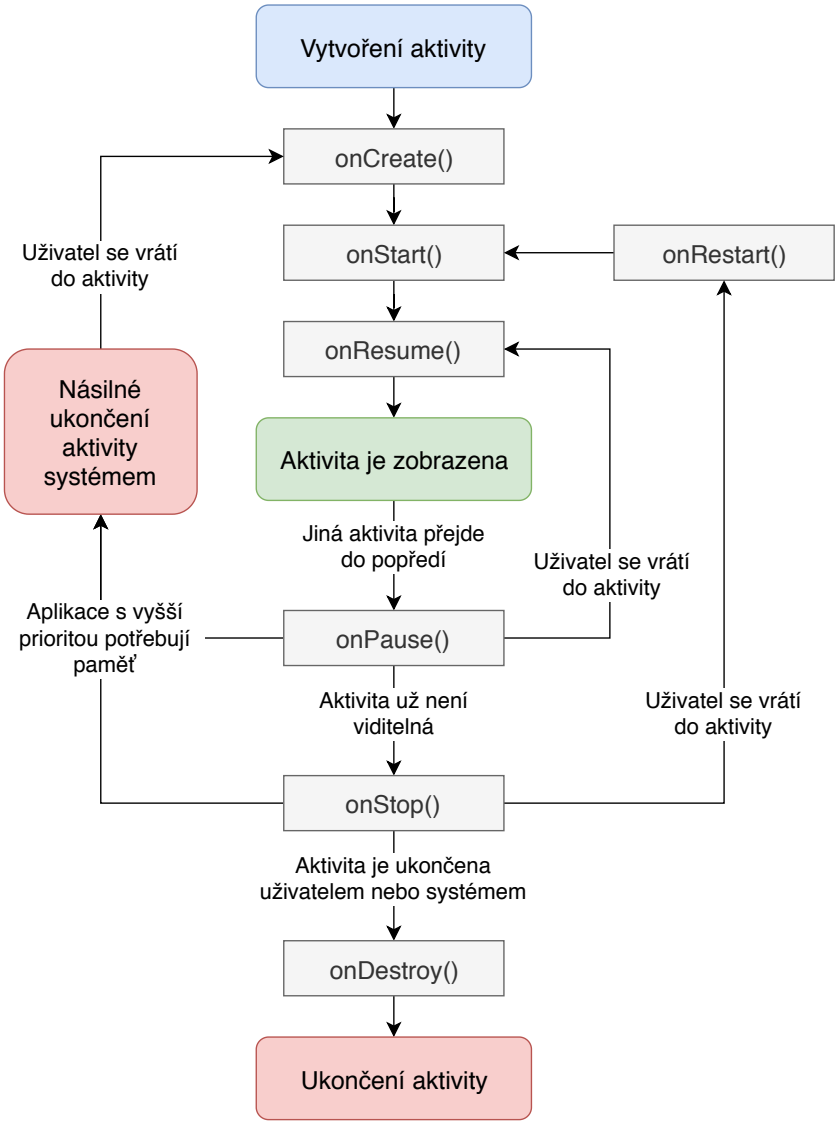
- **Fragmenty**
  Aktivita se může skládat z fragmentů. Fragment má stejně jako aktivita svůj životní cyklus, který není závislý na nadřazené aktivitě. Aktivita tak může být složena z částí, které fungují nezávisle na ní a tyto části dynamicky měnit.

- **Služby**
  Služby se používají pro déle trvající procesy, které běží na pozadí v systému. Pokud chceme, aby proces pokračoval nezávisle na přepínání různých obrazovek (aktivit) v naší aplikaci, použijeme právě služby. Jako příklad lze uvést přehrávání hudby pomocí hudebního přehrávače na pozadí i při přepínání obrazovek aplikace nebo přepnutí do jiné aplikace.

- **Broadcast Receivery**
  V systému Android si mezi sebou aplikace mohou posílat zprávy o nějakých událostech, které nastaly. Další komponentou aplikací jsou broadcast receivery, které mají za úkol čekat na zprávy, které mají zaregistrované, a v případě obdržení takové zprávy provést nastavené příkazy. Tyto zprávy mohou přijít od operačního systému, jiné aplikace běžící v systému nebo od jiné komponenty v rámci jedné aplikace. Například po připojení sluchátek do mobilního telefonu odešle systém tuto informaci všem broadcast receiverům, kteří na tuto zprávu čekají.

- **Content Providery**
  Content providery se starají o předávání dat. Zajišťují přístup k datům ze systému, z databází nebo jiných aplikací.

- **Intenty**
  Intenty jsou objekty, pomocí kterých si jednotlivé komponenty aplikace mezi sebou posílají zprávy. Většinou se jedná o nějaký požadavek na cílový objekt. Např. pokud chci vytvořit novou aktivitu, vytvořím intent a odešlu ho třídě dané aktivity. Nebo když chci zapnout Bluetooth, pošlu intent objektu *BluetoothAdapter* s žádostí o zapnutí Bluetooth. Na rozdíl od zpráv přijímaných broadcast receivery je tato zpráva odeslána jen konkrétnímu objektu, ne v rámci celého systému.



Obrázek 2.2: Životní cyklus aktivity v OS Android. Převzato z [16], přeloženo autorem práce.

2.4 Bezdrátová komunikace

Na jedné straně máme zařízení Raspberry Pi Zero W, na druhé straně zařízení s operačním systémem Android. Abychom mohli ze zařízení s Androidem ovládat Raspberry Pi, musíme nějak vyřešit komunikaci mezi nimi. Nejrozumnější bude vyřešit tuto komunikaci bezdrátově, je to pohodlné a při používání zařízení vždy budeme blízko něj, nemusíme tedy řešit vysoký dosah.

2.4.1 Přehled technologií

Zařízení Raspberry Pi Zero W nám nabízí dva způsoby bezdrátové komunikace – Wi-Fi a Bluetooth [4]. Oba tyto způsoby podporuje i drtivá většina dnešních mobilních zařízení a tabletů s operačním systémem Android. Bylo by tak možné použít obě technologie. Zhodnotíme si výhody a nevýhody těchto dvou technologií. Informace jsou čerpány z [17].

Hlavním rozdílem mezi Wi-Fi a Bluetooth je přenosová rychlost. Zatímco maximální přenosová rychlost klasického Bluetooth je 2,1 Mbps, maximální rychlost Wi-Fi je až 300krát vyšší. V tomto projektu si však zařízení nebudou posílat žádná objemná data a bohatě stačí přenosové rychlosti obou technologií.

Dalším rozdílným parametrem je dosah. U Bluetooth se uvádí vzdálenost 5–30 metrů. U Wi-Fi je to zhruba čtyřikrát více, záleží na podmínkách a vybavení. Těžko si ale představit, že by uživatel při ovládání zvolených periferií byl v jiné místnosti nebo byl tak vzdálený od počítače, že by neviděl podrobnosti na monitoru. Proto je i tady dosah Bluetooth dostačující.

Výhodou Bluetooth je, že má oproti Wi-Fi nižší energetickou spotřebu. Ostatní parametry, ve kterých se technologie liší, nejsou pro tento projekt nijak zásadně důležité. Z hlediska parametrů jsou tedy obě technologie dobře použitelné.

Hlavní nevýhodou použití Wi-Fi je, že pokud by se touto technologií propojil mobilní telefon s Raspberry Pi, nemohl by být zároveň připojený pomocí Wi-Fi k jiné síti a k internetu. Při použití Bluetooth tento problém nenastává, navíc může být přes Bluetooth připojeno více zařízení. Díky tomu bude nejlepší použít ke komunikaci technologii Bluetooth.

2.4.2 Technologie Bluetooth

Technologie Bluetooth byla vyvinuta v roce 1994. Využívá 2,4 GHz frekvenční pásmo stejně jako Wi-Fi. [17] Od jejího vzniku bylo vytvořeno několik verzí. Nejnovější verze má označení 5.0. Zařízení Raspberry Pi Zero W obsahuje verzi 4.1, je tedy nutné přizpůsobit implementaci právě této verzi. Tuto nebo novější verzi obsahuje i drtivá většina dnešních mobilních zařízení a tabletů s operačním systémem Android. Všechny novější verze jsou s touto verzí zpětně kompatibilní.

2.4.2.1 Bluetooth classic vs. Bluetooth LE

Od verze 4.0 přišla technologie Bluetooth s novým způsobem komunikace s názvem Bluetooth Low Energy (zkráceně Bluetooth LE nebo BLE, občas také pod názvem Bluetooth Smart). Hlavním rozdílem oproti původní technologii (Bluetooth classic) a účelem vzniku je znatelně nižší spotřeba energie. Spotřeba BLE je často zhruba 1–5% spotřeby klasického Bluetooth. Některá zařízení s BLE mohou být napájena malou knoflíkovou baterií (používá se do hodinek, kalkulaček, dálkových ovladačů atd.) a na tuto baterii mohou vydržet až pět let. [18]

Důvod nižší spotřeby energie u BLE je ten, že většinu času je zařízení v režimu spánku. Probouzí se pouze za účelem odeslání nebo příjmu dat a následně zase přejde do režimu spánku, kdy spotřebovává minimální množství energie. [18]

Zařízení s BLE má oproti Bluetooth classic trochu nižší dosah a nižší přenosovou rychlost. Tato nízkospotřebová technologie se používá například v chytrých hodinkách a náramcích, bezdrátových myších a klávesnicích, sportovních a zdravotních pomůckách nebo na ovládání chytré domácnosti. Naproti tomu klasické Bluetooth se používá díky své vyšší přenosové rychlosti tam, kde je potřeba přenášet větší objem dat. Např. u bezdrátových sluchátek a reproduktorů, k propojení mobilů s automobily, k přeposílání malých souborů atd.

V tabulce 2.4 můžete vidět porovnání několika nejdůležitějších parametrů obou technologií.

Tabulka 2.4: Porovnání parametrů Bluetooth classic s Bluetooth LE [19]

Parametr	**Bluetooth classic**	**Bluetooth LE**
Spotřeba energie	méně než 30 mA	méně než 15 mA (při plném výkonu bez režimu spánku)
Přenosová rychlost	0,7–2,1 Mbps	méně než 0,3 Mbps
Latence	100 ms	3 ms
Maximum připojených slave zařízení	7 zařízení	neomezeně

Rozdíl v dosahu technologií je minimální a oba pokryjí potřebnou vzdálenost pro vytvářené zařízení. Přenosová rychlost BLE je dostačující, protože si zařízení mezi sebou nebudou posílat objemná data. Na technologii BLE fungují bezdrátové klávesnice i myši, tudíž musí stačit i pro moji implementaci. Ostatní parametry nasvědčují tomu, že Bluetooth Low Energy bude správná volba pro tento projekt. Je to nová technologie, která se postupně dostává do spousty zařízení, proto ji využiji i v tomto projektu.

2.4.3 Bluetooth Low Energy

V této sekci popíši princip fungování Bluetooth Low Energy. Informace jsou čerpány z [20], pokud není uvedeno jinak.

Jakým způsobem se posílají data ve standardu BLE, stanovuje tzv. Generic Attribute Profile (zkráceně GATT). Tento profil popisuje kromě jiných věcí i běžně dostupné typy zařízení a způsob jejich komunikace. Zaručuje tak, že zařízení stejného typu od různých výrobců budou komunikovat stejným způsobem.

V technologii Bluetooth Low Energy jsou stejně jako v případě USB dvě role zařízení – master a slave. Master zařízení se v případě BLE nazývá „central", slave zařízení se nazývá „peripheral". Síť BLE má hvězdovou topologii. K jednomu central zařízení může být připojeno několik (teoreticky neomezeně) peripheral zařízení, ovšem každé peripheral zařízení je připojeno maximálně k jednomu central zařízení. Zařízení se stejnou rolí nelze spolu propojit.

Central zařízení je tzv. GATT client. Jeho úlohou je odesílat požadavky na server a zpracovávat odpovědi. Než začne GATT client odesílat požadavky, musí si zjistit od serveru, ke kterému se připojil, jaké možnosti komunikace poskytuje.

Peripheral zařízení má za úkol zprostředkovat tzv. GATT server. Tento server přijímá požadavky od klienta a na jejich základě něco provede nebo vrátí správnou odpověď. Pokud je k tomu server nastaven, může také odesílat upozornění připojenému klientovi o tom, že se změnila nějaká data, která si klient vyžádal sledovat. Poté, co klient obdrží zprávu o změně dat, musí odeslat na server požadavek pro jejich přečtení.

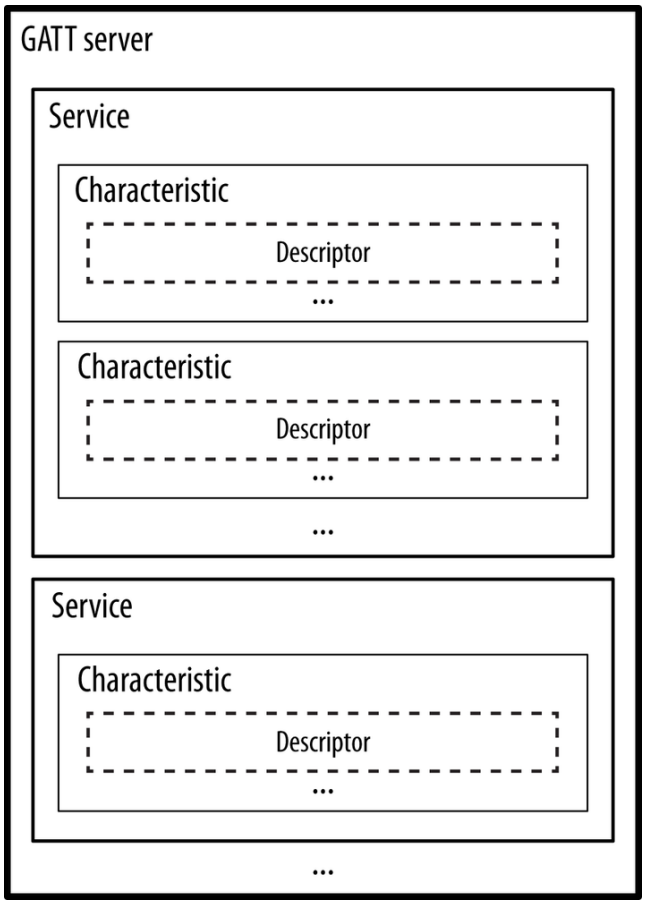
Základní datové entity, které jsou definované v GATT, se nazývají atributy. Jsou to adresovatelné části, které obsahují buď uživatelská data nebo informaci o struktuře a seskupení dalších atributů definovaných na serveru. Klient má přístup pouze k těmto atributům, takže všechny informace musí být organizovány touto formou.

Každý atribut je adresovatelný pomocí jednoznačného identifikátoru s názvem UUID (Universally Unique Identifier). Tento identifikátor musí být pro každý atribut v rámci jednoho GATT serveru unikátní. Je tvořen buď 128 nebo 16 (případně 32) bitovým číslem. Kratší identifikátory jsou definované v Bluetooth specifikaci pro různá zařízení, funkce těchto zařízení nebo výrobce. Měly by být použity jen ty, které jsou definované, a jen pro ta použití, pro která mají být použity. Delší z těchto dvou variant jsou pro uživatelské aplikace a pro zařízení, která nemají oficiálně rezervované své UUID. Z kratší varianty lze vytvořit i delší variantu dosazením (s případnými nulami na začátku) za písmena X v následujícím hexadecimálním vzoru:

`XXXXXXXX-0000-1000-8000-00805F9B34FB`.

Pokud navrhujeme atributům vlastní 128 bitové UUID, musíme si dát pozor, aby neodpovídaly vzoru pro převod kratší verze UUID.

Atributy v GATT serveru jsou rozděleny na služby, charakteristiky a deskriptory. Aby byla dodržena GATT kompatibilita, musí každý atribut definovaný v GATT serveru být právě v jedné z těchto tří skupin.



Obrázek 2.3: Hierarchie dat v GATT. Převzato z [20]

2.4.3.1 Služby

Nejvyšší typ atributu v GATT je služba (service). Služba může obsahovat žádnou nebo libovolný počet charakteristik. Úlohou služeb je seskupit související informace do jednoho smysluplného celku. Jako přirovnání by se dala použít složka souborů v operačním systému. Služba je typ atributu, má proto i svoje unikátní UUID.

2.4.3.2 Charakteristiky

Dalším atributem v GATT je charakteristika (characteristic). Tento typ atributu je možné pochopit jako kontejner pro uživatelská data. Pokud chce

uživatel pracovat s daty, vždy se to děje přes charakteristiku. Pro přirovnání si je můžeme představit jako datové soubory ve složce. Charakteristika může mít kromě uživatelských dat i žádný až libovolný počet deskriptorů.

Charakteristikám můžeme nastavovat vlastnosti podle toho, jak se s nimi má pracovat. Každá charakteristika může mít i více vlastností. Tyto vlastnosti si musí přečíst připojené central zařízení, aby vědělo, které operace může u charakteristik využívat. Základní vlastnosti charakteristik jsou například:

- **Read** – umožňuje uživateli číst data z charakteristiky,
- **Write** – umožňuje uživateli zapisovat data do charakteristiky,
- **Write without response** – stejné jako write, jediný rozdíl je, že se uživatel nedozví výsledek zápisu dat – server nemusí z nějakého důvodu přijmout požadavek a jediný způsob, jak se může uživatel dozvědět výsledek požadavku, je následné přečtení hodnoty v charakteristice,
- **Notify** – pokud je nastaveno, dovoluje serveru odesílat připojenému central zařízení upozornění na změnu dat,
- **Indicate** – stejný význam jako notify, navíc by měl uživatel ověřit, zda jsou vyžádaná data od serveru opravdu ta, která chtěl,
- **Broadcast** – pokud je nastaveno, dovoluje použít data z charakteristiky z tzv. advertising paketech (informace, které peripheral zařízení vysílá než se se k němu někdo připojí).

2.4.3.3 Deskriptory

Deskriptor (descriptor) je posledním typem GATT atributu. Každý deskriptor je přiřazen nějaké charakteristice. Jeho úloha je fungovat jako přepínač, který zapíná (případně vypíná) upozorňování připojeného uživatele na změnu dat v příslušné charakteristice. Toto však může fungovat pouze na charakteristikách s vlastnostmi notify nebo indicate.

2.4.3.4 Postup připojení k BLE zařízení

Pokud chce začít central zařízení komunikovat s peripheral zařízením, musí postupně provést následující kroky:

1. Skenovat zařízení v dosahu.
   V tomto kroku central zařízení objeví všechny peripheral zařízení v jeho dosahu, které se nabízí pro připojení. Peripheral zařízení musí mít nakonfigurovaný GATT server a veřejně o sobě vysílat informace, pomocí kterých zařízení poznáme a zjistíme, jaké služby nabízí. Tyto informace

se nazývají „advertising data“ a proces jejich vysílání „advertising“. Obsahují většinou název zařízení, MAC adresu zařízení, identifikátor výrobce, UUID nabízených služeb a další informace.

2. Připojit se k nalezenému zařízení.
   Central zařízení odešle požadavek na připojení. Zařízení si mezi sebou vymění potřebné informace a připojí se.

3. Přečíst si seznam služeb a charakteristik.
   Central zařízení si vyžádá od peripheral zařízení seznam všech jeho služeb, charakteristik a deskriptorů včetně všech jejich UUID, vlastností a dalších informací o nich. Bez těchto informací by central zařízení nevědělo, jakým způsobem s peripheral zařízením komunikovat.

4. Číst a zapisovat jednotlivé charakteristiky
   Central zařízení už je připojeno a má všechny potřebné informace o peripheral zařízení, může tedy nastat výměna dat.

2.5 Konkurenční řešení

Než se pustím do realizace tohoto projektu, je třeba se podívat, zda již neexistují nějaká podobná řešení tohoto problému. Po důkladném prozkoumávání dostupných produktů jsem nenašel žádné užitečné USB zařízení, které by se po připojení k počítači dokázalo průběžně přeměňovat na různé periferie.

Existují samozřejmě různé přepínače aktivních USB portů, do kterých je možné připojit více USB periferií, nebo dvojice bezdrátové klávesnice a myši, jejichž přijímač se nechá připojit pouze jedním USB konektorem. Tato zařízení však není možné rozšířit o další funkce.

Dále jsem našel jedno zařízení jménem Phantom Keystroker [21] od firmy ThinkGeek, které vypadá jako flash disk a po připojení k počítači si dělá legraci z uživatele. Můžete si manuálním tlačítkem na zařízení přepnout mezi náhodným mačkáním CapsLocku, náhodným vkládáním vtipných textů nebo náhodnými pohyby myši. Nelze ale toto zařízení nijak více ovládat nebo ho využít na něco smysluplného.

Dále existuje řada softwarových řešení, jak bezdrátově ovládat počítač pomocí mobilního telefonu. Všechna tato řešení vyžadují instalaci aplikace do mobilu a zároveň instalaci serveru na ovládaný počítač. K tomuto serveru se mobilní zařízení připojí pomocí Wi-Fi nebo Bluetooth a pomocí něj ovládá počítač. Příklad několika aplikací najdete v [22]. Takovéto řešení ale na rozdíl od mého hardwarového řešení nelze použít bez uživatelských práv na instalaci potřebného serveru, na serverem nepodporovaném operačním systému, na počítači nepodporujícím použité bezdrátové technologie nebo například při bootování před načtením operačního systému.

Podle mně dostupných informací tedy žádné podobné řešení tohoto problému zatím neexistuje.

Kapitola 3

Návrh a realizace

3.1 Raspberry Pi strana

Jako první věc je potřeba zvolit vhodný operační systém, který poběží na Raspberry Pi. Nejpodporovanější systém pro tato zařízení je Raspbian, proto ho použiji i v tomto projektu, konkrétně jeho verzi Raspbian Stretch Lite. Tato verze neobsahuje grafické uživatelské rozhraní a množství doporučeného softwaru a má nejmenší velikost ze všech verzí Raspbianu. [23] Je tudíž ideální pro tento projekt.

3.1.1 USB kernel moduly

Na funkčním Raspbianu nainstalovaném na micro SD kartě je nejprve potřeba nastavit systém, aby podporoval vytváření USB gadgets. Jak již bylo řečeno v sekci 2.1.2, linuxové jádro už v sobě obsahuje ovladače pro ovládání UDC. Je však potřeba tyto ovladače povolit a nastavit device tree overlay na „dwc2“, což nám umožní používat USB v režimu OTG (On The Go). Tento režim umožňuje přepínat USB mezi rolemi master a slave a je potřeba pro přepnutí USB na Raspberry Pi z výchozí role master zařízení do role slave. Tato nastavení se provedou následujícími příkazy:

```
echo "dtoverlay=dwc2" | sudo tee -a /boot/config.txt
echo "dwc2" | sudo tee -a /etc/modules
echo "libcomposite" | sudo tee -a /etc/modules
```

V tuto chvíli je systém nastaven a mohou se začít vytvářet gadgets. Pro každou USB gadget je potřeba vytvořit vlastní podadresář v adresáři

`/sys/kernel/config/usb_gadget/`.

V tomto podadresáři je potřeba specifikovat ID výrobce (`idVendor`) a ID produktu (`idProduct`). Dále je možno uvést verzi zařízení (`bcdDevice`) a verzi

USB portu (`bcdUSB`). Následně specifikuji důležité textové řetězce – sériové číslo, výrobce a název produktu. Dále je třeba vytvořit podsložky pro všechny konfigurace, specifikovat parametry konfigurací, vytvořit podsložky pro jednotlivé funkce a i jim specifikovat potřebné parametry. Po těchto všech krocích následuje propojení funkcí s konfiguracemi pomocí symbolických linků. [5]

V tuto chvíli je gadget připravena k použití, ale není namapovaná na UDC. Toto se provede napsáním názvu UDC kontroleru do atributu s názvem UDC v konfigurační složce dané gadget. Seznam dostupných UDC v systému je ve složce `/sys/class/udc`. Raspberry Pi Zero W obsahuje pouze jeden UDC, napíšu tedy právě jeho název. [5]

V následující ukázce je příklad mého skriptu konfigurace USB gadget jako klávesnice:

```
#!/bin/bash
cd /sys/kernel/config/usb_gadget/
mkdir smart_usb_keyboard
cd smart_usb_keyboard

echo 0x1d6b > idVendor  # Linux Foundation
echo 0x0104 > idProduct # Multifunction Composite Gadget
echo 0x0100 > bcdDevice # v1.0.0
echo 0x0200 > bcdUSB    # USB 2.0
mkdir -p strings/0x409  # US English
echo "baba1949deda1947" > strings/0x409/serialnumber
echo "Tomas Kuchar"     > strings/0x409/manufacturer
echo "Smart USB Device" > strings/0x409/product

mkdir -p configs/c.1/strings/0x409
echo "Config 1: Keyboard" > configs/c.1/strings/0x409/configuration
echo 250 > configs/c.1/MaxPower

mkdir -p functions/hid.usb0
echo 1 > functions/hid.usb0/protocol
echo 1 > functions/hid.usb0/subclass
echo 8 > functions/hid.usb0/report_length
echo -ne \\x05\\x01\\x09\\x06\\xa1\\x01\\x05\\x07\\x19\\xe0\\x29
\\xe7\\x15\\x00\\x25\\x01\\x75\\x01\\x95\\x08\\x81\\x02\\x95\\x01
\\x75\\x08\\x81\\x03\\x95\\x05\\x75\\x01\\x05\\x08\\x19\\x01\\x29
\\x05\\x91\\x02\\x95\\x01\\x75\\x03\\x91\\x03\\x95\\x06\\x75\\x08
\\x15\\x00\\x25\\xff\\x05\\x07\\x19\\x00\\x29\\xff\\x81\\x00\\xc0
> functions/hid.usb0/report_desc

ln -s functions/hid.usb0 configs/c.1/
ls /sys/class/udc > UDC
```

V této ukázce konfigurace klávesnice jsou zajímavé dva parametry funkce. Parametr `report_length` nám udává délku odchozího reportu klávesnice, jak bylo vysvětleno v sekci 2.2.1. Strukturu reportů zařízení nám určují data zadaná do parametru `report_desc`. Tento binární řetězec lze vytvořit pro konkrétní tvar požadovaného report deskriptoru pomocí informací z dokumentu [8]. V uvedeném dokumentu jsou také uvedeny příklady základních report deskriptorů pro běžná HID zařízení.

Pro ilustraci přikládám popis (jako v dokumentu [8]) mnou použitého report deskriptoru pro klávesnici po jednotlivých dvojicích bajtů, kde první bajt vždy značí typ informace a druhý její parametry:

```
0x05, 0x01,     // Usage Page (Generic Desktop)
0x09, 0x06,     // Usage (Keyboard)
0xA1, 0x01,     // Collection (Application)
0x05, 0x07,     //     Usage Page (Key Codes)
0x19, 0xe0,     //     Usage Minimum (224)
0x29, 0xe7,     //     Usage Maximum (231)
0x15, 0x00,     //     Logical Minimum (0)
0x25, 0x01,     //     Logical Maximum (1)
0x75, 0x01,     //     Report Size (1)
0x95, 0x08,     //     Report Count (8)
0x81, 0x02,     //     Input (Data, Variable, Absolute)
0x95, 0x01,     //     Report Count (1)
0x75, 0x08,     //     Report Size (8)
0x81, 0x03,     //     Input (Constant)
0x95, 0x05,     //     Report Count (5)
0x75, 0x01,     //     Report Size (1)
0x05, 0x08,     //     Usage Page (LED page)
0x19, 0x01,     //     Usage Minimum (1)
0x29, 0x05,     //     Usage Maximum (5)
0x91, 0x02,     //     Output (Data, Variable, Absolute)
0x95, 0x01,     //     Report Count (1)
0x75, 0x03,     //     Report Size (3)
0x91, 0x03,     //     Output (Data, Variable, Absolute)
0x95, 0x06,     //     Report Count (6)
0x75, 0x08,     //     Report Size (8)
0x15, 0x00,     //     Logical Minimum (0)
0x25, 0xff,     //     Logical Maximum (255)
0x05, 0x07,     //     Usage Page (Key codes)
0x19, 0x00,     //     Usage Minimum (0)
0x29, 0xff,     //     Usage Maximum (255)
0x81, 0x00,     //     Input (Data, Array)
0xC0            // End Collection (Application)
```

Obdobný postup vytváření USB gadget platí i pro ostatní zvolené periferie, konkrétní detaily lze vyčíst v přiloženém zdrojovém kódu.

3.1.2 Bluetooth Low Energy

Z hlediska Bluetooth LE propojení je zařízení Raspberry Pi v roli peripheral. Je tedy potřeba na něm zprovoznit GATT server a BLE advertising. Zařízení se bude nabízet pro připojení a po úspěšném připojení k centrálnímu mobilnímu zařízení bude Raspberry Pi přijímat od tohoto zařízení pokyny a podle nich přepínat konfigurace jednotlivých gadget a mapovat je na UDC, v případě klávesnice a myši navíc ještě odesílat do USB portu požadované reporty.

Přístup k Bluetooth protokolům zajišťuje v operačním systému tvz. Bluetooth stack, což je sada ovladačů a rozhraní. Nejrozšířenějším příkladem takového Bluetooth stacku je BlueZ, který podporují všechny známé linuxové distribuce. [24] BlueZ jsem použil i v tomto projektu.

Hlavní nevýhodou BlueZ je, že nemá žádnou kvalitní dokumentaci. Další nevýhodou mého projektu je, že v žádných dostupných zdrojích nejsou skoro žádné příklady použití BLE jako peripheral zařízení nebo dokonce nějaké knihovny zjednodušující takové použití, které by mi pomohly v mé implementaci. V praxi totiž není časté vytvářet vlastní BLE peripheral zařízení. Většina těchto zařízení je vytvořena specializovanými výrobci a vývojáři pak pouze píší aplikace na central zařízení, která se připojují k již vytvořeným peripheral zařízením a ovládají je nebo od nich sbírají data. Navíc technologie BLE zatím není natolik rozšířená jako klasické Bluetooth a donedávna BlueZ obsahoval řadu chyb v používání této technologie a navíc pro zpřístupnění funkcí na BLE se musel BlueZ přepnout do experimentálního módu. V aktuální verzi BlueZ 5.50 už je ale BLE oficiálně podporováno a už je i opravena většina chyb.

Jediným materiálem, ze kterého se nechá vycházet při vytváření GATT serveru a advertisingu, jsou ukázkové zdrojové kódy v jazyce Python, které ukazují základní příklad použití BlueZ na technologii BLE a jsou součástí instalace BlueZ. Dále BlueZ obsahuje pár velice stručných textových souborů s dokumentací, kde však není mnoho užitečných informací o použití. Jedinou informací, kterou jsem z této dokumentace použil, byl seznam podporovaných vlastností BLE charakteristik na BlueZ GATT serveru.

Implementace GATT serveru a advertisingu na Raspbianu nejdříve obnášela nastudování vzorových ukázek zdrojového kódu a jednotlivých použitých funkcí v nich. Ze zjištěných informací jsem následně vytvořil vlastní GATT server s vhodnou strukturou pro tento projekt a nastavil data vysílaná při advertisingu.

3.1.3 Výsledek

Pro naprogramování celé aplikace na straně Raspberry Pi jsem použil jazyk Python, což je jeden z primárně podporovaných programovacích jazyků pro tuto platformu. Zároveň jsou pro tento jazyk dostupné základní ukázky použití BlueZ. Aplikace využívá BlueZ pro zpřístupnění technologie Bluetooth Low Energy a k vytvoření GATT serveru a spuštění BLE advertisingu. Pro konfiguraci USB gadget na zařízení volá Python konfigurační skripty napsané pro Bash.

3.2 Bluetooth komunikace

Při používání Bluetooth Low Energy musí být dodržována nějaká pravidla komunikace. Je potřeba nastavit vhodnou strukturu GATT serveru pro tento projekt – vytvořit vhodné služby a pro každou službu vhodné charakteristiky a s těmito charakteristikami následně komunikovat ze strany mobilního zařízení.

Pro tento projekt jsem použil následující strukturu. Pro každou službu nebo charakteristiku jsem vygeneroval náhodné 128 bitové UUID podle pravidel BLE (více v sekci 2.4.3).

- Služba `device_service`
  UUID: `83ADCEEC-AF88-478A-A228-BBA5FD48EE2E`
  Tato služba seskupuje všechna data, která se týkají celého zařízení a jeho ovládání nezávisle na právě aktivní periferii.
  - Charakteristika `device_name_char`
    UUID: `6F14B945-16C6-4ED0-AADF-69DE9F5EB380`
    Vlastnosti: read, write
    Tato charakteristika má na starost název zařízení. Při přečtení vrátí aktuální název, při zápisu nastaví nový název zařízení.
  - Charakteristika `device_mode_char`
    UUID: `90106C9C-516A-4EEE-9A1A-D453FD0AD6D9`
    Vlastnosti: read, write
    Charakteristika pro čtení a nastavování módů zařízení (jaká periferie je zrovna aktivní nebo se má aktivovat).
  - Charakteristika `device_command_char`
    UUID: `73D54DE4-088A-4C8D-AFFB-4FB19E045E6A`
    Vlastnosti: write
    Skrz tuto charakteristiku je možné odesílat zařízení různé příkazy. Zatím je podporovaný pouze příkaz pro vypnutí zařízení.
- Služba `keyboard_service`
  UUID: `47EA1973-B412-4616-BC22-1B8811255CC8`
  Tato služba seskupuje všechna data pro periferii klávesnice.

  - Charakteristika `keyboard_write_char`
    UUID: `2856C451-221C-49A0-9BDD-639F6BCBDA98`
    Vlastnosti: write
    Do této charakteristiky se odesílají data ze simulované klávesnice (stisknuté klávesy).

- Služba `mouse_service`
  UUID: `CFC53B6A-3ACF-4728-BA69-991BFF539C13`
  Tato služba seskupuje všechna data pro periferii myš.

  - Charakteristika `mouse_write_char`
    UUID: `409ECD2C-DE0C-471A-A741-30757B5CAE0F`
    Vlastnosti: write
    Do této charakteristiky se odesílají data ze simulované myši (stisknutá tlačítka, pohyb myši).

- Služba `flash_service`
  UUID: `3A3BC59D-87C4-4593-8DF4-1A3FA4C4C2B8`
  Tato služba seskupuje všechna data pro periferii flash disk.

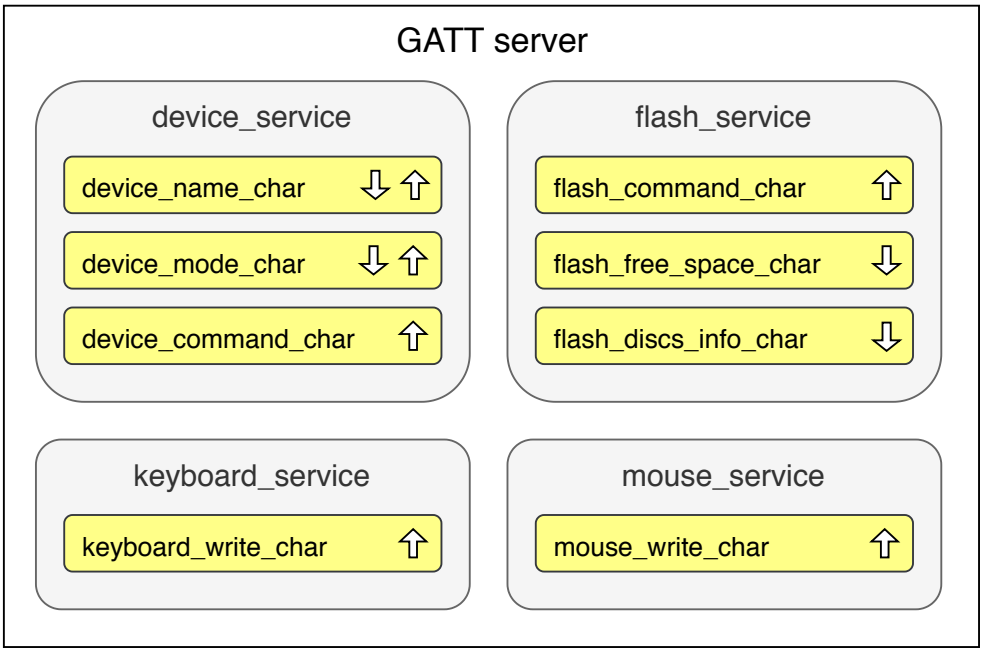
  - Charakteristika `flash_command_char`
    UUID: `B1D7F6E6-64BB-4F64-97E3-D2D6596923AD`
    Vlastnosti: write
    Tato charakteristika slouží k odesílání příkazů pro práci s virtuálními flash disky. Podporované příkazy jsou:

    ```
    activate <id>  // aktivuje disk s příslušným ID
    create <kapacita> <heslo> <název>  // vytvoří nový disk
    edit <id> <heslo> <název>  // upraví parametry disku
    delete <id>  // smaže zvolený disk
    ```

  - Charakteristika `flash_free_space_char`
    UUID: `50ADC1EA-9FE7-456F-8E67-061D3FC83F17`
    Vlastnosti: read
    Charakteristika vrací volnou kapacitu na použité SD kartě v zařízení Raspberry Pi. Volná kapacita je potřeba k omezení maximální velikosti nově vytvářeného flash disku.

  - Charakteristika `flash_discs_info_char`
    UUID: `9346AC56-3CE2-4B0B-8C67-AC67563CBC96`
    Vlastnosti: read
    Tato charakteristika vrací seznam všech vytvořených flash disků na zařízení.

Zvolenou strukturu GATT serveru pro tento projekt včetně jednotlivých služeb a charakteristik s jejich vlastnostmi ilustruje obrázek 3.1.



Obrázek 3.1: Použitá struktura GATT serveru pro tento projekt

3.3 Android strana

Hlavním cílem na straně operačního systému Android je vytvořit intuitivní aplikaci, pomocí které zvládne jakýkoliv uživatel jednoduše ovládat zařízení připojené k počítači. Tato aplikace bude podporovat všechny činnosti, které nabízí připojené zařízení.

3.3.1 Případy užití

Pro začátek by bylo vhodné uvést možnosti, které uživatel očekává od této aplikace a které aplikace bude podporovat. Jednotlivé případy užití popisují hlavní aktivity, které lze v systému vykonat uživatelem. Základní případy užití této aplikace jsou:

1. **Zobrazení seznamu aktuálně dostupných zařízení**
   Po zapnutí aplikace se spustí vyhledávání dostupných podporovaných zařízení v okolí. Nalezená zařízení se zobrazí v přehledném seznamu.

2. **Aktualizace seznamu dostupných zařízení**
   Uživatel může spustit opětovné vyhledávání dostupných podporovaných zařízení v okolí.

3. **Připojení k nalezenému zařízení**
   Uživatel se připojí k nalezenému dostupnému zařízení pomocí Bluetooth Low Energy.

4. **Zvolení aktivní periferie**
   Uživatel má na výběr ze tří podporovaných periferií. Z nich si jednu vybere a přesměruje se na obrazovku s ovládáním pro konkrétní periferii. Připojené zařízení se změní ve vybranou periferii.

5. **Přejmenování zařízení**
   Uživatel může přejmenovat připojené zařízení. Jméno zařízení se zobrazuje v seznamu dostupných zařízení.

6. **Vypnutí zařízení**
   Zařízení je možné vypnout, aby mohlo být bezpečně vysunuto z USB portu počítače, ke kterému je připojeno. Tato akce vrátí uživatele zpět na obrazovku se seznamem dostupných zařízení.

7. **Odpojení zařízení**
   Uživatel se může odpojit od aktivního připojeného zařízení. Tato akce ho vrátí zpět na obrazovku se seznamem dostupných zařízení.

8. **Ovládání klávesnice**
   Po zvolení klávesnice jako aktivní periferie může uživatel mačkat klávesy a odesílat data vybranému připojenému zařízení.

9. **Přepnutí rozložení klávesnice**
   Uživatel může přepnout jazykové rozložení klávesnice. Po zvolení jazyka se zobrazí dostupné klávesy pro toto rozložení. Jelikož USB zařízení nemá možnost zjistit softwarově nastavené rozložení klávesnice na připojeném počítači, musí si uživatel sám ohlídat, aby se zvolené rozložení v aplikaci shodovalo s rozložením na připojeném počítači. Bez této shody některé klávesy odesílají jiné znaky.

10. **Ovládání myši**
    Po zvolení myši jako aktivní periferie může uživatel tuto myš ovládat – pohybovat kurzorem a mačkat dostupná tlačítka.

11. **Zobrazení seznamu flash disků**
    Uživatel má možnost si zobrazit seznam všech vytvořených virtuálních flash disků na připojeném zařízení.

12. **Aktivování vybraného flash disku**
    Ze seznamu vytvořených flash disků si může uživatel vybrat disk, který chce aktivovat a připojit do USB portu. Následně se tento disk objeví v počítači a uživatel na něj z počítače může zapisovat nebo z něj číst data.

13. **Vytvoření nového flash disku**
    Pokud je na zařízení dostatek volného místa, má uživatel možnost vytvořit nový flash disk. Novému disku nastaví zvolenou kapacitu, název, případně může disk chránit heslem. Pro další práci s ním bude muset uživatel zadat zvolené heslo, pokud bylo nastaveno.

14. **Úprava existujícího flash disku**
    Uživatel může upravit název a heslo existujícího flash disku. Kapacita již vytvořeného disku upravovat nelze z důvodu ztráty dat.

15. **Odstranění flash disku**
    Uživatel může odstranit nepotřebný flash disk. Přijde tak o data na něm, ale uvolní v zařízení místo, které odstraněný virtuální flash disk zabíral.

3.3.2 Struktura aplikace

Jednotlivé obrazovky aplikace jsou v Androidu tvořeny aktivitami (viz. sekce 2.3.1). Každá aktivita se stará o zobrazování výstupu a získávání vstupu od uživatele. Moje aplikace je tvořena těmito aktivitami:

- `MainActivity` – Hlavní aktivita, zobrazí se při spuštění aplikace. Zobrazuje seznam dostupných zařízení v okolí. Tato aktivita pokrývá první tři případy užití.

- `DeviceDetailActivity` – Tato aktivita se zobrazí po rozkliknutí detailu připojeného zařízení. Nabízí uživateli jednotlivé možnosti, které může se zařízením provést. Tato aktivita pokrývá případy užití číslo 4 až 7.

- `KeyboardActivity` – Aktivita se zobrazením klávesnice v nastaveném rozložení. Jednotlivá rozložení se v aktivitě přepínají pomocí fragmentů. Aktivita se zobrazí po výběru klávesnice jako aktivní periferie v aktivitě `DeviceDetailActivity`. Pokrývá případy užití 8 a 9.

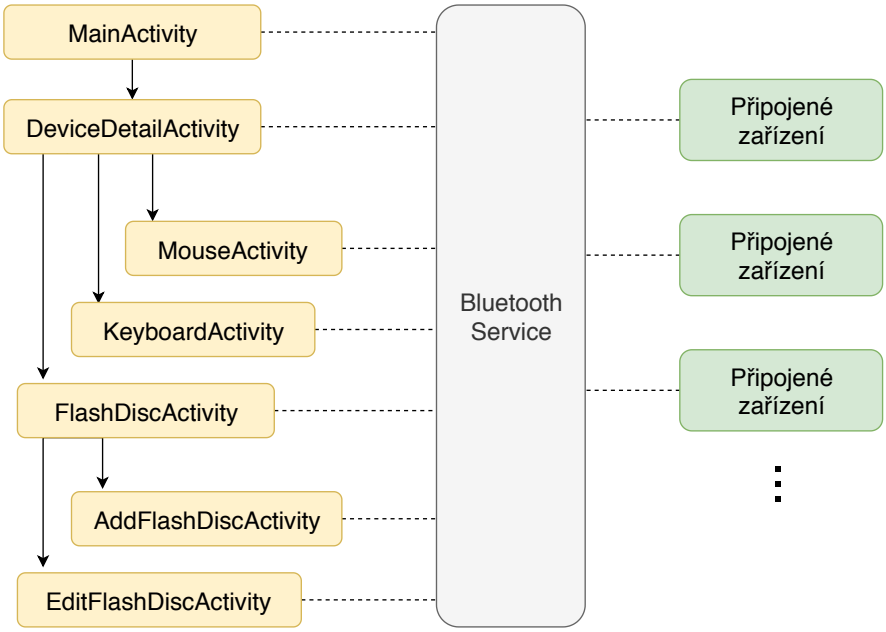
- `MouseActivity` – Aktivita se zobrazením tlačítek myši a touchpadem pro pohyb kurzoru. Zobrazí se po výběru myši jako aktivní periferie v aktivitě `DeviceDetailActivity`. Tato aktivita pokrývá případ užití číslo 10.

- `FlashDiscActivity` – Tato aktivita zobrazuje seznam existujících flash disků na připojeném zařízení. Zobrazí se při výběru flash disku jako aktivní periferie v aktivitě `DeviceDetailActivity`. Nabízí možnost práce s jednotlivými disky nebo přechod na aktivitu k vytvoření nového disku. Pokrývá případy užití 11, 12 a 15.

- `AddFlashDiscActivity` – Aktivita pro vytvoření nového flash disku. Lze na ni přejít z `FlashDiscActivity` a pokrývá 13. případ užití.

- `EditFlashDiscActivity` – Aktivita pro úpravu existujícího flash disku. Lze na ni přejít z `FlashDiscActivity`. Pokrývá případ užití číslo 14.

Po celou dobu spuštění aplikace běží na pozadí služba `BluetoothService`, která se stará o správu připojených zařízení a práci s nimi. Tato služba se vytvoří při spuštění aplikace a ukončí se při jejím vypnutí. Služba udržuje spojení se všemi připojenými zařízeními bez ohledu na právě zobrazenou aktivitu. Pomocí ní je možné připojit nebo odpojit zařízení. Služba obsahuje kolekci objektů jednotlivých zařízení, která jsou připojena. Pokud chce nějaká aktivita posílat nebo přijímat data z právě aktivního zařízení, pošle požadavek službě, která vybere správné zařízení a s tímto zařízením vyřeší požadavek. Po úspěšném ukončení požadavku je aktivita zpětně upozorněna příslušnou broadcast zprávou s výsledkem požadavku.



Obrázek 3.2: Zjednodušená struktura Android aplikace

Objekty jednotlivých zařízení mají kromě jiných i metody pro čtení a zápis do jednotlivých BLE charakteristik podle zvolené struktury GATT serveru. Nevýhodou Androidu je, že v jednom BLE spojení se zařízením nedokáže paralelně odeslat více požadavků (na čtení i zápis) tomuto zařízení (i různým charakteristikám). Pokud ještě nebyl dokončen předchozí požadavek, Android nevyhodí chybu, pouze další požadavek neprovede. Z tohoto důvodu jsem každému objektu zařízení (pro každé BLE spojení) vytvořil frontu na požadavky.

Aktivita, která chce odeslat požadavek na zařízení, tento požadavek předá přes `BluetoothService` aktivnímu zařízení a pokud toto zařízení zrovna provádí nějaký jiný požadavek, uloží si nový do fronty a zpracuje ho, až bude

mít předchozí požadavky splněny. Dotazující aktivita však o této frontě neví, pouze předá požadavek a čeká na broadcast zprávu s výsledkem, která je tímto systémem zajištěna.

Obrázek 3.2 ilustruje zjednodušenou strukturu aplikace a hierarchii zmíněných aktivit.

3.3.3 Bluetooth v Androidu

S každou novou verzí Androidu vychází i nová verze jeho SDK. Verze SDK se označují celým číslem, které se nazývá Android API level. V každé nové verzi jsou přidány nové funkce pro ovládání Androidu nebo vylepšeny, změněny nebo nahrazeny staré funkce. Všechna starší API jsou kompatibilní se všemi novějšími.

Podpora pro Bluetooth Low Energy byla poprvé přidána do Android API 18, což odpovídá verzi systému 4.3. V této verzi byly dostupné pouze funkce pro centrální roli BLE zařízení, což já potřebuji. [25] Funkce v tomto API jsou však omezené a ne dost dobře vyladěné. Například při připojování k duálnímu Bluetooth zařízení (zařízení, které podporuje Bluetooth classic a zároveň BLE) se pomocí funkcí v tomto API občas nepodaří navázat spojení, atd.

Problém s tímto připojováním byl vyřešen až v API 23 (verze Androidu 6.0), kde v této funkci přibyl volitelný parametr, který určuje typ Bluetooth, kterým se má Android k zařízení připojit [26]. Zároveň do této verze už je jiným způsobem vyřešeno vyhledávání zařízení v okolí. Pro lepší funkčnost aplikace jsem se proto rozhodl použít toto novější API 23.

Dalším oříškem při používání BLE v Androidu je nastavení správných oprávnění aplikace. Pro práci s potřebnými Bluetooth funkcemi je potřeba nastavit v manifestu aplikace oprávnění: [25]

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission
     android:name="android.permission.BLUETOOTH_ADMIN" />
```

Pokud ale chceme použít Bluetooth Low Energy, je potřeba kromě toho povolit také oprávnění pro určování polohy zařízení, což už se není možné dočíst v dokumentaci Androidu. Bez tohoto oprávnění aplikace nic nezahlásí, pouze při skenování dostupných zařízení žádné nenajde. Toto oprávnění je potřeba povolit, protože existují zařízení, která pomocí Bluetooth LE vysílají data o své poloze a při skenování zařízení by mohla aplikace podle těchto informací ze zařízení v dosahu zjistit přibližnou polohu mobilního telefonu. Pro povolení polohy je potřeba přidat do manifestu aspoň jedno z následujících povolení:

```
<uses-permission
    android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

```
<uses-permission
    android:name="android.permission.ACCESS_FINE_LOCATION" />
```

Od verze Androidu 6.0 je navíc rozdíl oproti předchozím verzím při udělování oprávnění aplikaci. Dříve se přidělila potřebná oprávnění při instalaci aplikace. Od této verze však aplikace nedostává oprávnění při instalaci a musí o ně přímo požádat při běhu, než chce použít funkcionality s potřebnými oprávněními. Oprávnění je více druhů. Pro použití Bluetooth funkcí není potřeba přímo žádat uživatele, toto oprávnění spadá do kategorie normálních. Naopak při žádosti o přístup k poloze je potřeba vyvolat systémový dialog, který požádá uživatele o povolení. Toto oprávnění spadá do skupiny nebezpečných. [27]

3.3.4 Výsledek

Byla naprogramována intuitivní aplikace pro operační systém Android, která splňuje všechny potřebné případy užití. Minimální verze systému, na které bude aplikace fungovat, je verze Android 6.0. Tato verze byla zvolena kvůli nově dostupným funkcím pro práci s Bluetooth Low Energy. V současné době (duben 2019) má tuto nebo novější verzi nainstalováno více než tři čtvrtiny zařízení s tímto operačním systémem [28].

Grafický náhled vytvořené Android aplikace je dostupný v příloze C.

Kapitola 4

Bezpečnost a testování

4.1 Zabezpečení v tomto projektu

V této sekci popíši, při čem je potřeba v tomto projektu dbát na bezpečnost a jak to vyřešit.

Na straně Androidu nepotřebuje vytvořená aplikace žádné speciální zabezpečení. Tato aplikace sama o sobě neuchovává žádná data, pouze se vždy připojí k nějakému zařízení a po odpojení zase všechny informace o dříve připojeném zařízení zapomene. Bez dostupných podporovaných zařízení v dosahu nemá aplikace žádné funkce.

Důležité je zabezpečit komunikaci mezi Androidem a Raspberry Pi. Mezi těmito zařízeními se posílají data o ovládání připojeného počítače, je potřeba tedy šifrovat tuto komunikaci. Pokud by nebyla komunikace šifrována, dalo by se odposlouchávat, jaké klávesy uživatel zmáčkl, pohyby myši nebo nastavená hesla pro jednotlivé flash disky. Bluetooth LE podporuje několik úrovní zabezpečení, více o šifrování komunikace rozeberu v následující sekci.

Na straně Raspberry Pi jsem zatím žádné zabezpečení neprováděl. Pro periferie klávesnici a myš není potřeba žádné zabezpečení, protože se v zařízení žádná data neukládají. Pouze data přijmu přes Bluetooth a hned je ve správném formátu odešlu do USB portu připojenému počítači. V případě flash disku se na SD kartě vložené do Raspberry Pi vytvoří binární soubor a namapuje se jako disk na USB port. Data se pak ukládají do tohoto binárního souboru a neodesílají se Android zařízení. Jedna možnost, jak se k datům dostat, je připojit opět zařízení k počítači a aktivovat příslušný binární soubor. Druhá možnost je připojit se do Raspbianu, namountovat tento binární soubor a pracovat s daty v Raspbianu. V každém případě ale uživatel musí mít fyzický přístup k zařízení.

Při používání klasického flash disku se data na něm také běžně nešifrují – pokud má někdo fyzický přístup k disku, má přístup i k datům. Nechal jsem to tak zatím i ve svém projektu. Další věcí je, že uživatel při používání

mého řešení má plný přístup k zařízení, SD kartě, operačnímu systému na ní i zdrojovým kódům pro tento projekt. Nemělo by tedy moc smysl se starat o nějaké složité zabezpečení.

Při vytvoření flash disku je možné nastavit tomuto disku heslo. Opět je to ale zatím spíše symbolické, jelikož to při neznalosti pouze brání mobilní aplikaci odeslat příkazy pro práci s disky. Nic ale uživateli nezabrání ve fyzickém přístupu ke kartě a k datům na ní. Do budoucna bych však rád naimplementoval pomocí těchto hesel přímo možnost šifrování binárního souboru s daty na SD kartě, aby při neznalosti hesla neměl uživatel možnost přijít k datům i při fyzickém přístupu ke kartě.

4.2 Bezpečnost Bluetooth LE

V této sekci velice zjednodušeně popíši druhy zabezpečení Bluetooth LE podle článku [29]. Pro podrobnější informace doporučuji přečíst uvedený článek.

V technologii Bluetooth Low Energy jsou definovány 2 módy zabezpečení. Každý mód přitom obsahuje ještě různé úrovně zabezpečení. První mód zajišťuje bezpečnost pomocí šifrování a má následující úrovně:

- **Úroveň 1** – žádné zabezpečení,
- **Úroveň 2** – neautentizované párování + šifrování pomocí AES–CCM,
- **Úroveň 3** – autentizované párování + šifrování pomocí AES–CCM,
- **Úroveň 4** – lepší autentizované párování pomocí Diffie–Hellmanova protokolu s využitím eliptických křivek + šifrování pomocí AES–CCM.

Druhý mód zajišťuje bezpečnost pomocí podpisování dat a skládá se ze dvou úrovní:

- **Úroveň 1** – neautentizované párování + podpis dat,
- **Úroveň 2** – autentizované párování + podpis dat.

Pro tento projekt jsem zvolil čtvrtou úroveň prvního módu. Jedná se o nejlépe zabezpečenou šifrovanou komunikaci.

4.3 Testování

Po úspěšném dopsání tohoto projektu jsem podrobil USB zařízení i Android aplikaci důkladnému testování. Na straně zařízení Raspberry Pi jsem testoval především:

- data vysílaná při BLE advertisingu,
- strukturu GATT serveru,
- možnost připojení se k zařízení pomocí BLE z různých aplikací různých mobilních telefonů,
- možnost odpojení se a opětovného připojení,
- simulaci komunikace s vytvářenou Android aplikací pomocí vhodných nástrojů,
- přepínání různých USB gadget během jednoho spuštění zařízení,
- konkrétní chování všech dostupných USB gadget,
- funkčnost všech implementovaných USB gadget na nejpoužívanějších operačních systémech (Microsoft Windows, Linux, macOS).

V Android aplikaci jsem podrobně testoval funkčnost všech vypsaných případů užití v sekci 3.3.1. Dále jsem také testoval:

- možnost připojení více zařízení najednou (bylo testováno s 5 zařízeními),
- možnost přepínání ovládání mezi více připojenými zařízeními v jedné aplikaci,
- funkčnost aplikace na různých verzích operačního systému Android (od verze 6.0 až po verzi 9.0),
- funkčnost aplikace na mobilních telefonech od různých výrobců,
- správné zpracovávání BLE požadavků v aplikaci a frontu na tyto požadavky.

Podle výsledků jednotlivých testů jsem případně opravil nedostatky na obou implementovaných stranách tak, aby vše fungovalo, jak se očekává. Nyní jsou všechny požadavky na projekt splněny a všechny testy dopadly podle očekávaných výsledků.

Závěr

Cílem této práce bylo vytvořit bezdrátově ovladatelné zařízení z Raspberry Pi Zero, které bude po připojení pomocí USB do libovolného počítače simulovat různé běžně používané USB periferie. Toto chytré zařízení je ovládáno pomocí vytvořené aplikace z mobilního telefonu s operačním systémem Android. Pro bezdrátovou komunikaci mezi mobilním telefonem a USB zařízením byla vybrána technologie Bluetooth Low Energy.

Jako ukázku funkčnosti tohoto zařízení jsem zvolil možnost simulování tří často používaných USB periferií – klávesnice, myši a flash disku. V mobilní aplikaci si uživatel po připojení k zařízení vybere nějakou z nabízených periferií. Připojené zařízení se pro hostitelský počítač přemění ve vybranou periferii a tuto periferii pak uživatel ovládá z mobilní aplikace.

Ve své práci jsem dbal na bezpečnost komunikace mezi zařízeními, na hladkou funkčnost mobilní aplikace na většině dostupných Android zařízeních, na možnost propojení více USB zařízení k jedné aplikaci v jednu chvíli a na výsledné testování vytvořeného produktu.

Jelikož jsem zatím neobjevil žádný podobný projekt, který by se tímto problémem zabýval, chtěl bych v tomto projektu pokračovat i nad rámec své bakalářské práce a vylepšit ho a rozšířit o další možné funkcionality. Do budoucna je mým cílem rozšířit počet podporovaných USB periferií, zařízení by dále mohlo podporovat například simulaci CD/DVD mechaniky, použití zařízení jako prezentéru, joysticku nebo simulaci ethernetového připojení, zatímco ve skutečnosti budou data odesílána přes Wi-Fi. Dále by bylo možné vytvořit vlastní jednoduchou linuxovou distribuci pro zařízení Raspberry Pi, která bude obsahovat pouze minimální potřebný software pro funkčnost projektu – zvýšilo by se tím dostupné místo na SD kartě a rychlost bootování. Ačkoliv je tento projekt poměrně rozsáhlý, má ještě spoustu možností dalšího vývoje a rád na něm budu pracovat i nadále.

Bibliografie

1. OPENSOURCE.COM. *What is a Raspberry Pi?* [online] [cit. 2019-04-04]. Dostupné z: https://opensource.com/resources/raspberry-pi.
2. ESCOBAR, Eric. *What Is the Raspberry Pi?* [online]. 2013 [cit. 2019-04-05]. Dostupné z: https://www.quickanddirtytips.com/tech/computers/what-is-the-raspberry-pi.
3. RASPBERRY PI FOUNDATION. *FAQs* [online] [cit. 2019-04-05]. Dostupné z: https://www.raspberrypi.org/documentation/faqs/.
4. RASPBERRY PI FOUNDATION. *Raspberry Pi Zero W* [online] [cit. 2019-04-05]. Dostupné z: https://www.raspberrypi.org/products/raspberry-pi-zero-w/.
5. PIETRASIEWICZ, Andrzej. *Modern USB gadget on Linux and how to integrate it with systemd (Part 1)* [online]. 2019 [cit. 2019-04-05]. Dostupné z: https://www.collabora.com/news-and-blog/blog/2019/02/18/modern-usb-gadget-on-linux-and-how-to-integrate-it-with-systemd-(part-1)/.
6. BRAIN, Marshall. *How USB Ports Work* [online]. 2000 [cit. 2019-04-06]. Dostupné z: https://computer.howstuffworks.com/usb.htm.
7. JEFF TYSON, TRACY V. WILSON. *How Computer Keyboards Work* [online]. 2000 [cit. 2019-04-06]. Dostupné z: https://computer.howstuffworks.com/keyboard.htm.
8. USB IMPLEMENTERS' FORUM. *Device Class Definition for Human Interface Devices (HID)*. 2001. Verze 1.11. Dostupné také z: https://www.usb.org/sites/default/files/documents/hid1_11.pdf.
9. USB IMPLEMENTERS' FORUM. *HID Usage Tables*. 2004. Verze 1.12. Dostupné také z: https://www.usb.org/sites/default/files/documents/hut1_12v2.pdf.

10. MICROSOFT CORPORATION. *Enhanced Wheel Support in Windows*. 2010. Dostupné také z: http://download.microsoft.com/download/B/D/1/BD1F7EF4-7D72-419E-BC5C-9F79AD7BB66E/Wheel.docx.

11. TYSON, Jeff. *How Flash Memory Works* [online]. 2000 [cit. 2019-04-07]. Dostupné z: https://computer.howstuffworks.com/flash-memory.htm.

12. LINUX KERNEL DOCUMENTATION. *Linux Kernel Documentation. Mass storage*. Dostupné také z: https://www.kernel.org/doc/Documentation/usb/mass-storage.txt.

13. CALLAHAM, John. *The history of Android OS: its name, origin and more* [online]. 2018 [cit. 2019-04-09]. Dostupné z: https://www.androidauthority.com/history-android-os-name-789433/.

14. LEE, Joel. *To Build an Android App, You Need to Learn These 7 Programming Languages* [online]. 2017 [cit. 2019-04-09]. Dostupné z: https://www.makeuseof.com/tag/build-android-app-programming-languages/.

15. TUTORIALS POINT. *Android – Application Components* [online] [cit. 2019-04-20]. Dostupné z: https://www.tutorialspoint.com/android/android_application_components.htm.

16. GOOGLE AND OPEN HANDSET ALLIANCE N.D. *Android documentation. Understand the Activity Lifecycle* [online] [cit. 2019-04-10]. Dostupné z: https://developer.android.com/guide/components/activities/activity-lifecycle.

17. DIFFEN.COM. *Bluetooth vs. Wi-Fi* [online] [cit. 2019-04-10]. Dostupné z: https://www.diffen.com/difference/Bluetooth_vs_Wifi.

18. PLUGINTOIOT.COM. *Bluetooth Classic Vs Bluetooth Low Energy (BLE): Which one is right for you?* [online]. 2016 [cit. 2019-04-10]. Dostupné z: https://www.plugintoiot.com/bluetoothclassic-vs-bluetoothlowenergy/.

19. RF WIRELESS WORLD. *Bluetooth vs BLE – difference between Bluetooth and BLE (Bluetooth Low Energy)* [online] [cit. 2019-04-10]. Dostupné z: http://www.rfwireless-world.com/Terminology/Bluetooth-vs-BLE.html.

20. DAVIDSON, ROBERT A DALŠÍ. *Getting Started with Bluetooth Low Energy. Chapter 4. GATT (Services and Characteristics)* [online] [cit. 2019-04-10]. Dostupné z: https://www.oreilly.com/library/view/getting-started-with/9781491900550/ch04.html.

21. THINKGEEK, INC. *Phantom Keystroker* [online] [cit. 2019-04-11]. Dostupné z: https://www.thinkgeek.com/product/jknk/.

22. DAS, Abhilekh. *5 Best Applications To Control PC From Android Smartphone* [online]. 2017 [cit. 2019-04-11]. Dostupné z: `https://fossbytes.com/android-apps-control-pc/`.

23. RASPBERRY PI FOUNDATION. *Raspbian* [online] [cit. 2019-04-13]. Dostupné z: `https://www.raspberrypi.org/downloads/raspbian/`.

24. BLUEZ PROJECT. *About* [online] [cit. 2019-04-13]. Dostupné z: `http://www.bluez.org/about/`.

25. GOOGLE AND OPEN HANDSET ALLIANCE N.D. *Android documentation. Bluetooth low energy overview* [online] [cit. 2019-04-20]. Dostupné z: `https://developer.android.com/guide/topics/connectivity/bluetooth-le`.

26. GOOGLE AND OPEN HANDSET ALLIANCE N.D. *Android documentation. BluetoothDevice* [online] [cit. 2019-04-20]. Dostupné z: `https://developer.android.com/reference/android/bluetooth/BluetoothDevice`.

27. GOOGLE AND OPEN HANDSET ALLIANCE N.D. *Android documentation. Permissions overview* [online] [cit. 2019-04-20]. Dostupné z: `https://developer.android.com/guide/topics/permissions/overview.html`.

28. STATCOUNTER.COM. *Mobile and Tablet Android Version Market Share Worldwide* [online]. 2019 [cit. 2019-04-20]. Dostupné z: `http://gs.statcounter.com/android-version-market-share/mobile-tablet/worldwide`.

29. DUQUE, Alexis. *Deep Dive into Bluetooth LE Security* [online]. 2018 [cit. 2019-04-20]. Dostupné z: `https://medium.com/rtone-iot-security/deep-dive-into-bluetooth-le-security-d2301d640bfc`.

Příloha A

Seznam použitých zkratek

API Application Programming Interface

BLE Bluetooth Low Energy

BT Bluetooth

GATT Generic Attribute Profile

HID Human Interface Device

LE Low Energy

MAC Media Access Control

OTG On The Go

RAM Random–Access Memory

SDK Software Development Kit

UDC USB Device Controller

USB Universal Serial Bus

UUID Universally Unique Identifier

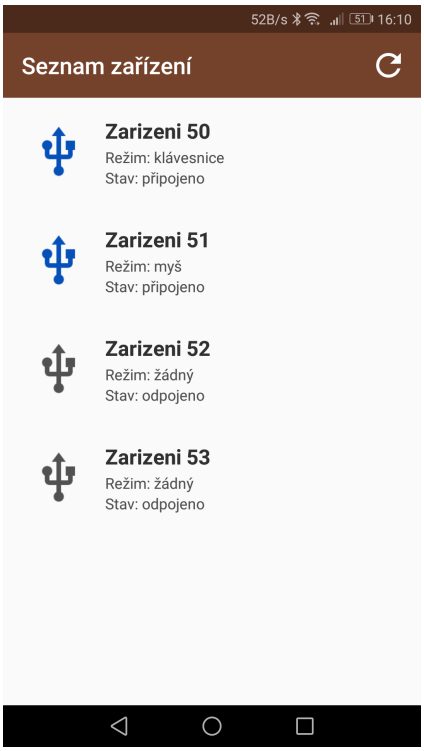
Příloha **B**

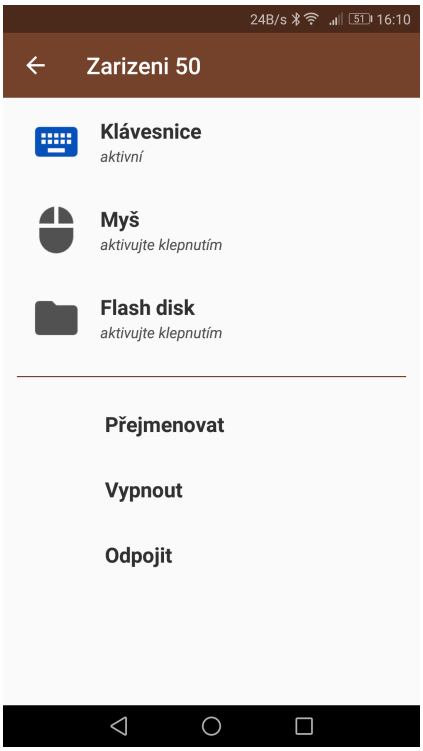
Obsah přiloženého CD

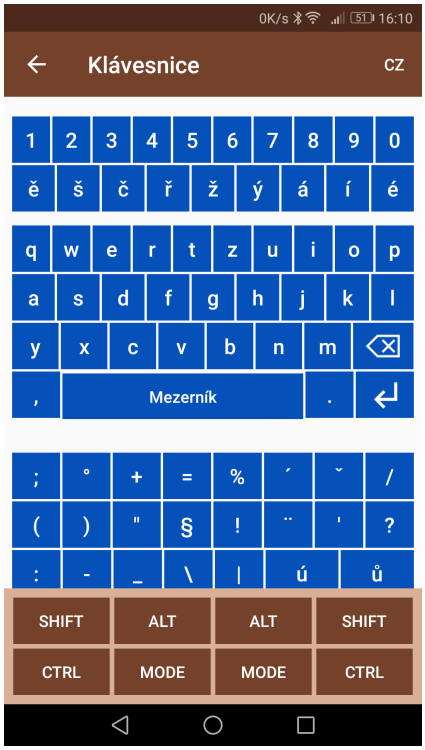
- `readme.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou formou implementace
  - `SmartUSBdevice.apk` ..... instalační soubor Android aplikace
- `src`
  - `impl` ..... zdrojové kódy implementace
    - `android` ..... zdrojové kódy Android aplikace
    - `raspberry` ..... zdrojové kódy pro Raspberry Pi
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF

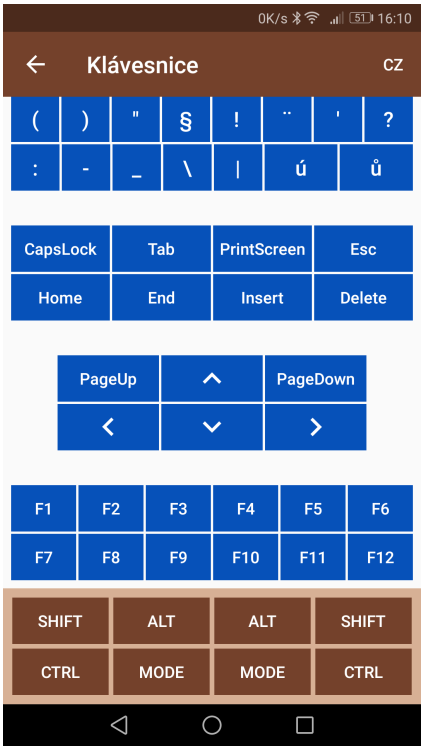
Příloha C

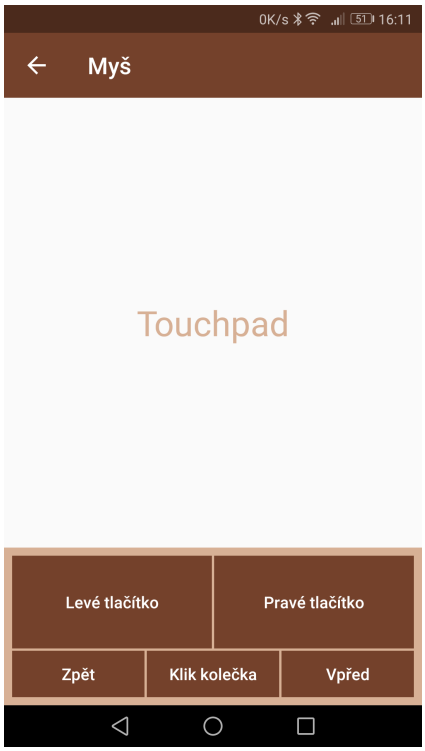
Grafická podoba mobilní aplikace

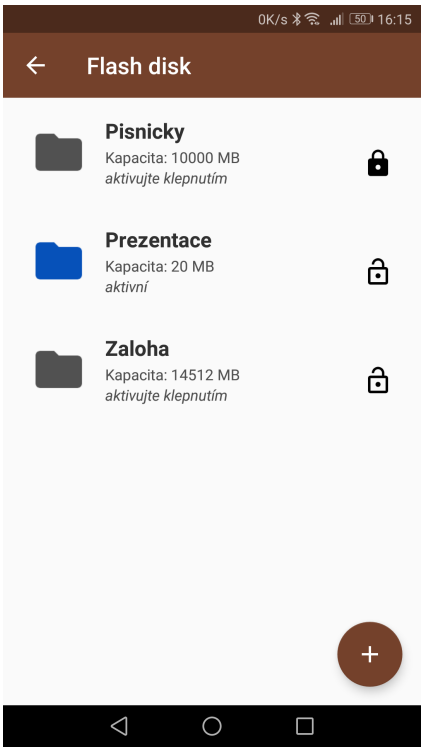


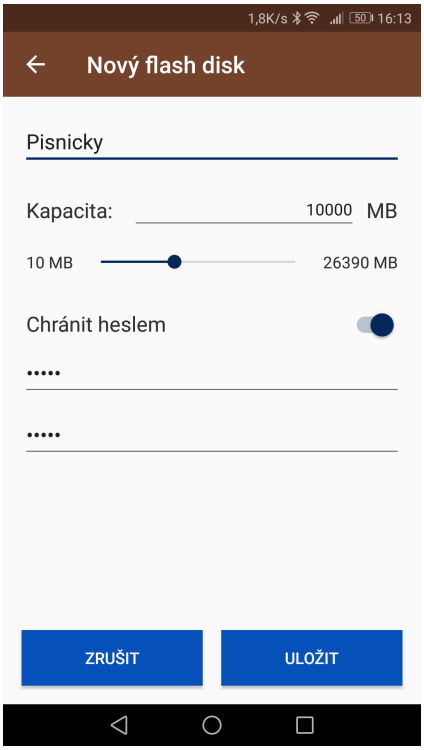


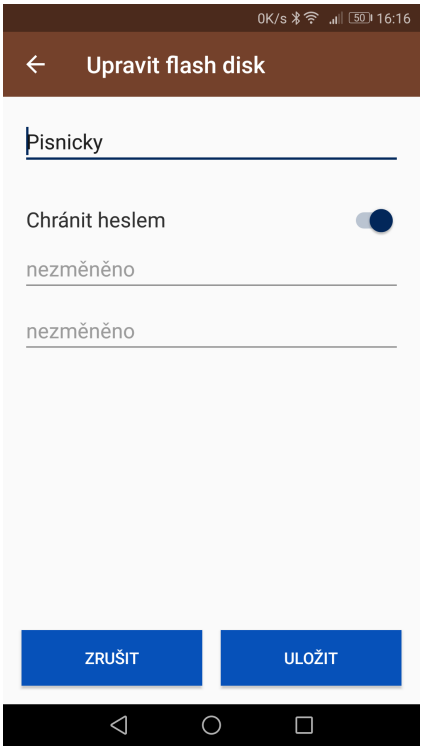

0K/s 16:10
Klávesnice
CZ
1 2 3 4 5 6 7 8 9 0
ě š č ř ž ý á í é
q w e r t z u i o p
a s d f g h j k l
y x c v b n m
,
Mezerník
.
; ° + = % ´ ˇ /
( ) " § ! ¨ ' ?
: - _ \ | ú ů
SHIFT ALT ALT SHIFT
CTRL MODE MODE CTRL

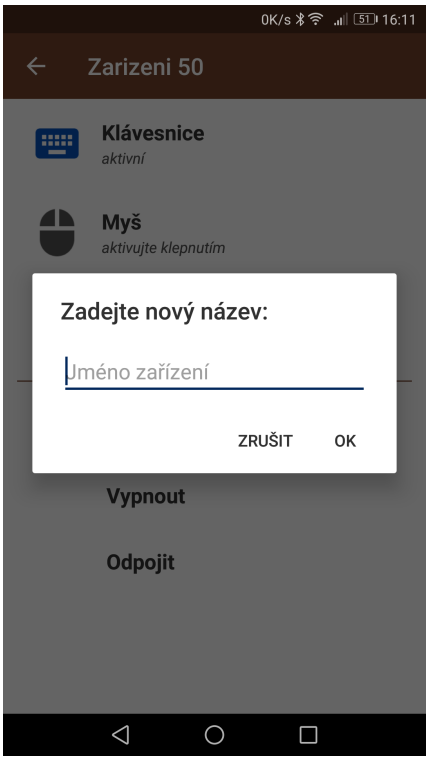

0K/s 16:10
Klávesnice
CZ
( ) " § ! ¨ ' ?
: - _ \ | ú ů
CapsLock Tab PrintScreen Esc
Home End Insert Delete
PageUp PageDown
F1 F2 F3 F4 F5 F6
F7 F8 F9 F10 F11 F12
SHIFT ALT ALT SHIFT
CTRL MODE MODE CTRL


0K/s 16:11
Myš
Touchpad
Levé tlačítko
Pravé tlačítko
Zpět
Klik kolečka
Vpřed


0K/s 16:15
Flash disk
Pisnicky
Kapacita: 10000 MB
aktivujte klepnutím
Prezentace
Kapacita: 20 MB
aktivní
Zaloha
Kapacita: 14512 MB
aktivujte klepnutím


1,8K/s 16:13
Nový flash disk
Pisnicky
Kapacita:
10000
MB
10 MB
26390 MB
Chránit heslem
•••••
•••••
ZRUŠIT
ULOŽIT


0K/s 16:16
Upravit flash disk
Pisnicky
Chránit heslem
nezměněno
nezměněno
ZRUŠIT
ULOŽIT


0K/s 16:11
Zarizeni 50
Klávesnice
aktivní
Myš
aktivujte klepnutím
Zadejte nový název:
Jméno zařízení
ZRUŠIT
OK
Vypnout
Odpojit