Západočeská univerzita v Plzni

Fakulta elektrotechnická
Katedra elektroniky a informačních technologií

Bakalářská práce

Chytrá aplikace pro zrakově a sluchově postižené jedince

Autor práce: **Tomáš Mareš**
Vedoucí práce: **Ing. Vladimír Pavlíček, Ph.D.**
Konzultant práce: **Ing. Vladimír Pavlíček, Ph.D.**

2023

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: 2022/2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Tomáš MAREŠ**
Osobní číslo:	**E19B0251P**
Studijní program:	**B2612 Elektrotechnika a informatika**
Téma práce:	**Chytrá aplikace pro zrakově a sluchově postižené jedince**
Zadávající katedra:	**Katedra elektroniky a informačních technologií**

Zásady pro vypracování

1. Prostudujte možnosti komunikace s lidmi postiženými ztrátou zraku, sluchu a mluveného projevu (hlucho-slepo-němí jedinci).
2. Navrhněte jednoduchou aplikaci pro tablet, schopnou zobrazovat zadaný krátký text, ve vhodném formátování a volitelné velikosti fontu, na displeji tabletu.
3. Vytvořte funkcionalitu aplikace, schopnou sledovat pozici přiloženého prstu na displeji, detekovat zobrazovaný znak textu pod prstem a která tento znak zašle ve vhodné formě na spárované zařízení.
4. Navržené zařízení realizujte a vyzkoušejte v praxi.

Rozsah bakalářské práce: **30 – 40**
Rozsah grafických prací: **5**
Forma zpracování bakalářské práce: **elektronická**

Seznam doporučené literatury:

1. Vývoj aplikací pro Android / iOS.
2. Emily C.Bouck: Assistive Technology, SAGE Publications 2015, ISBN 1483374459.
3. Hersch, Johnson: Assistive Technology for the Hearing-impaired, Deaf and Deafblind, SpringerLink 2015, ISBN 978-1447139218.
4. Duální senzorické onemocnění: www.ddmsberoun.cz
5. LORMOVA abeceda, wiki https://www.lorm.cz/
6. Taktilní senzory.

Vedoucí bakalářské práce: **Ing. Vladimír Pavlíček, Ph.D.**
Katedra elektroniky a informačních technologií

Datum zadání bakalářské práce: **7. října 2022**
Termín odevzdání bakalářské práce: **26. května 2023**

L.S.

Prof. Ing. Zdeněk Peroutka, Ph.D.
děkan

Doc. Ing. Jiří Hammerbauer, Ph.D.
vedoucí katedry

V Plzni dne 7. října 2022

Abstrakt

Cílem této bakalářské práce je vytvořit mobilní aplikaci pro tablet, kterou jsou schopni používat hluchoslepí lidé a číst z ní a k tomu vytvořit další aplikaci, na které bude toto čtení posíláno. Tato práce se bude skládat ze dvou částí, jedna bude teoretická a ta druhá bude praktická.

V teoretické části budu popisovat zdravotní postižení jako je slepota, hluchota a hluchoslepot. Bude zmíněno, jak se od sebe liší a jejich druhy. V dalším úseku budu popisovat různé pomůcky, které jsou využívány hluchoslepými osobami pro komunikaci s ostatními lidmi a budou popsány různé platformy pro vytváření mobilních aplikací a na jakém softwaru se využívají.

V praktické části se budu zabývat tím, jakou platformu jsem si vybral, jak tuto platformu použít pro vytvoření mobilní aplikace, tedy i vytvořením projektu v ní. V další sekci budu popisovat můj postup pro vytvoření dvou aplikací s popisem vzhledu a programovacího kódu, kde bude jedna aplikace dělaná pro personál či rodinu a ta druhá bude pro hluchoslepou osobu.

Klíčová slova

Hlucho-slepo-němí jedinci, Alternativní a augmentativní komunikační systémy, Hluchoslepota, Tvorba mobilní aplikace, Mobilní aplikace pro iOS a Android, Firebase, Android Studio

Abstract

The aim of this bachelor thesis is to create a mobile application for a tablet that deafblind people are able to use and read from, and to create another application to which this reading will be sent. This thesis will consist of two parts, one will be theoretical and the other will be practical.

In the theoretical part I will be describing disabilities such as blindness, deafness and deafblindness. It will be mentioned how they differ from each other and their types. In the next section, I will describe the different aids that are used by deafblind people to communicate with other people and the different platforms for creating mobile apps and what software they are used on will be described.

In the practical section I will discuss what platform I have chosen, how to use this platform to create a mobile application, including creating a project in it. In the next section, I will describe my process for creating two apps, describing the look and feel and programming code, where one app will be made for staff or family and the other will be for a deafblind person.

Keywords

Deafblind-mute individuals, Alternative and augmentative, Deafblindness, Mobile application development, Mobile application for iOS and Android, Firebase, Android Studio

Prohlášení

Prohlašuji, že tato práce je mým autorským dílem, které jsem vypracoval samostatně. Všechny zdroje, prameny a literaturu, které jsem použil a z nichž jsem čerpal v práci cituji s uvedením úplného odkazu na příslušný zdroj.

V Plzni dne ..... podpis studenta .....

Obsah

Seznam obrázků

Úvod

V našem okolí existuje několik lidí, kteří trpí vážným postižením jako jsou slepota, hluchota nebo obojí. Tato postižení výrazně ztěžují jejich každodenní život ve společnosti. Ovlivňují jejich komunikaci s vnějším světem a to, jak po informační stránce, tak i po stránce komunikace, a tudíž mohou mít velmi nedozírné následky z pohledu psychiky. Proto se při nástupu chytrých telefonů začaly i pro takto postižené lidi vyvíjet aplikace. Tyto aplikace byly a jsou vytvářeny přesně pro jejich potřeby, aby začaly s postupem času více vnímat svět kolem sebe a umožnily jim přístup k více informacím nebo ke komunikačním prostředkům.

Moje bakalářská práce pojednává o vytvoření aplikace zejména pro osoby se hluchoslepotou. Tuto práci jsem se rozhodl rozdělit na dvě části, kde první část bude obsahovat definice těchto postižení a bude probrána tématika alternativní komunikace pro tyto osoby, zejména jak se dorozumívají. Také tato část bude obsahovat popis platforem, které se dají využít pro programování mobilních aplikací.

Druhá část bakalářské práce bude spíše praktická, kde budu věnovat důraz na to, jakou platformu jsem si vybral pro vytvoření těchto mobilních aplikací. Tato část bude rozdělena na to, jak začíst pracovat s vybranou platformou, jak v ní vytvořit projekt a co vše tento projekt obnáší. Jako další budu popisovat první vytvořenou aplikaci, která je směřována pro rodinu, či pečovatelský personál, který se bude o postiženou osobu starat. Tato aplikace bude mít možnost posílat text dané handicapované osobě a také zjistit, jak na tom jsou se čtením této zprávy. Poté popíši, jaká jsou možná spojení se spárovaným zařízením. Jako poslední sekce mé praktické části bude vysvětlení funkce druhé aplikace, která je mířena pro hluchoslepé osoby. Budu se zabývat rozborem kódu a jeho rozhraní, vysvětlením, jak se posílají písmena na spárované zařízení a co je používáno pro detekci dotyku.

Závěrem mé bakalářské práce je vyvinout aplikaci pro hluchoslepé, která by jim v budoucnu mohla být nápomocná ve čtení zpráv. Aplikaci připravit na budoucnost, kde by pro ni bylo možné udělat další zařízení, které by toto čtení zpráv přenášelo do fyzické podoby.

1 Teoretická část

1.1 Zrakový handicap

Zrakový handicap je postižení, při kterém člověk přestává vidět či vůbec nevidí. Tento handicap je jedním z vážných postižení, protože tento člověk prožívá svět kolem sebe bez toho, aby mohl pohledět na okolí před sebou. Zrakový handicap vzniká z několik důvodů jako jsou nemoci, úrazy nebo genetické vady.

Ztráta zraku je velmi náročné postižení, jelikož člověk ztrácí vidění kolem sebe a tudíž i orientaci a pohyb v prostoru. Tento jedinec také ztrácí přehled ve svém životě, jelikož pro něj může být těžké rozpoznávat tváře, číst či psát. Může dojít i k takové situaci, kde se osoby trpící tímto postižením distancují od ostatních lidí a jejich okolí. Toto může mít pak pro tyto lidi psychické následky.

I když o tomto postižení mluvím jako vážném, tak ho ale nepovažuji za nejhorší. Člověk s tímto handicapem se nadále může účastnit komunikace s ostatními lidmi. Tento handicap ho neodděluje od lidí, ale od věcí. Pro tyto lidi existuje několik pomůcek, které jim pomáhají. Je to třeba slepecká hůl nebo Braillovo písmo, ve kterém jsou pro ně psané knihy a jsou vydávané společností Knihovna a tiskárna pro nevidomé K. E. Macana v Praze. Významnou roli v pomoci těmto lidem také hrají různé organizace či služby jako například Sjednocená organizace nevidomých a slabozrakých, která třeba zařizuje výcvik vodivých psů.

Zrakové postižení je určováno pomocí Snellenovo zlomku, což je pomůcka pro určování zrakové ostrosti. Zrakový handicap lze rozdělit do pěti skupin, což jsou střední slabozrakost (6/18), silná slabozrakost (6/60), těžce slabý zrak (3/60), praktická slabozrakost (1/60) a úplná slepota. Kdyby chom tady určovali kvalitu vidění u těžce slabého zraku, tak se podle Snellenovo zlomku dozvíme, že kvalita vidění je 0,5/10. To nám udává, že tento člověk vidí objekt jasně do 3 metrů, což u normálního člověka je 60 metrů. [1]

1.2 Sluchový handicap

Sluchové postižení je handicap, při kterém člověk přestává slyšet nebo už neslyší. Toto postižení je vážné, může člověka ovlivnit v různých úrovních jako třeba komunikace s ostatními lidmi. Toto může mít vliv na jeho psychologii zejména u malých dětí, kde dochází k pozdějšímu rozvoji myšlení.

Ztráta sluchu je velmi vážné postižení, protože člověk ztrácí povědomí o tom, co se kolem něj odehrává a ztrácí způsob získávání informací mluvenou formou. Pro tyto osoby existuje několik pomůcek. Jedna z nich je třeba kochleární implantát, který slouží pro kompenzaci sluchové poruchy. Další pomůcky jsou například prstová abeceda, znakový jazyk či odezírání. Také tady existují organizace, které se zabývají pomocí těmto osobám jako je například skupina České unie neslyšících.

Definování sluchového postižení můžeme rozdělit na dva typy. Jsou to vady sluchu a ztráty sluchu. Vady sluchu můžeme rozdělit na to, kdy tyto vady vznikly, zda byly získané dědičně či již po narození. Dále se dělí na to, zda daná osoba zná řeč a jazyk či ne. Dále se dělí podle místa postižení, zda je příčina ve středním uchu či vlivem mozku. Ztráta sluchu se definuje pomocí decibelů, kde velikost ztrát u normálního člověka je 0-25 dB. Ztrátu sluchu můžeme dělit na lehkou nedoslýchavost (26–40 dB), středně těžkou nedoslýchavost (41–55 dB), těžkou nedoslýchavost (56–70 dB), praktickou hluchota (71–90 dB) a úplnou hluchotu (nad 90 dB). [2]

1.3 Hluchoslepota

Mezinárodní definice přijatá při založení Evropské unie hluchoslepých (EDBU): „Hluchoslepota je jedinečné postižení, které je způsobeno různorodými kombinacemi sluchového a zrakového postižení. Způsobuje potíže při komunikaci a sociální a funkční interakci a zabraňuje plnohodnotnému zapojení do společnosti.“ (lorm.cz)

Definice v Písemném prohlášení 1/2004 o právech hluchoslepých osob přijatá Evropským parlamentem 1. 4. 2004: „Hluchoslepota je jedinečné postižení vzniklé kombinací zrakového a sluchového poškození, které způsobuje potíže v přístupu k informacím, komunikaci a mobilitě.“ (lorm.cz)

Hluchoslepota je postižení, kde člověk ztratí sluch a zrak. Toto postižení může mít vážné následky na komunikaci s ostatními lidmi a orientaci v prostoru. Může mít vliv na přijímání a zpracování informací z okolí nebo při komunikaci s přáteli a rodinou. Hluchoslepotu lze dělit na různé úrovně dle ztráty sluchu a zraku.

Hluchoslepota může mít několik příčin. Mohou to být genetické vady, infekce, traumatická poranění hlavy nebo stárnutí. Jedna z nejčastějších vad z této kategorie je genetická vada, z čehož je Usherův syndrom ten nejčastější. Hluchoslepota v kategorii genetické vady je způsobena mutací v genu, který má na starost vyvíjení zraku a sluchu a tak může vést k jejich poruchám. Tyto mutace jsou často dědičné. Další kategorií hluchoslepoty jsou infekce, které jsou především způsobeny napadením plodu a představují vážné problémy u novorozence. Lze je dělit do několika druhů, například cytomegaloviróza, toxoplazmóza nebo rubeola. Další kategorií je traumatické poranění hlavy, které je způsobeno poraněním hlavy například při dopravní nehodě nebo pády. Poslední kategorií je stárnutí, kdy se s narůstajícím věkem může zhoršovat zrakový a sluchový systém, který může nakonec vést ke ztrátě zraku a sluchu.

Hluchoslepota je velmi vážné onemocnění, kdy člověk ztrácí ponětí o tom, kde se nachází a ztrácí způsob získávání informace mluvenou formou. Proto pro osoby s tímto postižením existuje několik pomůcek, které jim v tomto pomáhají. Jednou z nich jsou například sluchadla. Další je Prstová abeceda, Lormova abeceda, Braillovo písmo nebo Tadoma, které jim pomáhají v komunikaci s ostatními lidmi. Také existuje mnoho organizací a uskupení, které mají za úkol těmto lidem pomáhat. Jedná se třeba o Klub přátel červenobílé hole nebo LORM. [3] [4]

1.3.1 Komunikace a navigace

Komunikace a navigace pro hluchoslepé osoby je vážnou výzvou. Z důvodu ztráty zraku a sluchu, je ovlivněna jejich schopnost porozumět svému okolí a reagovat na ně. Existuje však několik pomůcek, které jim pomáhají překonat tuto výzvu v jejich životě.

Komunikace pro hluchoslepé může být velmi náročná, jelikož trpí ztrátou zraku a sluchu může pro ně být obtížné prolomit komunikační bariéru. Existuje několik systémů, které jim tuto překážku pomáhají překonat jako je například Braillovo písmo, znakový jazyk, Prstová abeceda, Tamado, Tiskací písmena psaná do dlaně nebo Psaná forma komunikace. Díky těmto komunikačním pomůckám mohou hluchoslepí lidé komunikovat s ostatními lidmi a začít se orientovat ve svém okolí.

Navigace v okolí je další výzva, kterou hluchoslepí lidé trpí. Jejich smysly jsou omezené. Naštěstí už existují technologie a nástroje, které si s tímto umí poradit a tak jim pomoci. Jedním z nich je například zvukový navigační systém, který využívá hlasové instrukce, pro hluchoslepé osoby, které ještě nemají vysoký stupeň hluchoty. Další je třeba bílá hůl, která je důležitá pro orientaci v prostoru pomocí hmatu a sluchu. Jako další technologie užitečná pro hluchoslepé lidi je GPS navigační systém a mobilní aplikace. Tyto aplikace mohou být významné třeba v poskytnutí trasy přes hlasový výstup nebo poskytnutí vizualizace prostředí přes vibrace či hmatné signály.

1.3.2 Organizace pro pomoc hluchoslepým

V České republice se nachází několik organizací a institucí, které se zabývají pomocí, těmto jedincům. Mohou zabývat výukou komunikačních prostředků s postiženými lidmi. Máme například:

LORM - Jedna z nejvýznamnějších organizací v ČR, která poskytuje výuku lormu a podporu lidem s tělesným postižením Sdružení Hluchoslepých ČR - Jedná se o Českou organizaci, která poskytuje podporu a pomoc lidem s hluchoslepým postižením. Nabízí různé služby, včetně vzdělávání a podporu v oblasti komunikace. [3]

Tyfloservis – Jedná se o společnost, která poskytuje služby osobám nevidomým a slabozrakým na území celé České republiky. [5]

Česká unie neslyšících - Jedná se o organizaci, která poskytuje informace a podporu pro neslyšící a hluchoslepé osoby v České republice. Mezi její aktivity patří také výuka lormu.

1.4 Alternativní a augmentativní komunikační metody (AAK)

Co je AAK?
"AAK je soubor nástrojů a strategií, které jedinec používá k řešení každodenních komunikačních výzev."
(podle Mezinárodní společnost pro augmentativní a alternativní komunikaci (ISAAC))

"AAK znamená všechny způsoby, jakými se komunikuje kromě mluvení. Lidé všech věkových kategorií mohou používat AAK pokud mají potíže s řečovými nebo jazykovými dovednostmi. Augmentativní znamená doplnit něčí řeč, alternativní znamená použít místo řeči."
(podle Americká asociace pro řeč, jazyk a sluch (ASHA))

Komunikace s lidmi, kteří trpí ztrátou sluchu, zraku či sluchu a zraku, není lehká. Je to z toho důvodu, že ve světě existuje několik komunikačních metod pro různé typy postižení a postižený člověk může znát pouze jednu. V komunikaci s těmito postiženými osobami hraje roli také prostředí, ve kterém se nachází, jako například hlučné prostředí či klidné prostředí nebo jak se člověk zrovna cítí.

Pro komunikaci s postiženými lidmi byly vynalezeny alternativní a augmentativní komunikační metody nebo-li AAK. Tento systém metod, prostředků a pomůcek slouží k tomu, aby člověk mohl nahradit mluvenou řeč jiným komunikačním systémem, díky němuž bude mít možnost postižený jedinec komunikovat s ostatními. Pro efektivitu těchto metod by se je měl postižený člověk učit co nejdříve, aby mohl vyjádřit své potřeby nebo emoce. Každý člověk se může v průběhu života naučit více komunikačních metod, aby mohl komunikovat s postiženou osobou.

Tyto metody pomáhají dětem zapojit se do školy i postarším lidem k zapojení do komunikace. Tyto metody byly vyvinuty proto, aby postižený člověk mohl komunikovat samostatně nebo s pomocí asistenta. Existují dvě kategorie metod. Jednou z nich je metoda bez pomůcek a druhou metodou je metoda s pomůckami.

Metoda bez pomůcek zahrnuje pohyb těla mluvícího, a to pomocí pohybu rukou, kde pomocí nich vyjadřujeme gesta nebo pomocí manuálních znaků jako je psaní do dlaně. Výhodou této metody je, že se používá mezi více postiženými lidmi a tím pádem si mohou navzájem porozumět. Metoda s pomůckami spočívá v tom, že postižený člověk si pomáhá hmatovými symboly, fotografií, komunikačními tabulkami nebo počítačem. Výhodou této metody je možnost vzdálené komunikace. Výběr metody je pak na každém člověku zvlášť a uživatel by měl být schopen využít více metod v různých situacích. [6]

1.4.1 Lormova abeceda

Lormova abeceda je speciální forma komunikace používaná pro osoby s částečnou nebo totální hluchoslepotou. Tento způsob komunikace jim umožňuje komunikovat s ostatními lidmi prostřednictvím doteků ukazováčkem do dlaně a prstů ruky. Je to založeno na principu toho, že jednotlivým prstům a sektorům dlaně jsou přiřazena různá písmena. (viz. příloha č.1).

Lormova abeceda se nejvíce používá pro levou ruku, její prsty jsou mírně protáhlé a roztažené. Pokud to nestačí, tak lze použít dlaň pravé ruky. Člověk komunikuje s hluchoslepou osobou tím, že dotykem ukazováčku přenáší jednotlivá písmena do dlaně adresáta. Příjemce pak může rozpoznat o jaké písmeno se jedná a převést ho do slov či vět. Lormova abeceda tak umožňuje lidem s hluchoslepotou komunikovat s ostatními lidmi bez použití mluvené komunikace.

Lormova abeceda je velmi důležitá pro lidi, kteří s hluchoslepými jedinci pracují. Jsou například speciální pedagogové, asistenti nebo tlumočníci. Tito lidé musí ovládat Lormovu abecedu, aby s těmito lidmi mohli spolupracovat a úspěšně s nimi komunikovat. Je to také důležité pro rozvoj bezbariérové komunikace, kde každý má možnost komunikovat a přijímat informace bez ohledu na zdravotní omezení.

1.4.2 Znakový jazyk

Znakový jazyk je komunikační systém, který umožňuje komunikaci pomocí vizuálních prostředků. Znakový jazyk využívá komunikaci pomocí gest, která jsou dělána prostřednictvím rukou a prstů. Jedná se o komunikační systém, který je využívaný osobami z těžkou ztrátou sluchu, ale dostatečným zbytkem zraku na rozeznání posuvků a gest. Tento jazyk se liší od mluveného jazyka nejen gramatikou, ale i slovní zásobou a strukturou vět. Má své vlastní základní jednotky, které jsou složeny z různých kombinací rukou a prstů či tvarů a poloh. Znakový jazyk se používá po celém světě a existuje mnoho různých variant a dialektů.

Člověk komunikující pomocí znakového jazyka, musí být dobře viditelný, protože zrakové vnímání je pro tento jazyk klíčové. Existuje mnoho institucí, které se tímto jazykem zabývají a vyučují ho jako například TichýSvět. Výhodou tohoto jazyka je jeho lehkost v porovnání s jazykem mluveným. Znakový jazyk nemá složitá gramatická pravidla a mnoho frází a slov lze vyjádřit pomocí malých gest. To znamená, že se osoba může rychle a jednoduše naučit základy znakové komunikace.

Znakový jazyk také umožňuje větší škálu v komunikaci. Osoba, která tímto jazykem komunikuje, může využít k vyjádření stejného významu různá gesta a posunky, jež umožňují větší škálu vyjadřovacích prostředků a tedy i širší jazykovou bohatost.

1.4.3 Prstová abeceda

Prstová abeceda, také známá pod názvem manuální abeceda, je komunikační systém, který umožňuje hluchoslepým lidem se zbytky sluchu a zraku komunikovat pomocí gest a pohybů prstů. Systém je založen na latinské abecedě a používá se k vyjádření jednotlivých písmen pomocí různých tvarů a poloh prstů. Písmena se tedy vyjadřují prostřednictvím pohybů prstů, které se pak spojují do slov. (viz. příloha č.2)

V České republice se používá prstová abeceda jednoruční nebo obouruční. Jednoruční prstová abeceda slouží především pro komunikaci s lidmi, kteří znají tento způsob komunikace, zatímco obouruční se používá při výuce nebo k vyjádření složitějších případů. Prstová abeceda je hlavně důležitá pro lidi s mírnou ztrátou sluchu a zraku a umožňuje rychlou a jednoduchou výměnu informací. Tato metoda je také vhodná pro hlučné prostředí. Výhodou tohoto jazyka je, že je snadno naučitelný a všeobecně uznávaný.

Tato metoda se tedy využívá pro komunikaci mezi hluchoslepými osobami a také pro komunikaci s lidmi mimo jejich prostředí. V současnosti je prstová abeceda stále používána jako způsob komunikace mezi hluchoslepými osobami a je součástí vzdělávacích programů pro ně. Rozvoj v technologií však umožnil využívat i jiné metody a zařízení jako jsou Braillovy řádky nebo speciální aplikace, které jsou užitečné pro komunikaci mezi hluchoslepými lidmi. [7] [8]

1.4.4 Braillovo písmo

Braillovo písmo je komunikační systém pro zrakově postižené nebo hluchoslepé. Dává jim možnost číst a psát, aniž by potřebovali zrak nebo sluch. Braillovo písmo vynalezl v 19. století Louis Braille, francouzský pedagog, který byl sám zrakově postiženým. (viz. příloha č. 3)

Braillovo písmo se skládá z vyvýšených bodů uspořádaných do řad a sloupců. Každý znak je skládán ze šesti bodů uspořádaných do dvou sloupců po třech řadách a dá se pomocí něj zobrazit abeceda, čísla, interpunkčí znaménka, matematické symboly a další znaky, které se v písmu nachází. Braillovo písmo je založené na tom, že člověk použitím prstů nahmatává vystouplé tečky a po hmatu rozeznává o jaký znak se jedná.

Braillovo písmo se používá po celém světě a má několik využití. Používá se především pro psaní a čtení knih, novin, časopisů a dalších publikací. Díky tomuto písmu mohou zrakově postižení lidé studovat a pracovat v určitých profesích. Toto písmo je důležité jelikož umožňuje komunikaci mezi zrakově postiženými a dává jim možnost psát a číst zprávy a dopisy.

S vývojem technologie se také změnil způsob použití Braillova písma. Dnes již existují speciální zařízení, která dokáže zobrazit Braillovo písmo na řádku s plastovými tečkami. Tato zařízení lze propojit s počítačem nebo mobilním zařízením, což umožňuje osobám se zrakovým postižením pracovat s texty a komunikovat s ostatními.

1.4.5 Tadoma

Tadoma je komunikační systém určený pro lidi, kteří jsou vrozeně hluchoslepí a nemají zbytky sluchu ani zraku. Tito lidé nejsou schopni používat tradiční alternativní komunikační systém jako je například znakový jazyk, prstová abeceda nebo Braillovo písmo. Proto byl pro ně vyvinut alternativní komunikační systém Tadoma, což je metoda založená na vnímání vibrací hrdla při řeči. (viz. příloha č. 4)

Hluchoslepý člověk se palcem dotkne rtů mluvčího a ostatní prsty přiloží ke tváři. Malíček se položí na krk mluvícího a s ním je vnímáno chvění hlasivek, které se přenáší přes krk a následně je cítit na prstu položením na krku. Tento systém využívá specifické vibrace, které se vyskytnou v dutině ústní a jsou tak cítit na krku. Vibrace jsou pak přeloženy do slov a vět, aby hluchoslepý porozuměl, o čem člověk mluví.

Tadoma se však velmi špatně učí a používá. Je to z důvodu toho, že ne každý člověk dokáže vnímat vibrace stejně dobře. Někteří s tím mají problém a je pro ně těžké cítit a rozeznávat specifické vibrace, které jsou potřebné k pochopení řeči. Tato metoda vyžaduje také trpělivost a čas, protože hluchoslepý člověk ne vždy porozumí mluvenému slovu.

I když Tadoma není běžně používanou metodou ke komunikaci, je cennou možností pro lidi, kteří jsou vrozeně hluchoslepí a nenaskytují se jim tak žádné jiné komunikační prostředky. Naučit se tento systém může být obtížné, ale pro některé lidi je to jediný způsob, jak porozumět světu kolem sebe.

Obrázek 1: Ukázka použití metody Tadoma

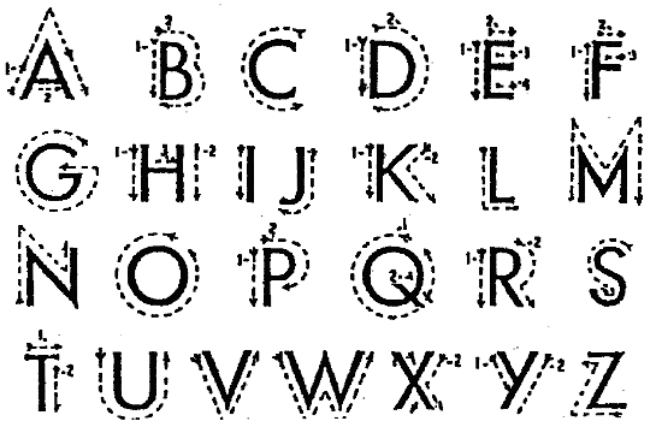
1.4.6 Tiskací písmena psaná do dlaně

Tiskací písmena psaná do dlaně je komunikační systém, který umožňuje lidem s hluchoslepostí komunikovat s ostatními pomocí psaných písmen do dlaně. Tento systém se hlavně doporučuje a používá u lidí, kteří ztratili zrak a sluch v pozdějším věku a mají stále citlivé dlaně.

Písmena jsou psána pravou rukou do levé ruky osobě s hluchoslepotou, ta pak odpovídá nazpátek stejným způsobem. Písmena jsou psána velkými tahy rukou, po kterých následuje definice určení směru tahů po dlani, což vyžaduje dobrou paměť a koncentraci. Dorozumívání pomocí těchto systémů může být pomalejší než v jiných komunikačních metodách, ale dovoluje

těmto lidem komunikovat s ostatními a získat větší samostatnost.

Tento komunikační systém má více výhod. Jednou z nich je jejich jednoduchost a finanční nenáročnost, jelikož nevyžaduje žádné speciální zařízení nebo technologii. Další výhodou je, že se může využívat kdekoliv bez ohledu na okolí. Nicméně tento komunikační systém má i své omezení. Jedním z nich je, že vyžaduje citlivost v dlani a dobrou paměť, což může být pro některé starší osoby obtížné. Další problém je v tom, že ne všechna písmena jsou dobře psaná a může tedy nastat problém v jejich rozpoznání. Lze tak říci, že tento systém vyžaduje trpělivost a soustředění.



Obrázek 2: Psaní do dlaně

1.4.7 Psaná forma komunikace

Psaná forma komunikace je pro mnohé lidi běžnou formou komunikace, ale pro hluchoslepé lidi představuje důležitou komunikaci při interakci s ostatními lidmi. Tento typ komunikace lze využít v situacích, jako například při rozhovoru s lékařem, úředníkem, či v běžném životě s rodinou a přáteli. Písemná forma dává hluchoslepým lidem příležitost vyjádřit své myšlenky a potřeby a také přijímat informace, které jsou důležité.

Pro psanou formu komunikace jsou užívané různé pomůcky jako například speciální pera a tužky, braillské psací stroje nebo počítač s hlasovým výstupem a speciálním softwarem. Dnes už jsou k dispozici unikátní telefony pro neslyšící, které umožňují psací formu a obsahují specifické funkce, jako například zvukové notifikace a světelné indikátory pro příchozí hovory a zprávy.

Písemná komunikace může být pro hluchoslepé lidi velmi užitečným nástrojem komunikace a zapojení se do společnosti. Naučit se psát a číst psaný text však může být pro některé lidi obtížné, zejména pro ty, kteří ztratili sluch a zrak v mladším věku. Proto je důležité, aby tito lidé měli potřebnou pomoc a aby se jim dostalo podpory, která jim pomůže v učení psání a čtení.

1.5 Mobilní aplikace a přístroje

Hluchoslepota je postižení, v němž lidé ztrácí sluch a zrak. Takto postižení lidé přestávají mít přístup k informacím a ke komunikaci se svými přáteli na dálku, což může mít dopad na jejich psychiku. Proto se s rozvojem mobilních telefonů a technologií objevily další možnosti, které mohou pomoci lidem s hluchoslepotou a tak zajistit jejich nezávislost na ostatních a pomoci jim k získání k informací.

V této době existuje několik technologií a mobilních aplikací, které byly vytvořeny pro pomoc lidem s handicapem k získání informací a ke komunikaci na dálku s ostatními lidmi. Do mobilních aplikací zařazujeme například aplikace jako je Be my Eyes, Seeing AI, TaklBack nebo VoiceOver. Dále do technologií zařazujeme pomůcky jako je Braillovo řádek nebo sluchadla kotvená do kosti.

1.5.1 VoiceOver

VoiceOver je pomocná aplikace pro lidi se ztrátou zraku. Tato aplikace je součástí operačního systému iOS pro mobilní zařízení od společnosti Apple. Tato aplikace má za úkol pomoci lidem se zrakovým postižením k zapojení se do internetových aktivit nebo k použití telefonu. Aplikace VoiceOver funguje na principu toho, že všechno co se pod prstem nachází nebo děje přemění do mluvené podoby.

Hlavní funkcí VoiceOveru je poskytnout hlasovou zprávu o tom, co se děje na obrazovce. Díky této funkci je možné pro zrakově postižené ovládat svá zařízení a snížit tak jejich závislost na ostatních lidech. VoiceOver také funguje jako navigátor v zařízení a umožňuje uživatelům snáze najít ikony na obrazovce, přepínat mezi různými aplikacemi a používat různé funkce zařízení.

Tato aplikace nejenže zlepšuje nezávislost a sebejistotu postiženého člověka, ale také přispívá k většímu povědomí o potřebách takto postižených osob a jejich zapojení do společnosti. Aplikace rovněž pomáhá osobám k zapojení do digitálního světa, kde mohou komunikovat a psát si s rodinou nebo zjistit informace ze zpráv.

1.5.2 TalkBack

TalkBack je také pomocná aplikace pro lidi se ztrátou zraku. Aplikace je součástí operačního systému Android. Byla vyvinuta společností Google a využívá moderní technologie, aby svým uživatelům poskytla jednoduší použití mobilních zařízení. Funguje na stejném principu jako VoiceOver.

Hlavní funkcí aplikace je hlasový výstup, přes který je slyšet informace o tom, co se děje na obrazovce mobilního zařízení. Tato funkce dává možnost uživatelům snáze ovládat své zařízení, psát textové zprávy, procházet aplikace a vyhledávat informace na internetu. Další funkcí aplikace je zlepšení orientace v mobilním zařízení, pomoc rychle nalézt ikony aplikací na obrazovce. Přináší lepší ovládání pro osoby se zrakovým omezením a snižuje jejich závislost na ostatních lidech či

pomůckách.

Aplikace TalkBack má mnohostranný dopad na společnost. Jednou z nejdůležitějších výhod je zlepšení nezávislosti na dalších lidech a také pomoc v jejich sebedůvěře. Tato aplikace jim umožňuje dostat se do digitálního světa a účastnit se všeho co normální člověk dokáže jako například komunikace, vzdělání nebo zábava. Slepí lidé se mohou připojit na internet, což vede k jejich porozumění a uznání. [9]

1.5.3 Seeing AI

Seeing AI je aplikace vytvořená Microsoftem, která je určena pro zrakově handicapované. Tato aplikace je využívána pro rozpoznání textu, barvy, objektu a dokonce i lidí prostřednictvím zpracování obrazu. Seeing AI je založena na využití fotoaparátu chytrého telefonu nebo tabletu, přes který pak uživatel dostává popis svého okolí.

Jednou z hlavních funkcí je čtení textu, který pak umožňuje čtení knihoven, štítků a dalších psaných materiálů. Tato aplikace také umožňuje přepisování textu do zvukové podoby, což je dobré pro lidi, kteří se potýkají s horší vadou zraku. Další funkcí této aplikace je detekce objektů a jejich následné rozpoznání. Díky této aplikaci lze identifikovat nádobí, člověka nebo nějaké předměty v domácnosti či okolí. Součástí aplikace je i rozpoznávání lidí a sdělení zrakově postiženému člověku, zda se člověk přibližuje nebo ne. Aplikace také pomáhá zjistit, zda se někdo nachází v místnosti.

Dopad této aplikace na společnost je obrovský, jelikož pomáhá zrakově handicapovaným se zapojením do společnosti. Díky této aplikaci lze dávat informace o jejich okolí. Pomáhá jim v získání informací, které by pro ně byly jinak nedostupné. Aplikace jim udává například jak je daný člověk vzdálený od objektu či popisuje barvu, která se nachází v okolí.

Budoucnost Seeing AI je plná možností. Lze ji rozšít v prostoru o další funkce jako je rozpoznání obličeje, která uživatelům pomůže rozpoznat lidi podle jejich obličejů. Aplikace lze také využít v jiných oblastech, jako je školství či zaměstnání.

1.5.4 Be My Eyes

Be My Eyes je mobilní aplikace, která prostřednictvím videohovorů propojuje lidi se zrakovým handicapem s dobrovolníky po celém světě. Tato aplikace byla vyvinuta Hansem Jørgenem Wibergem roku 2015. Jedním z důvodů jejího vývoje je to, že sám zakladatel je zrakově postižený. Cílem platformy je zlepšit kvalitu života zrakově handicapovaným osobám tím způsobem, že jim poskytne přístup k popisu okolí pomocí dobrovolníků.

Funkce aplikace Be My Eyes jsou jednoduché a přátelské. Aplikace je založena na principu, že zrakově postižení lidé mohou kdykoliv požádat o pomoc s otázkami, kde je potřeba zrakové vnímání. Pak dobrovolníci dostanou upozornění, které jim říká, že někdo potřebuje pomoc. Jakmile pomoc přijme, tak může zrakově postiženému uživateli pomoc k popsání barvy, přečtení textu nebo v orientaci v prostoru.

Dopad této aplikace na společnost je kladný, protože poskytuje postiženým lidem sebedůvěru a nezávislost ve vykonání denních prací či úkolů. Aplikace jim umožňuje překonat bariéru, která by jim jinak omezila účastnit se ve společnosti. Pokud to vezmeme z druhé strany, tak dobrovolníkům pomáhá příležitost se zapojit k pomoci do života ostatních a rozvíjí jejich porozumění zrakově postiženým lidem.

Aplikace má další příležitost pro rozvoj, kde s rostoucím počtem uživatelů a dobrovolníků otevírá nové možnosti pro integraci dalších technologiích a služeb. V budoucnu by se do aplikace mohla přidat AI, které by mohlo být použito k rozpoznání textu, barev a dalších prvků. To by snížilo náročnost pro dobrovolníky a zlepšily rychlost a efektivitu poskytování pomoci.

1.5.5 Braillovo řádek

Braillovo řádek nebo-li hmatový displej, byl vyvinut roku 1970. Jedná se o zařízení, které pomáhá zrakově postiženým lidem číst text přenesený z elektronické podoby. Zařízení se skládá z řádku, na kterém jsou vytyčeny řady pinů, které mohou být hmatně prstem cítěné. Pracuje na principu toho, že si převezme zobrazený text na počítači či mobilu a přenese ho na tento řádek, který ho pak přemění na Braillovo písmo, jenž umožňuje uživateli číst daný text.

Braillovo řádek funguje tak, že text poslaný z počítače je přeložen do Braillova písma. Zobrazování textu je prováděno přes piny. Každý sloupec se skládá ze šesti pinů, které mohou být kdykoliv sklopeny nebo zvednuty. Tato funkce umožňuje zobrazení textu v Braillovu písmu. Braillovo řádky se od sebe mohou lišit počtem charakterů, které se přes ně dají zobrazit v jednu dobu.

Braillovo řádek je velmi užitečný pro lidi se zrakovou vadou. Jednou z výhod tohoto řádku je, že jim pomáhá číst elektronickou formu a tudíž se nemusí spoléhat na učebnice či dokumenty napsané v Braillovo písmu. Další výhodou je, že se jedná o poměrně úsporné zařízení, což se týká prostoru a tím může být brán s sebou na výlety a také ho lze využít současně s tabletem či telefonem. Poslední výhodou je to, že si člověk může samostatně rozšiřovat piny podle jeho libosti.

Braillův řádek má i nevýhody. Jednou z nevýhod je to, že člověk využívající tento řádek musí mít nějaké znalosti jak ho nastavit, což může být pro někoho velmi těžké, pokud třeba mluvíme o starších lidech. Druhá nevýhoda Braillovo řádku je jeho cena. Poslední nevýhodou je, že Braillův řádek nemůže podat detailní zprávu nebo kontext, který by byl poskytnut přes knihu či dokument. [10]

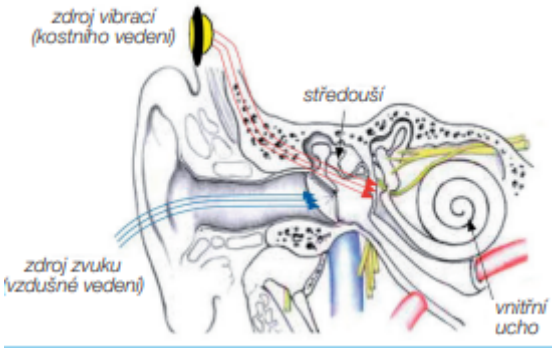
1.5.6 Sluchadla kotvená do kosti

Sluchadla kotvená do kosti jsou a byla revolučním přístupem k osobám se sluchovým postižením. Jsou určeny pro osoby, kterým už normální sluchadla nepomáhají. Byla vyvinuta roku 1970. Princip spočívá v tom, že sluchadla obejdou venkovní část ucha a posílají zvuk přímo do vnitřku ucha a dávají zvuku novou cestu k projetí.

Sluchadla jsou složena ze tří typů: implantát, abutment a zvukový procesor. Fungují tak, že implantát je pomocí operace umístěn do kosti hlavy za ucho, poté je abutment dán směrem ven, aby se mohl k němu přidávat zvukový procesor, který se dá odendat. Procesor přijímá zvukové zprávy z okolí a posílá je do implantátu, kde za pomocí vibrací je zvuk posílán do vnitřního ucha.

Sluchadla kotvená do kosti jsou velmi důležitá pro osoby trpící silnou sluchovou vadou. Pokud bychom vzali jeho výhody, tak jedna z nich je ta, že může být využíváno i lidmi s částečnou nebo silnou sluchovou vadou. Jeho druhá výhoda je to, že na rozdíl od normálních sluchadel má tento typ menší náchylnost na okolní zvuk a poslední výhodou je to, že tento typ sluchadel je méně viditelný.

Sluchadla kotvená do kosti mají také jisté nevýhody. Jednou z těchto nevýhod je, že člověk musí na operaci, aby mohl získat tento typ sluchadel, což může vést k určitým rizikům. Jsou nevhodná pro osoby, které mají nějaký hlavový úraz nebo jejich kosti kolem hlavy jsou deformované. Poslední nevýhodou těchto sluchadel je to, že pro osoby které mají už vysoce poškozený sluch, nemusí přinést zlepšení. [11]



Obrázek 3: Schéma vzdušného a kostního vedení zvuku do vnitřního ucha [12]

1.6 Tvorba mobilních aplikací a jejich platformy

S rozšířením technologie mobilních telefonů a tabletů roste také jejich využití v životě. Toto způsobuje, že začíná být větší důraz na vytváření aplikací pro tyto technologie. Mobilní aplikace jsou softwarové programy, které jsou vytvořené proto, aby se člověk s nimi co nejvíc bavil jako například mobilní hry nebo získal přes ně určité informace, jako například počasí. Proto se vývoj těchto mobilních aplikací stal důležitým aspektem v technice.

Pro tvorbu mobilních aplikací je důležité mít znalost programovacího jazyka. Mobilní aplikaci

lze programovat přes Javu, Kotlin, Objective-C nebo Swift. Z hlediska softwaru se mobilní aplikace vytváří pro Android a iOS, kde každý má svůj vlastní nástroj pro tvorbu aplikací na něj. Programátoři také musí vzít v potaz to, že ne všechny jazyky jsou plně využitelné pro určité systémy. Aplikace se musí přizpůsobit různým zařízením o jiných délkách obrazovky.

Pro tvoření mobilních aplikací existuje několik variant, ale zejména jsou děleny do dvou táborů, kde do jednoho táboru patří platformy, které jsou přímo určené na programovaní pro daný operační systém. Jejich výhodou je to, že programátor může využít všechno co mu daný operační systém nabízí. Nevýhodou je to, že se nemůže využít na dalších systémech. Poté je tu druhá skupina, kde se nacházejí platformy, které se dají využívat pro více operačních systémů. Jejich výhodou je, že se dají využít na více operačních systémech, ale jejich nevýhodou je, že nemusí být dobře optimalizované.

1.7 Druhy platforem pro vytváření mobilních aplikací

V dnešní době jsou mobilní aplikace důležitou součástí trhu. Existuje několik platforem přes které lze vytvořit mobilní aplikace, ale každá z nich má své výhody i nevýhody. Jeden z těchto parametrů je výběr platformy na tom, pro jaký operační software je aplikace tvořena. Platformy můžeme rozdělit na přímé a duální.

1.7.1 Android Studio

Android Studio je integrované vývojové prostředí (IDE), které je dělané pro vyvíjení přesné aplikace na Android. Tato platforma byla vyvinuta společností Google. Nabízí vývojářům komplexní a efektivní systém, který obsahuje několik nástrojů pro tvorbu úspěšných aplikací pro jejich zařízení. Toto studio je jedno z nejpoužívanějších z ohledu vývoje aplikací mířené přímo na operační systém Android.

Android studio může zaujmout vývojáře tím, že je přesně dělané pro Android zařízení a také že podporuje dva programovací jazyky. Jeden z těchto jazyků je Java a druhý je Kotlin. Oba dva tyto jazyky se používají k vývoji pro Android, kde programovací jazyk Kotlin lze využít i pro iOS systém. Toto studio obsahuje integrovaný editor, který zahrnuje dokončování kódů nebo pomáhá vývojářům psát čistější kódy. Obsahuje identifikátor problému ve výkonu nebo ladění.

Architektura Android studia je postavena na prostorovém editoru, který umožňuje uživatelům navrhovat a předvádět rozhraní aplikace. Editor poskytuje funkci drag-and-drop, což je funkce pro umísťování komponentů a vývojář může vidět, jak tyto komponenty vypadají v jeho aplikaci. Navíc toto studio zahrnuje šablony, přes které lze aplikace udělat přístupnější. Další výhodou tohoto studia je to, že podporuje testování zařízení, kde si člověk může přes své zařízení otestovat, jaký bude mít jeho aplikace vzhled a jestli fungují všechny komponenty. Android studio také nabízí emulátor, což je testovací sada přes kterou si může člověk zobrazit svoji aplikaci na jakémkoli zařízení. [13]

1.7.2 Xcode

Xcode je integrované vývojové prostředí (IDE), které bylo vyvinuto společností Apple pro to, aby mohli vývojáři vyvíjet aplikace přesně na jejich zařízení a systém, jako například macOS, iOS, watchOS a tvOS. Tato platforma poskytuje vývojářům komplexní a funkční systém s několika nástroji pro vývoj úspěšných aplikací pro jejich zařízení. Toto studio je jedno z nejpoužívanějších z ohledu vývoje aplikací mířené přímo na operační systém iOS.

Xcode může vývojáře zaujmout tím, že je přesně dělaný pro Apple zařízení s možností použití několika programovacích jazyků, například Swift, Objactive-C a C++. Xcode také obsahuje editor, který umožňuje uživatelům pomoc při dokončování kódu nebo ověřování chyb a syntaxe. Toto jim tedy může pomoci v pozdější fázi vývoje, jelikož budou mít méně práce v upravování kódů.

Architektura Xcode je postavena tak, že umožňuje vizuálně navrhovat a předvádět uživatelské rozhraní své aplikace. Obsahuje editor, přes který si vývojář může stavět svůj vzhled aplikace a tak i vidět, jak se s postupem času mění. Navíc také zahrnuje vysokou škálu nástrojů, přes které může člověk vytvářet přívětivé aplikace. Další významnou zajímavostí této platformy je možnost si testovat své aplikace ve virtuální rozhraní pro různé typy zařízení a tím vidět, jak aplikace reaguje na dané zařízení z pohledu rozhraní a funkčnosti komponentů. [14]

1.7.3 React Native

React Native je duální platforma, ze které může vývojář programovat pro systémy Android a iOS. Tato platforma byla vyvinuta společností Facebook. React Native je platforma, která je tvořena na základě React, což je JavaScript knihovna pro tvoření. React Native je dobrá v tom, že nám stačí jen jeden kód, který lze využít na více platforem.

React Native může vývojáře zaujmout tím, že kód jde psát v JavaScriptu, takže člověk který je znalý ohledně takového jazyka se může velmi dobře zorientovat v této platformě a tím pádem i rychle navrhovat aplikace pro Androi a iOS. Tento systém používá různé množství komponentů od tlačítka po textview, kde všechno toto může být využito pro zpracování aplikace, která je dobře poskládaná a dobrá i z hlediska vzhledu. Další zajímavou věcí na React Native je to, že má ve svých knihovnách zabudované komponenty pro každou platformu. Takže si může vzít výhody ostatních platforem a využít je pro sebe. React Native také nabízí, že vývojář kdykoliv může vidět, jaký má aplikace vzhled v reálném čase, a také tuto aplikaci otestovat. [15]

1.7.4 Flutter

Flutter je bezplatná duální platforma, která byla vyvinuta Googlem. Tato platforma je užívána pro programování aplikací pro Android i iOS. Je s ní možno naprogramovat webové stránky nebo počítačové aplikace. Tato platforma je velmi používaná z toho důvodu, že má velkou kompatibilitu a že vývojář může pro vytvoření aplikace použít pouze jednu databázi.

Flutter může vývojáře aplikací zaujmout tím, že má v sobě nastavené pomůcky (widgety), které jsou už předem definované, ale člověk si je může bez problémů sám nastavit od barvy a styl písma. V této platformě je dána funkce, která ukazuje co se děje v reálném čase, takže ukazuje změny nebo přidané widgety, což je jednodušší pro náhled.

Architektura Fletteru je pak postavena na programovacím jazyku Dart. Je to programovací jazyk, který byl vyvinut Googlem. Tento programovací jazyk je objektově orientační a je kompilován do strojového jazyka, což zajišťuje jeho rychlejší výkon. Další výhodou tohoto systému je, že přes něj se mohou vytvořit nativní aplikace jak pro iOS tak i pro Android. Umožňuje vývojářům používat kameru nebo senzory. Flutter také nabízí i jiné aplikace uvnitř této platformy jako jsou Flutter Inspector, která umožňuje prohlížení struktury a chování aplikace. Další je Flutter DevTools, který poskytuje informace o výkonu aplikace v této platformě vytvořené. [16]

2 Praktická část

2.1 Cíl tvoření aplikace pro hluchoslepé

Cíl praktické části je postavit a naprogramovat mobilní aplikaci, která bude přístupná pro tablety a mobily. Tato aplikace bude vyvíjena pro hluchoslepé lidi. Rozeberu tedy, v čem jsem vytvořil mobilní aplikaci a jak funguje. Potom bych rozebral, jakou formu komunikace mezi zařízeními používám a jaké se další dají použít (tvz. bluetooth, wifi) pro přenos informací z tabletu na hardwarová zařízení nebo na další mobilní zařízení. Praktická část se bude skládat z popsání platformy, ve které jsem zadanou aplikaci vytvářel. Poté bude následovat popis aplikace, kterou jsem vytvořil pro rodinu hluchoslepé osoby a budou v ní popsány i způsoby komunikace. Jako poslední bude popsána aplikace pro hluchoslepé lidi.

2.2 Tvorba aplikace v Android Studiu

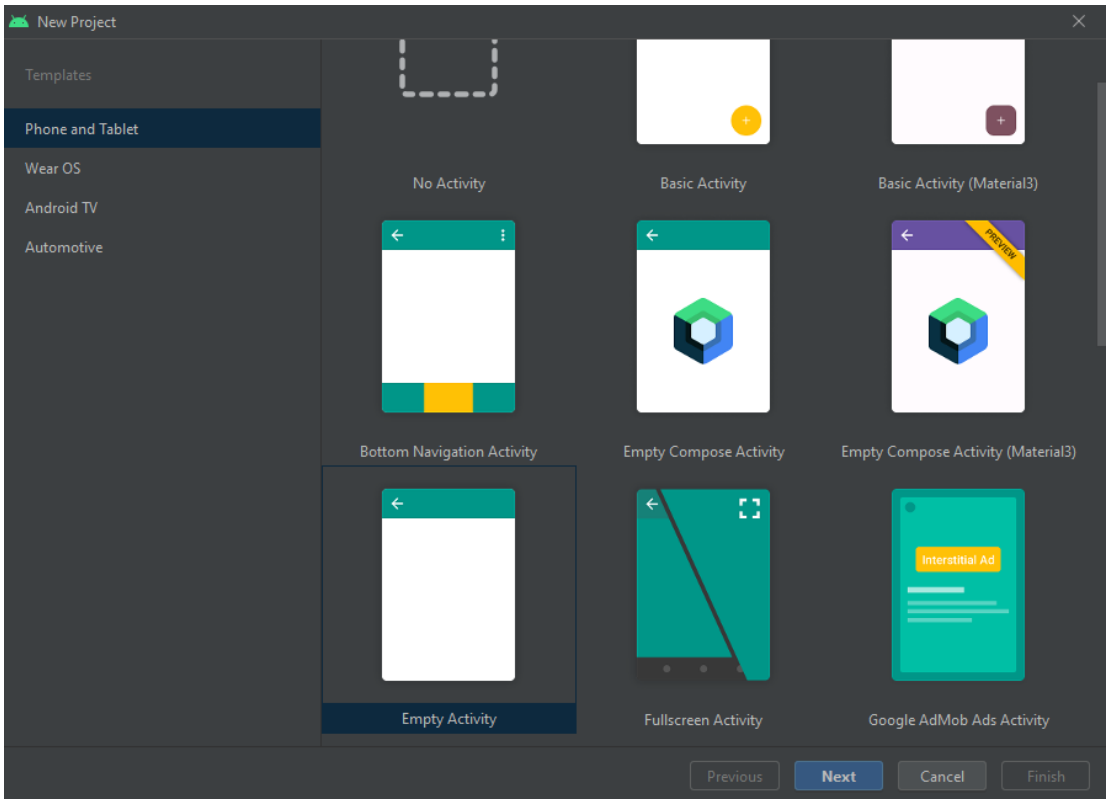
Pro vytvoření mobilní aplikace jsem si vybral Android studio. Jeden z důvodů pro výběr tohoto studia je to, že je velmi přívětivý pro začátečníky. Jeden z dalších důvodů, proč jsem si toto studio vybral je to, že mi umožňuje vyzkoušení mého programu na mobilním zařízení, které mám. Studio má také možnost přepsání kódu do programovacího jazyka Kotlin, jenž je poté přístupný pro operační systém iOS nebo použít už daný java kód, který by se pak musel zkompilovat do nativní binární iOS.

V Android studiu budu vytvářet dvě mobilní aplikace. Jedna z nich bude tvořena pro pracovní personál či rodinu a známé a ta druhá pro hluchoslepé lidi. Mobilní aplikace bude tvořena pro personál a rodinu z toho důvodu, aby jim mohli posílat zprávy a pak vidět, jestli si odeslanou zprávu přečetli. Druhá aplikace bude tvořena pro hluchoslepé lidi, zejména pro ty ve starším věku. Aplikace bude obsahovat přívětivé zobrazení. Bude fungovat na principu toho, že při příchodu textu jim zavrní tablet, aby byli upozorněni na příchozí text. Po příchodu se text stane klikacím, kdy po dotyku či přejezdu po písmenu mi odešle to dané písmeno zpět do první aplikace. Na každé odeslané písmeno je navázáno vrnění a změny barvy, aby hluchoslepý člověk věděl, že přejel přes písmeno. Tato aplikace bude mít také možnost úpravy textu a to jak jeho velikostí, tak i mezerou mezi písmeny. Přes úpravu textu si pak upravit text na text, který jim bude vyhovovat. Nastavení bude muset případně nastavit personál či rodina.

2.2.1 Vytvoření projektu

Jako v každém studiu se nejdříve musí vytvořit jeho projekt, ve kterém budete programovat. Jako první krok nás čeká to, že si musíme stáhnout Android studio ze stránek vývojářů, kde nám nabídnou nejvyspělejší verzi (teď je to Flamingo), ale také můžeme stáhnout i starší verze a programovat v nich (Electric Eel, Dolphin, Giraffe, Hedgehog). I Když jsou tyto verze starší, tak stále na ně vychází vylepšení. Sám programuji ve verzi Dolphin, jelikož v době, kdy jsem začal, byla tato verze nejmodernější a mně se nechtělo přecházet na další vyspělejší verze z důvodu toho, že by tam byla možnost porušit fungování kódu.

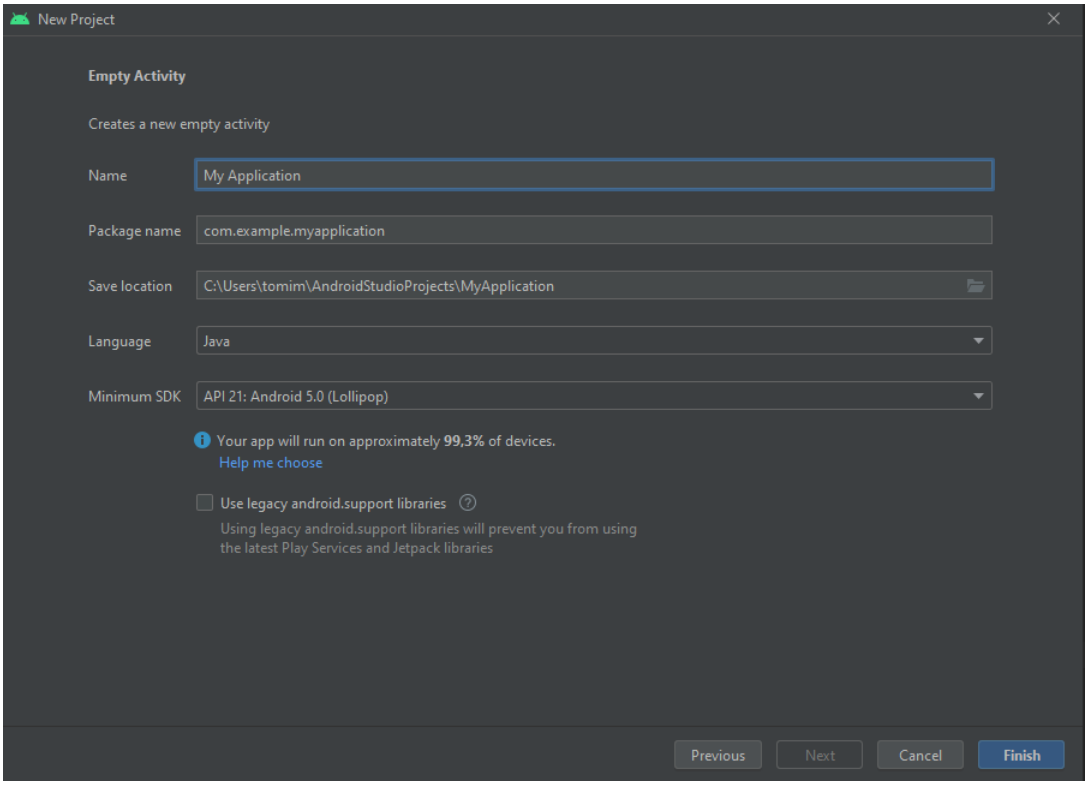
První věcí po stažení verze, ve které budete chtít pracovat, je otevření programového balíčku a projití si instalací, kde nemusíte nic nakonfigurovat či nastavovat. Po instalaci se vám objeví nabídka pro vytvoření projektu a v jaké aktivitě to chcete udělat a také pro jaké zařízení tato aplikace bude využívána. Máme na výběr ze čtyř segmentů. Jsou to mobily a tablety, hodinky, televize a nebo aplikace do automobilu. Každá z těchto položek má svůj vlastní list aktivit, které mohou být použity v závislosti na typu aplikace, kterou vytváříte. V segmentu mobily a tablety si můžete vybrat Google mapy, Google placení a nebo aktivitu pro přihlášení.



Obrázek 4: Nabídka pro vytvoření programu

Po výběru segmentu a aktivity přichází na čas pojmenování vašeho projektu a určení místa, kam se budou ukládat. Následně nastavit v jakém programovacím jazyce budete programovat, což je na výběr Java a Kotlin. Dále přichází na řadu nastavení pro jaký minimální SDK, což nejvíc týká minimálního API co bude používáno (maximální API je teď 33 minimální si můžeme nastavit). Na základě toho se určí nastavení SDK, přičemž budete vědět na kolika zařízeních bude vaše aplikace fungovat. Potom tam bude nabídka pro použití legacy android.support knihoven, ale od roku 2018 se už moc nepoužívají, jelikož Google přišel s novějšími knihovny zvané AndroidX.

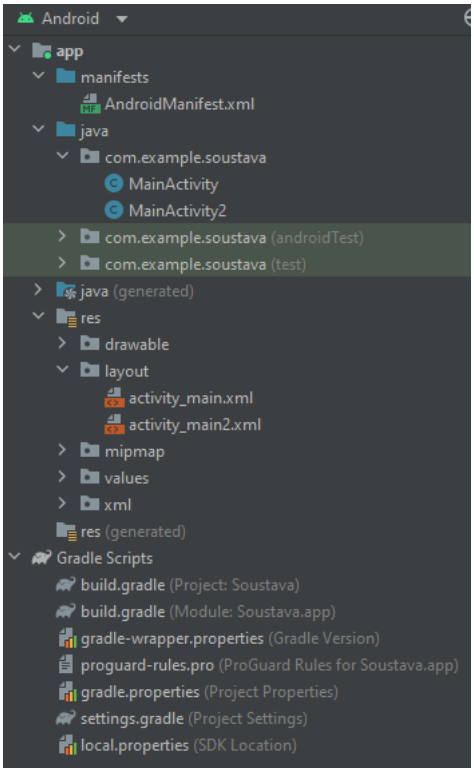
- **SDK** - což znamená Software Development Kit, je balíček nástrojů, knihoven, dokumentů a ukázkových kódů, který vývojářům usnadňuje vytváření aplikací pro konkrétní platformu nebo systém. SDK zjednodušuje proces vývoje tím, že poskytuje všechno potřebné pro práci s určitým operačním systémem, hardwarem nebo softwarovým rozhraním.
- **API** - což je zkratka pro Application Programming Interface, je kolekce pravidel, postupů a nástrojů, které usnadňují komunikaci a spolupráci mezi různými částmi softwaru. Jednoduše řečeno, API je jako spojovací most mezi dvěma aplikacemi nebo službami, který umožňuje výměnu informací a služeb mezi nimi.
- **Legacy android.support** knihovna - jsou to staré knihovny pro podporu Androidu, jejichž soubory se používaly při tvorbě aplikací pro Android. Tyto knihovny umožňovaly kompatibilitu se staršími verzemi a nabízeli různé nástroje a součásti, aby vývojáři mohli vytvářet aplikace fungující na různých verzích Androidu.



Obrázek 5: Nabídka pro přesnější určení aktivity

2.2.2 Projektové soubory

Po vytvoření projektu se dostáváme ke kódovací části, která obsahuje hlavní aktivitu pro jazyk Java či Kotlin, podle toho co si určíme. Dále nás čeká .xml soubor, který definuje, jak bude aplikace vypadat. V tomto souboru se programuje pomocí drop-in funkce, kterou je pak možné kódem upravit. Při načtení projektu se nám zprvu zobrazí pouze android soubory, což je udělané, proto aby se v těch souborech člověk lépe vyznal, jsou rozděleny. Také je možnost si přepnout do aktuálního projektu, kde uvidíme jak jsou uloženy a rozprostřeny všechny položky včetně těch, které nejsou vidět v souboru Android.



Obrázek 6: Soubory Androidu

Nyní Vám vysvětlím, co všechno určité složky znamenají, co se za nimi schovává a jak fungují. Když se podíváme na obrázek 6, tak Soubory Android můžeme dělit na dvě sekce a to app a Gradle Scripts. App pak můžeme dělit na tři skupiny, jež jsou manifest, java a res. Následně si podrobněji vysvětlíme, co znamenají jednotlivé složky v rámci skupiny app.

– **Manifest** - jedná se o složku která obsahuje AndroidManifest.xml, která určuje, jak se bude aplikace chovat. Tedy nám bude určovat konfiguraci systému aplikace. Tato složka udává, jak se aplikace jmenuje, její ikony, povolení jako například povolení pro použití wifi, či bluetooth. Také se používá pro spouštění aplikace, kdy systém Android z tohoto souboru čerpá potřebné informace.

- **java** - jedná se o složku, ve které se nachází všechno co tvoří základ aplikace. Je rozdělená na různé funkce podle použití. Jedna s nejpoužívatelnějších funkcí je MainActivity (činnosti), kde se definuje, jak se program chová a co všechno má dělat. Z tohoto programu ovládáme třeba chování tlačítek.
- **res** - jedná se o složku, ve které jsou uložené ostatní (externí) zdroje, které používá aplikace. Bude se tedy jednat o obrázky, styly, animace či layouty.
  - **drawable** - jedná se o podsložku kam ukládáme různé obrázky jak externí či interní, a pak je pomocí kódu používáme jinde. Také obsahuje už i dané obrázky a grafické zdroje.
  - **layout** - jedná se o podsoubor typu .xml, kde definujeme, jak bude naše aplikace vypadat pomocí reálného zobrazení vzhledu, kam si můžeme dávat různé komponenty, jako tlačítko, které se nám pak zapíše do .xml souboru, kde ho pak můžeme zlepšovat.
  - **mipmap** - tato podsložka obsahuje různé velikosti ikony aplikace.
  - **values** - tato podsložka má .xml soubory, které se používají pro hodnoty, jako barvy, či řetězce.
  - **xml** - tato podsložka slouží k ukládání souborů, které nikam jinam nepatří.

Následně vám vysvětlím, co představují různé položky v sekci Gradle Scripts. Gradle je nástroj klíčové pro sestavování, testování a nasazení aplikace.

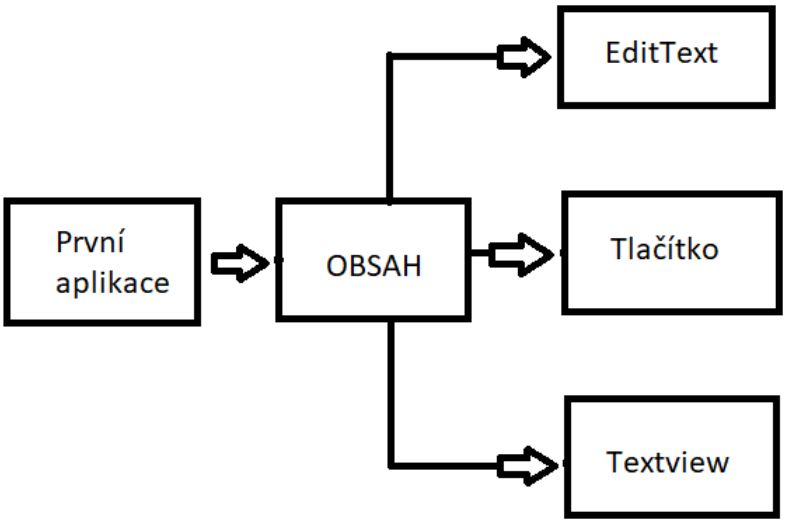
- **build.gradle** (Projekt) - tento soubor obsahuje celou konfiguraci projektu. Zde se zadává, jaký plugins se používá a také se sem dávají případně přípojky na externí síť.
- **build.gradle** (Modul) - tento soubor obsahuje konfiguraci na modulu. Zde se definují různé knihovny a nebo SKD, kde můžeme nastavit jeho minimální a i maximální velikost. Udává mi to cílovou verzi Androidu a další věci pro modul.
- **setting.gradle** - tento soubor obsahuje informace o modulech. Pro lehký projekt obsahuje jen pár odkazů, při těžkých pak pro všechny.
- **gradle.properties** - tento soubor obsahuje globální konfiguraci Gradle, jde zde nastavit alokace paměti.
- **local.properties** - tento soubor obsahuje lokální nastavení přístroje, je třeba zde umístění SDK.

2.3 První aplikace - Aplikace pro personál

První aplikace je tvořena pro rodinu, kamarády nebo pečovatelský personál, kdy tato aplikace je proto, aby mohli s hluchoslepým člověkem komunikovat a posílat text do druhé aplikace a přitom ještě dostávat informace o tom, co si z daného textu přečetli nebo jak se jim daří pochopit a přečíst daná slova či písmena. Na komunikace se bude používat připojení k internetu a data budou odeslána přes cloudovou službu Firebase. Tuto metodu jsem zvolil s ohledem na budoucnost, kdy mohou rodinní příslušníci, kamarádi nebo pečovatelé komunikovat s hluchoslepými lidmi, i když nejsou fyzicky na stejném místě.

2.3.1 Projektový plánek

Tato aplikace obsahuje EditText, který je pro zadání textu či zprávy, kterou budeme chtít odeslat. Tlačítko nám slouží k tomu, aby to odeslalo přes wifi na správné zařízení a pod tlačítkem bude Textview, přes který se bude zobrazovat přečtený text.



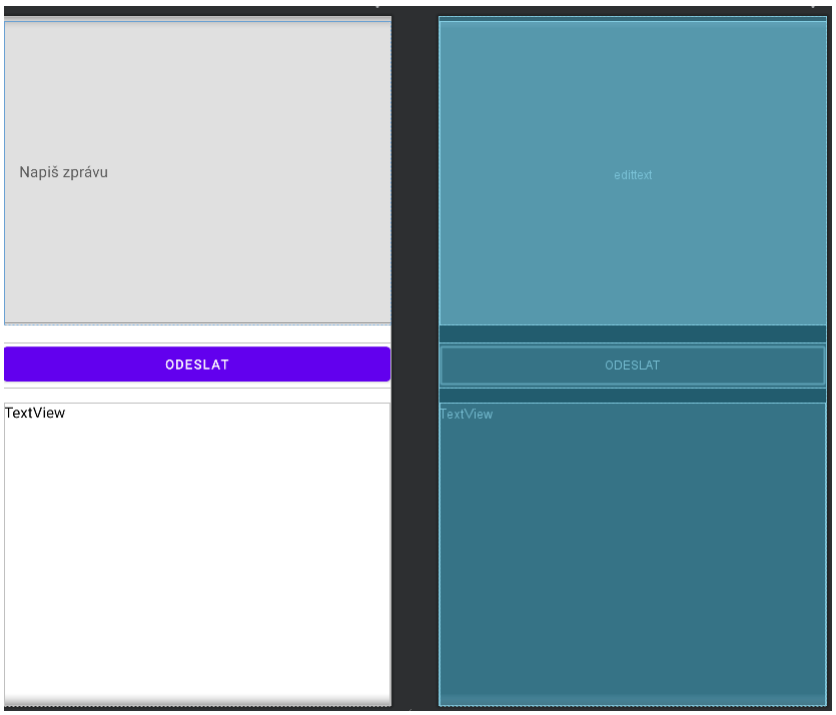
Obrázek 7: Zobrazení struktury

2.3.2 Zobrazení

V této sekci ukážu, jak jsem řešil vzhled této aplikace (viz. příloha č. 6). Celé zobrazení mé aplikace je dáno do LinearLayout, což je flexibilní způsob, jak řadit prvky aplikace vedle sebe (horizontálně) nebo za sebou (vertikálně). Díky tomuto nastavení je aplikace přizpůsobivá pro různé velikosti obrazovek mobilních telefonů a tabletů. Zobrazení je dáno do vertikální polohy. Vše je rozmístěno tak, aby se při každém otevření aplikace prvek TextView a EditText byly rovnoměrně rozprostřeny po obrazovce.

```
1 android:layout_weight="1"
```

Jak je vidět na obr. 8, prvky se rozprostřou po celé obrazovce a rozdělí ji na polovinu. Při zapnutí aplikace na uživatele vyskočí zpráva „Napiš zprávu"v EditTextu. Po kliknutí na toto okénko zpráva zmizí a tak uživatel může začít psát zprávu a odeslat ji pomocí tlačítka "odeslat". Pod tímto tlačítkem se už jen nachází Textview, který ukazuje, co hluchoslepá osoba přečetla z odeslané zprávy.

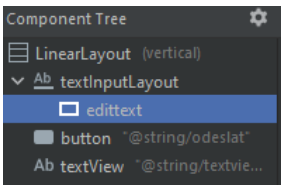


Obrázek 8: Zobrazení aplikace v Android studiu

Každý z komponentů má své unikátní pojmenování (id), aby mohl být snadno vyvolán v hlavním programu, kde je nastavena logika toho, co má daný prvek dělat. Nastavení barvy textu je nastavené na černou, pomocí kódu,

```
1  android:textColor="@color/black"
```

aby byl text dobře vidět. Text je také nastavený na výšku písma 16. Textview je pak nastavený na "multiline", což znamená, že když se text nevejde na jednu řádku a automaticky pokračuje na další. Tímto způsobem je zajištěna přehlednost a čitelnost textu bez ohledu na jeho délku.



Obrázek 9: Zobrazení uspořádání komponentů

2.3.3 Kód Java

V tomto oddílu budu rozebírat způsob, jakým jsem zapojil komponenty do jazyka Java (viz. příloha č.5). Začínám tím, že deklaruji komponenty EditText, TextView a tlačítko.

```
protected void onCreate(Bundle savedInstanceState) {
        TextView ZobrazeniTextu = findViewById(R.id.textView);
        Button tlacitko = findViewById(R.id.button);
        EditText VlozitText = findViewById(R.id.edittext);
} viz. priloha c. 5 - radky 25-27
```

Tyto komponenty poté inicializuji v hlavní třídě onCreate. Z této třídy se spouští většina kódu. Dále jsem nastavil tlačítku akci, která se vykoná po jeho stisknutí. Tuto akci jsem definoval tak, že se při kliknutí na tlačítko pošle text, který je napsán v EditText, do Firebase databáze.

```
Tlacitko.setOnClickListener(v -> {
Odeslat.setValue(text);} viz. priloha c. 5 - radky29-35
```

Pro posílání a zase příjímání textu používám Firebase, což mi umožňuje posílat data přes internet. Na Firebase databázi jsem napojen přes url adresu. Užívám url proto, že se databáze nachází v regionu Evropa a ne v USA, kde to není potřeba. Poté se odkazuji na tuto databázi pomocí proměnných Odeslat a Prijmout a pomocí těchto odkazů posílám text do jejich odpovídajících referencí.

```
FirebaseDatabase database = FirebaseDatabase.getInstance
        ("https://wifi-7efe9-default-rtdb.europe-west1.firebasedatabase.app/");
        DatabaseReference Odeslat = database.getReference("prvni");
        DatabaseReference Prijmout = database.getReference("druhy");
viz. priloha c. 5 - radky 21-23
```

Proměnná Odeslat je používaná pro odesílání textu do reference "první", odkud je pak brána druhou aplikací. Pak proměnná Prijmout je spojená s referencí "druha"odkud beru text z přečtených písmen, které mi posílá druhá aplikace. Z toho je pak vidět, jak je na tom daný hluchoslepý člověk a jestli už se může poslat další zpráva. Text, který dostávám od druhé aplikace, se zobrazuje v TextView. Funkce Firebase je taková, že pokud se v databázi změní hodnota textu, pošle se notifikace, že je tam nová hodnota. Tato hodnota se poté vyzvedne pomocí kódu a zobrazí se v TextView.

```
 Prijmout.addValueEventListener(new ValueEventListener() {
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                String text = dataSnapshot.getValue(String.class);
                ZobrazeniTextu.append(text);} viz. priloha c. 5 - radky 37-45
```

2.4 Komunikace mezi zařízeními

V dnešní době je komunikace mezi zařízeními důležitá pro denní život, protože si přes ně vyměňujeme informace, které pomáhají v efektivitě dnešní práce či dozvídáme se nějaké nové zprávy. Sdílet data jde pomocí Wifi, Bluetooth či cloudové služby jako Firebase. Při používání těchto funkcí je také důležité porozumět jejich silným a slabým stránkám.

2.4.1 Wifi

Wi-Fi je bezesporu technologie, kterou všichni znají. Jedná se o bezdrátovou technologii umožňující připojení zařízení k internetu nebo komunikaci s jinými zařízeními. Wi-fi je běžně používána v domácnostech, kancelářích nebo ve veřejných prostorách. Tato síť je vysoce využívána telefony, tablety či počítači pro připojení k internetu na to, aby mohli sdílet data, či média nebo prostě komunikovat s dalšími lidmi.

Vývojáři mohou použít Wifi jako prostředek komunikace mezi zařízeními, pomocí kódu Wifi Direct, což umožňuje naprogramování propojení mezi zařízeními bez středního bodu. Ta se dá vytvořit pomocí P2P (Peer to Peer) a je užívána například pro rychlý přesun informací, nebo hraní her mezi zařízeními. Pro to, aby Wifi Direct fungoval, musí mít aplikace zadané správné oprávnění. Také může využívat i různé třídy pro kontrolu nad tímto spojením, jako třeba WifiP2pManager nebo WifiP2pConfig.

2.4.2 Bluetooth

Bluetooth už je dnes méně používaná technologie, což se týče přenosu dat mezi mobilními telefony. Jedná se o přenášení dat či médií pomocí rádiových vln na krátké vzdálenosti. Tento typ přenosu je spíše lepší pro menší přenos dat a je ideální pro sdílení souborů mezi zařízeními jako například mezi chytrým telefonem a chytrýma hodinkami, či sluchátky. Bluetooth přenos je rovněž vhodný v situacích, kde není k dispozici internetové připojení.

Vývojáři tedy mohou využít spojení Bluetooth jako prostředek komunikace mezi různými zařízeními. Samozřejmě musí získat potřebná oprávnění pro využití této technologie v rámci své aplikace, následně pak mohou ovládat spojení pomocí různých tříd, jako například BluetoothAdapter, BluetoothServerSocket či BluetoothSocket. Tyto třídy pak umožní vyhledání dalšího zařízení nebo pomoc při spárování k zařízení a přenášení dat mezi nimi. Bluetooth je tedy vhodné pokud chcete přenášet data na krátké vzdálenosti.

2.4.3 Standard bezdrátové komunikační technologie (NFC)

NFC je bezdrátová komunikace na krátkou vzdálenost, která umožňuje přesnost informace mezi zařízeními, která jsou maximálně vzdálená 4m. Tato komunikace je běžně používána pro bezkontaktní platby či sdílení informací. Je to obzvlášť jednoduché při přenášení malých informací jako jsou například kontakty, fotografie nebo umožňuje i přenos nastavení zařízení. Co se týká bezkontaktních plateb. Ty většinou využívají Google Pay nebo Apple Pay pro zajištění bezpečného placení s těmito funkcemi.

Vývojáři na platformě Android studio mohou tuto funkci hned využít, jelikož je integrována do jejího prostředí a tím poskytuje vývojářům širší spektrum nástrojů. Pro práci s touto technologií je zase potřeba nastavit řádné oprávnění a pak je možnost toto ovládat pomocí různých tříd jako například NfcAdapter, NfcManager či PendingIntent. Tyto tři funkce se používají pro detekci, čtení a zápisu. Také pak pro bezpečnostní přenos mezi zařízeními, která tuto komunikaci využívají.

2.4.4 Firebase

Firebase je cloudová komplexní služba, jejímž majitelem je Google. Tato služba je integrovaná v Android studiu a tím poskytuje programátorům široké spektrum funkcí a nástrojů. I když sice tato technologie nemá přímé předávání informací mezi spárovaným zařízením jako Wifi či Bluetooth, tak tato platforma zahrnuje několik služeb pro vývoj, testování či nasazení aplikace. Obsahuje služby jako například Firebase Realtime Database, Firebase Cloud Messaging (FCM), Firebase Authentication a Firebase Storage.

Firebase Realtime Database je jedna z velmi důležitých služeb, která je stavěna na NoSQL databázi a umožňuje synchronizaci dat v reálném čase pro různá zařízení. Toto je používáno pro rychlé sdělování informací mezi různými aplikacemi, které potřebují tato data rychle a hlavně efektivně. Také v této databázi můžeme nastavit její pravidla, například to, že aplikace může čerpat data bez toho, aby se musel člověk do své aplikace přihlásit.

```
1 "rules": {
2       ".read": "auth.uid == null",
3       ".write": "auth.uid == null"
```

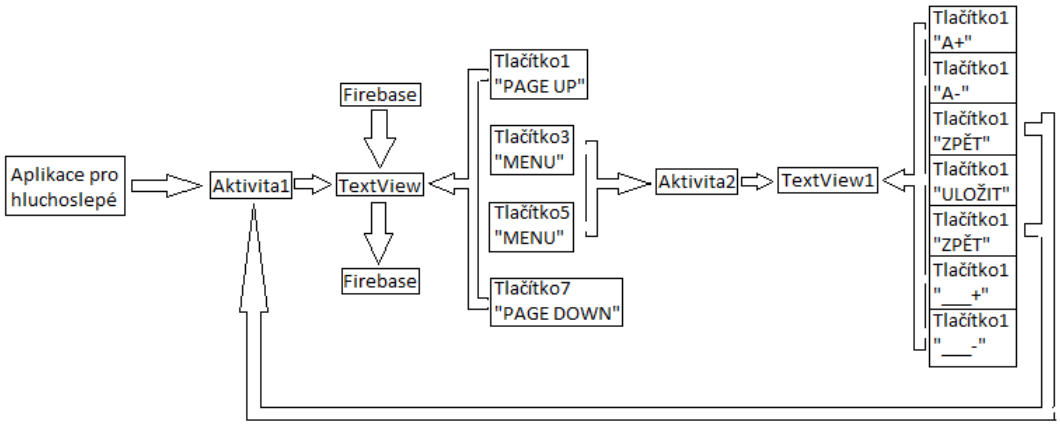
Firebase Cloud Messaging (FCM) je služba, která slouží k odesílání zpráv mezi platformami. Také umožňuje komunikaci mezi zařízeními bez omezení na spárování. Firebase Authentication je služba určená pro aplikace, které vyžadují přihlašovací funkci a synchronizaci dat mezi více zařízeními. Firebase Storage je služba využívaná, pokud chceme sdílet obrázky, média, či dokumenty. Firebase také obsahuje i jiné služby jako například Crashlytics, což je pro hlášení chyb a pádů aplikace.

2.5 Druhá aplikace

Druhá aplikace je navržena pro hluchoslepé lidi. Je tvořena tak, aby při příjmu textu z první aplikace mohli číst text slovo po slovu. Tato aplikace je tvořena na možnosti využívání funkce pro odesílání jednotlivých písmen na další spárované zařízení, které jim písmena převede do formátu, kterému budou rozumět. V aplikaci bude možnost si nastavit text do vyhovující výšky. Lze si tam zvětšovat či zmenšovat mezeru mezi písmeny. Aplikace také umožňuje, že každé zmáčknutí tlačítka je doprovázeno vibracemi, jejichž délku je možné později v kódu nastavit podle potřeby. Aplikace také obsahuje funkci, že po odeslání každého písmene je odesílateli zaslána zpětná vazba. A jakmile jsou písmena odeslána, změní se na jinou barvu.

2.5.1 Projektový plánek

Tato aplikace se skládá ze dvou aktivit, z nichž první aktivita slouží pro zobrazování textu a druhá aktivita slouží pro nastavení výšky textu a mezery mezi písmeny. V první aktivitě se nachází čtyři tlačítka a hlavní TextView. V první aktivitě se nachází tlačítko "PAGE UP", které slouží pro posouvání textu nahoru, pak dvě tlačítka "MENU", která slouží k přechodu do druhé aktivity a poté tlačítko "PAGE DOWN", jenž slouží pro posouvání textu dolů. Pak ve druhé aktivitě se nachází sedm tlačítek a TextView1. Kde tlačítka "A+"a "A+"slouží pro zvětšování a zmenšování textu, poté tlačítka "ZPĚT", jenž slouží pro vrácení do první aktivity s uloženými hodnotami a nebo bez nich. Mezi těmito tlačítky se pak nachází tlačítko "ULOŽIT", které slouží pro ukládání nastavených hodnot a jako poslední jsou tlačítka pro zmenšování a zvětšování mezery mezi písmeny. Všechny tyto změny jsou vidět v TextView1.



Obrázek 10: Zobrazení struktury Druhé aplikace

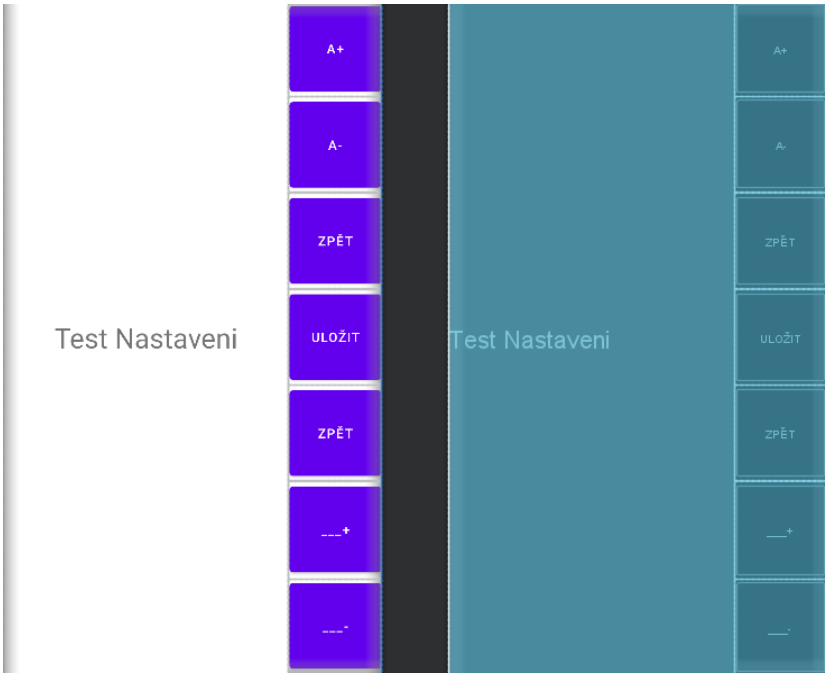
2.5.2 Zobrazení

V této sekci budeme probírat vzhled této aplikace (viz. příloha 8 a 10). Aplikace se skládá ze dvou aktivit. Celé zobrazení první aktivity je daná do LinearLayout (horizontální), aby se přizpůsobilo obrazovce. Poté do pravého rohu je dán další vektorový LinearLayout, který zařizuje

přesnost rozložení tlačítek a prázdných políček. Také můžeme zadat hodnotu, jak vysoká budou tato tlačítka. Fungování tlačítek je popsáno v podkapitole Projektový plánek.

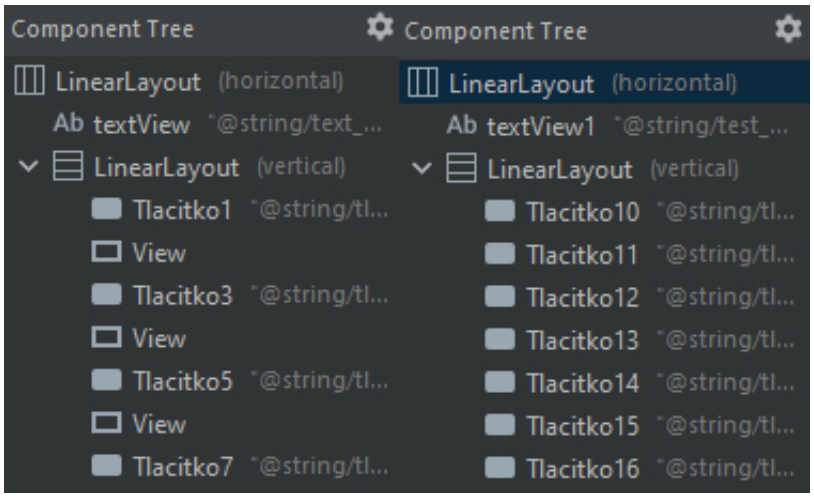
Obrázek 11: Zobrazení první aktivity druhé aplikace

Druhá část aplikace je založena na stejném rozložení a jediným rozdílem je, že místo prázdných polí se využívají další tlačítka. Text "Test Nastavení"je umístěn do středu. Tento text slouží pro testování nastavení a umožňuje uživateli vidět, zda mu vyhovuje nastavený text.



Obrázek 12: Zobrazení druhé aktivity druhé aplikace

Každý z komponentů má svoje unikátní ID, a tak se může vyvolat v programu. Jejich unikátní ID můžeme vidět na obrázku č. 13. Text je nastaven tak, aby se zobrazoval v černé barvě a výška tohoto textu je dána na 60 pixelů.



Obrázek 13: Zobrazení uspořádání komponentů Aktivity1 (levá) a Aktivity2 (pravá)

2.6 Programování druhé aplikace (Java)

V této sekci budeme zkoumat, jaké byly použity postupy a techniky pro začlenění dvou aktivit do jedné aplikace (viz. Přílohy č.7 a č.9). Budeme se zabývat funkcionalitou tlačítek, jejich různými aplikacemi a způsoby, s jakými mohou interagovat s ostatními komponenty aplikace. Důležitým prvkem této sekce bude také detekce dotyku, konkrétně jeho sledování. Toto nám umožní pohlédnout, jak může aplikace posílat data s písmeny na spárované zařízení v reálném čase. Tato sekce nám tedy poskytne široký přehled o tom, jak byly naprogramované komponenty aplikace.

2.6.1 První aktivita

Tento kód je napsán v Java a využívá Android SDK (viz. příloha č. 7). Je zaměřen na manipulaci s textem a jeho interakcí v Android aplikaci. Kód zahrnuje různé metody pro ukládání a načítání preferencí, detekci doteků a práci s textovými daty. Jako první v programu jsou deklarovány globální proměnné. Součástí těchto globálních proměnných jsou Tlacitko3(5)Zmacknuto, Souřadnicey, textView, Firebase (Odchozi a Prichozi), VyskaTextu až RozsahVsechRadku a jako poslední jsou UlozeniPismen a ResetovaniUlozeni. Zbytek proměnných je pak deklarovaný jen lokálně, což jsou jen tlačítka a textview.

První co je iniciované ve funkci **OnCreate** je to, že textView je udělán posuvným. Toto posunutí je pak zařízeno přes tlačítka 1 a 7. Tlačítko1 nebo-li "PAGE UP"je děláno pro posouvání textu nahoru a tlačítko7 nebo-li "PAGE DOWN"je děláno pro posouvání textu dolů. Obě dvě tyto tlačítka posouvají text o pět řádek.

```
textView.setScroller(new Scroller(this));
Tlacitko1.setOnClickListener(v -> {
  if (textView.getScrollY() > 0)
  textView.scrollTo(0, textView.getScrollY() - 5*textView.getLineHeight());
  }); viz. priloha c. 7 - radky 73, 90-93
```

Jako další jsou ve funkci **OnCreate** iniciována tlačítka 3 a 5 neboli "MENU". Tyto tlačítka slouží pro postup do druhé aktivity, kde na nás čeká nastavení textu. Obě dvě tato tlačítka musí být podržena určitou dobu, aby se proměnná Tlacitko(3)5Zmacknuto proměnila v pravdu, což by značilo zmáčknutí tohoto tlačítka. Tedy po zmáčknutí tlačítek jsou dané proměnné porovnávány v metodě **ZmacknutiObouchTlacitek**, a to zaručuje, že teprve pokud budou obě dvě tlačítka hlásat, že proměnné Tlacitko3Zmacknuto a Tlacitko5Zmacknuto jsou pravdivé, tak pošle do druhé aktivity stávající hodnoty nacházející se v textViewu a také tablet zavrní. Obě dvě tato tlačítka mají také nastaveno, že jejich proměnná zůstane pravdivá pouze po dobu dvou sekund.

```
Tlacitko3.setOnLongClickListener(v -> {
    Tlacitko3Zmacknuto = true;
    new Handler().postDelayed(() -> Tlacitko3Zmacknuto = false, 2000);
    ZmacknutiObouchTlacitek();
    return true;
    }); viz. priloha c. 7 - radky 95-100, 121-142
private void ZmacknutiObouchTlacitek() {
        if (Tlacitko3Zmacknuto && Tlacitko5Zmacknuto) {
        Intent Poslat = new Intent(this, MainActivity2.class);
        Poslat.putExtra(VyskaTextu, StavajiciVyskaTextu);
        Poslat.putExtra(RozsahMezery, StavajiciRozsahMezery);
        Poslat.putExtra(RozsahHorniRadky, topPadding);
        Poslat.putExtra(RozsahVsechRadku, LineSpace);
        NavazaniSpojeniSDruhouAktivitou.launch(Poslat); }
```

S metodou pro posílání hodnot do druhé aktivity je také spjato její přijímání. To je dělané pomocí metody **NavazaniSpojeniSDruhouAktivitou**. To mi zajišťuje přijímání dat z druhé aktivity přes funkci Prijmout, ze které jsou pak čerpána jednotlivá data pro nastavení textViewu.

```
ActivityResultLauncher<Intent> NavazaniSpojeniSDruhouAktivitou = ...
    registerForActivityResult(
            new ActivityResultContracts.StartActivityForResult(),
            result -> {
                if (result.getResultCode() == Activity.RESULT_OK) {
                    Intent Prijmout = result.getData();
                    if (Prijmout != null) { viz. priloha c. 7 - radky 144-168
```

Tato data jsou také uložena do metody **UlozitPromene** a slouží pro ukládání dat v případě odchodu z aplikace. Po znovuotevření této aplikace jsou pak tato data čerpána z metody **NacistPromene**, jenž znovu nastaví daný textView na hodnoty, se kterými se tato aplikace opouštěla.

```
1  private void UlozitPromene(float novaVyskaTextu, float noveRozmeziMezery, ...
       float novyRozmeziVsechRadku, int novyRozmeziHornihoRadku) {
2          SharedPreferences preferences = ...
               getSharedPreferences("UlozeneHodnoty", MODE_PRIVATE);
3   } viz. priloha c. 7 - radky 170-178
4
5  public void NacistPromene() {
6          SharedPreferences sharedPreferences = ...
               getSharedPreferences("MojeUlozeneHodnoty", MODE_PRIVATE);
7  } viz. priloha c. 7 - radky 180-195
```

Další součást funkce **OnCreate** je hledání horní osy y v textViewu. Jedná se o metodu **NacteniObrazovky**. Tato metoda má zaručit to, že teprve až po načtení všech prvků se bude teprve hledat, jaká je horní osa y. Tato osa je hledaná z toho důvodu, aby v metodě **onTouchEvent** byla osa y přizpůsobena z pohledu textViewu a ne celé obrazovky (je tam určitý rozdíl mezi hodnotami).

```
1  ViewTreeObserver NacteniObrazovky = textView.getViewTreeObserver();
2  NacteniObrazovky.addOnGlobalLayoutListener(new ...
       ViewTreeObserver.OnGlobalLayoutListener() {
3    @Override
4    public void onGlobalLayout() {
5      int[] lokace = new int[2];
6      textView.getLocationOnScreen(lokace);
7      int y = lokace[1];
8      Souradnicey = y;
9      textView.getViewTreeObserver().removeOnGlobalLayoutListener(this);
10             }}); viz. priloha c. 7 - radky 75-87
```

Toto nás tedy přivádí k metodě **onTouchEvent**. Tato metoda je založena na tom, že má za úkol sledovat přiložený prst ("ACTION DOWN") a také jeho pohyb po obrazovce ("ACTION MOVE"). Tato metoda mi pak vykazuje hodnoty osy x a y, které jsou pak posílány do metody **KontrolaPozicePismene**, jenž slouží pro porovnávání těchto hodnot s hodnotami písmen (osy x a y).

```
1  public boolean onTouchEvent(MotionEvent event) {
2    switch (event.getAction()) {
3      case MotionEvent.ACTION_DOWN:
4          float x = event.getX() - textView.getPaddingLeft();
5          float y = event.getY() - textView.getPaddingTop();
6           y = y - Souradnicey;
7          KontrolaPozicePismene(x, y);
8          break;
9          case MotionEvent.ACTION_MOVE:
10         stejny jako MotionEvent.ACTION_DOWN;
11         } return true; } viz. priloha c. 7 - radky 209-228
```

Toto všechno pak plyne k metodě **KontrolaPozicePismene**, jenž má za úkol kontrolovat, jestli se daná písmena neshodují s daným dotykem. První, co tato funkce udělá je export textu z textViewu, který je pak rozložen na jednotlivá písmena (Znak). Tyto písmena se vezmou a získají se z nich souřadnice os x a y, poté je z nich získána jejich tloušťka (TlouskaPisma) a šířka (VyskaPisma). Z těchto proměnných jsou pak získány osy středu písmene, jenž jsou pak využity pro výpočet vzdálenosti mezi středem písmene a souřadnicemi dotyku. Tuto vzdálenost pak používám pro to, abych zjistil, jestli byla vzdálenost písmene menší než jeho tloušťka. V tom případě se pak provede dotyk a písmeno je posláno do metody **PrejetiPresPismeno**, jenž je následně posláno do databáze. Součástí této akce je také to, že písmena poslaná do metody **PrejetiPresPismeno** jsou barevně obarvená, aby bylo vidět, že byla poslána a také se mi tady ta písmena ukládají do proměnné UlozeniPismen, jenž slouží proto, aby posílání do databáze nebylo zasyceno. Součástí podmínky if je také, že dává najevo, že se na daném místě se nachází písmeno. Jakmile by byla ale proměnná PismenoJeTam falešná, tak se přejde na podmínku, že tam dané písmeno není a hledá se, jestli se tam nenachází mezera pomocí metody **ZjisteniMezery**.

```
1  private void KontrolaPozicePismene(float x, float y) {
2          Layout rozlozeni = textView.getLayout();
3          String text = textView.getText().toString();
4          boolean PismenoJeTam = false;
5          for (int i = 0; i < text.length(); i++) {
6              char Znak = text.charAt(i);
7              int radek = rozlozeni.getLineForOffset(i);
8              float souradniceX = rozlozeni.getPrimaryHorizontal(i);
9              float souradniceY = rozlozeni.getLineBaseline(radek) - ...
                   textView.getScrollY();
10
11             Rect HranicePismene = new Rect();
12             textView.getPaint().getTextBounds(String.valueOf(Znak), 0, 1, ...
                   HranicePismene);
13             float TlouskaPisma = HranicePismene.width();
14             float VyskaPisma = HranicePismene.height();
15             float souradniceXprostedku = souradniceX + TlouskaPisma / 2;
16             float souradniceYprostedku = souradniceY - VyskaPisma / 2;
17             float vzdlenost = (float) Math.sqrt((souradniceXprostedku - x) ...
                   * (souradniceXprostedku - x) + (souradniceYprostedku - y) * ...
                   (souradniceYprostedku - y));
18
19             if(Znak == 'I' || Znak == 'i' || Znak == 'j' || Znak == 'J' || ...
                   Znak == 'l' || Znak == 't') {            }
20             if (vzdlenost < TlouskaPisma) {
21                 String OznacenePismeno = Znak + "_" + i;
22                 if (!UlozeniPismen.contains(OznacenePismeno)) {
23                     PrejetiPresPismeno(String.valueOf(Znak));
24                     UlozeniPismen.add(OznacenePismeno);
25                     ZmenaBarvyPismene(i);}
26                 PismenoJeTam = true;
27                 break;}}
28         if (!PismenoJeTam) {          }} viz. priloha c. 7 - radky 230-296
```

Metoda **ZjisteniMezery** je určená pro hledání mezery mezi písmeny. Začátek metody je založený na stejném principu jako předchozí metoda. Dále pak při vyskytnutí mezery mi určí jaké bude nadcházející písmeno. Poté následuje zjišťování, jestli je dotyk mezi mezerou a dalším písmem a také zda je rozdíl menší než 25. Jestli je tato podmínka splněna, tak mi to vyhazuje mezeru, ale pokud ne, tak to vyhazuje -1, což značí, že dotyk nebyl proveden na mezeře.

```
if (Znak == ' ') {
    float DalsiPismeno;
    if (i < text.length() - 1) {
      DalsiPismeno = rozlozeni.getPrimaryHorizontal(i + 1);
    } else {
      DalsiPismeno = textView.getWidth() - textView.getPaddingRight();
            }
    if (x ≥ souradniceX && x ≤ DalsiPismeno && Math.abs(y - souradniceY) < ...
        25) {
  return i;
  }
  }
}
return -1;      viz. priloha c. 7 - radky 298-322
```

Jednou z posledních metod této aktivity je **PrejetiPresPismeno**, jenž slouží pro posílání písmene s unikátním číslem do databáze Firebase, jenž je pak dále poslán do spárované aplikace.

```
private void PrejetiPresPismeno(String Znak) {
        long UnikatniCislo = System.currentTimeMillis();
        String OznacenyZnak = Znak + "_" + UnikatniCislo;
        if (UlozeniPismen.add(OznacenyZnak)) {
            Odchozi.setValue(OznacenyZnak);
        }
        ResetovaniUlozenychPismen();
    } viz. priloha c. 7 - radky 324-333
```

Jedna z hlavních metod je ještě **DostaniTextu**, jenž zařizuje dostání textu ze spárovaného zařízení a zobrazím v textViewu. V případě, že by text nebyl načten, tak dojde upozornilo na danou chybu.

```
private void DostaniTextu() {
        Prihozi.addValueEventListener(new ValueEventListener() {
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                String data = snapshot.getValue(String.class);
                if (data != null) {
                    textView.setText(data);}}
            public void onCancelled(@NonNull DatabaseError error) {
                Toast.makeText(MainActivity.this, "Chyba p i   ten   dat: " ...
                    + error.getMessage(), Toast.LENGTH_SHORT).show();}});}
viz. priloha c. 7 - radky 346-364
```

Jako poslední metody v této aktivitě jsou **ZmenaBarvyPismene** a **ResetovaniUlozenychPismen**. Metoda **ZmenaBarvyPismene** je založená na tom, aby mi měnila barvu po přejetém písmenu. A metoda **ResetovaniUlozenychPismen** zařizuje to, aby byla písmena znovu klikatelná po určitém uplynulém čase.

```
1 private void ResetovaniUlozenychPismen() {}
2 private void ZmenaBarvyPismene(int Znak) {}
3 viz. priloha c. 7 - radky 335-344
```

2.6.2 Druhá aktivita

Tento odstavec se zaměří na kód aplikace napsaný v Java, jenž představuje druhou aktivitu v aplikaci pro hluchoslepé (viz. příloze č. 9). Jeho hlavní funkcí je manipulace s textem a také posílání ho zpět do první aktivity. Pro lepší pochopení tohoto kódu rozebereme jeho klíčové aspekty jako jsou proměnné, funkce tlačítek, metody práce s textem a interakce mezi jednotlivými komponenty.

První, co v kódu najdeme je to, že v něm mám deklarovaných několik globálních proměnných. Jedna skupina proměnných se skládá z uložených hodnot (UlozitNovouVelikostTextu, atd.), při novém nastavení textu. Druhá skupina proměnných se skládá z obdržených hodnot (KurentniVelikostTextu, atd.), jenž naznačuje hodnoty, které byly obdrženy od první aktivity. Dále jsou definovány mezery řádků, jsou neměnné a slouží pouze pro uskutečnění příkazu změny horní mezery. Jako další je definovaný textView1 v němž je napsaný text "Text Nastavení". Dále jsou definovány proměnné pro určení, že byla zmáčknuta tlačítka. Potom když se podíváme do hlavní funkce **OnCreate**, tak můžeme vidět, že bylo lokálně deklarováno sedm tlačítek, která slouží pro manipulaci s textem. Dále jsou pomocí metody **Prijmout** získány proměnné z první aktivity. Po jejich získání jsou aplikovány na textView1.

```
1 textView1 = findViewById(R.id.textView1);
2 Button Tlacitko10 = findViewById(R.id.Tlacitko10);
3 Intent Prijmout = getIntent();
4 viz. priloha c. 9 - radky 47-54, 152-165
```

Jako první dvě tlačítka jsou tlačítko10 ("A+) a tlačítko11 ("A-). Tato dvě tlačítka mi zajišťují zvětšování a zmenšování textu s tím, že spodek textu zůstává na spodní části řádky a text se zvyšuje pouze do stran a nahoru s tím, že dolů se nehne. Toto je dosaženo pomocí podmínek NovaHorniRada a NoveRady, která slouží pro zvětšování mezer mezi řádky. Je to děláno tak, že jakmile se mi zvětší text, tak se mi zmenší mezery mezi řádky, a to má pak za efekt, že písmena zůstávají ve stejné ose y řádku. Toto je využito i pro zmenšování textu.

```
Tlacitko10.setOnClickListener(v -> {
            float VelikostKurentihoTextu = textView1.getTextSize();
            int HorniRada = textView1.getPaddingTop();
            float VsechnyRady = textView1.getLineSpacingExtra();
            int NovaHorniRada = HorniRada - 2;
            if (NovaHorniRada ≥ 40) {
                textView1.setPadding(LevaMezeraRadku, NovaHorniRada, ...
                    PravaMezeraRadku, DolniMezeraRadku);}
            float NoveRady = VsechnyRady - 2;
            if (NoveRady ≥ 40) {
                textView1.setLineSpacing(NoveRady, 1.0f);}
            float NovaVelikost = VelikostKurentihoTextu + 2;
            if (NovaVelikost ≤ 80) {
                textView1.setTextSize(TypedValue.COMPLEX_UNIT_PX, NovaVelikost);}
            VybraceTelefonu(v.getContext(), 500);
        }); viz. priloha c. 9 - radky 59-79
```

Další dvě důležité tlačítka jsou tlačítko15 a tlačítko16 , která slouží pro zvětšování a zmenšování mezery mezi písmeny. Je to dosažené tak, že při každém zmáčknutí tlačítka se mi zvětší či zmenší mezera mezi písmeny o jedno procento. Také tato tlačítka jsou vybavena vibrací při jejich stisknutí.

```
Tlacitko15.setOnClickListener(v -> {
    float RozsahKurentiMezery = textView1.getLetterSpacing();
    float NovyRozsah = RozsahKurentiMezery + 0.01f;
    if (NovyRozsah ≤ 0.2f) {
        textView1.setLetterSpacing(NovyRozsah);}
    VybraceTelefonu(v.getContext(), 500);}); viz. priloha c.9 - radky 131-149
```

Dále tu máme tlačítko pro uložení (tlačítko13), které slouží pro ukládání hodnot které byly nastaveny v textView1. Je to založeno na tom, že při dlouhém stisku tlačítka se objeví zpráva "Hodnoty uloženy"a ta značí, že proměnná Tlacitko13Zmacknuto je pravdivá. Značí to, že se do první aktivity pošlou uložené hodnoty pomocí metody OdeslatHodnotyNazpatek.

```
Tlacitko13.setOnLongClickListener(v -> {
            UlozitNovouVelikostTextu = textView1.getTextSize();
            ...
            Toast.makeText(MainActivity2.this, "Hodnoty ulo eny", ...
                Toast.LENGTH_SHORT).show();
            Tlacitko13Zmacknuto = true;
            return true;}); viz. priloha c. 9 - radky 112-122
```

Poslední dvě tlačítka ve funkci **OnCreate** jsou tlačítko12 ("Zpět") a tlačítko14 ("Zpět"). Tato dvě tlačítka mají za úkol dostat se do první aktivity po jejich zmáčknutí s proměnnými hodnotami, ze kterých se pak nastaví textView. Jejich funkce je obdobná jako pro tlačítka 3 a 5 v první aktivitě. Jejich stisknutím se vyvolá metoda **OdeslatHodnotyNazpatek**.

```
1 Tlacitko12.setOnLongClickListener(v -> {
2     Tlacitko12Zmacknuto = true;
3     new Handler().postDelayed(() -> Tlacitko12Zmacknuto = false, 2000);
4     OdeslatHodnotyNazptek();
5     return true;}); viz. priloha c. 9 - radky 104-109
```

Metoda **OdeslatHodnotyNazpatek** nám říká, jaká hodnota bude odeslána do první aktivity. Tato metoda má tedy na výběr ze dvou sad proměnných, kde jedna sada proměnných jsou Kurentni hodnoty, jenž obsahuje hodnoty, které byly poslány z první aktivity a to tady značí, že text bude neměnný (nebylo zmáčknuto tlačítko pro uložení). Druhá sada proměnných obsahuje Uložené hodnoty, které jsou vzaty z toho, že bylo zmáčknuto tlačítko pro uložení (tlačítko14).

```
1 private void OdeslatHodnotyNazptek() {
2         if (Tlacitko12Zmacknuto && Tlacitko14Zmacknuto) {
3             Intent Odeslat = new Intent();
4             if (Tlacitko13Zmacknuto) {
5                 Odeslat.putExtra(MainActivity.VyskaTextu, ...
                      UlozitNovouVelikostTextu);
6                 ...
7             } else {
8                 Odeslat.putExtra(MainActivity.VyskaTextu, KurentniVelikostTextu);
9                 ...
10            }
11            VybraceTelefonu(this, 500);
12            setResult(Activity.RESULT_OK, Odeslat);
13            finish();}} viz. priloha c. 9 - radky 179-199
```

2.7 Souhrn

V praktické části jsem popsal jak vytvořit projekt v Android studiu, co všechno v projektu jednotlivé složky znamenají a jaký mají účel. Popsal jsem první aplikaci, která je určena pro personál a druhou aplikaci určenou pro hluchoslepého člověka.

Pokud bych shrnul první aplikaci, tak bych řekl, že tato aplikace je funkční a kdykoliv použitelná a jsem s ní spokojen. Do budoucna by to ale chtělo vytvořit několik dalších funkcí. Jedna z těchto funkcí by byla přihlašovací stránka z důvodu využití pro více lidí. Současně vytvořená aplikace je zatím určena pro použití jedním člověkem. Dále co by se mohlo vytvořit v této aplikaci možnost připojit se přes více nástrojů jako například přes bluetooth. Zatím má totiž aplikace přístupné posílání textu pouze přes Firebase.

Pokud bych shrnul druhou aplikaci, tak aplikace má hotové všechny body co byly zadány pro tuto bakalářskou práci. Aplikace dokáže zobrazit krátký text ve vhodném formátování a ve volitelné velikosti fondu. Dokáže sledovat přiložený prst a detekovat písmeno pod tímto prstem, které je pak posláno na spárované zařízení. Přestože jsou tyto body splněny, tak tato aplikace má i několik nedostatků, které by se do budoucna mohli změnit. Jako první z těchto nedostatků je malé množství prostředků, přes které se dá přenášet zadaný text. Dalším nedostatkem je neúplné rozpoložení textu po celé obrazovce, kdy prozatím je rozpoložení textu limitováno určitou výškou. Toto všechno by se mohlo v budoucnu vyřešit. K této aplikaci by se také mohl připevnit držák ze špejlemi, které by byly v souladu s řádky, aby mohl hluchoslepý člověk jezdit prstem po displeji, tak aby mu prst nesjel z dané řádky.

3 Závěr

Cílem mé bakalářské práce bylo prostudovat komunikační možnosti pro lidi s postižením zraku, sluchu a zraku a sluchu. Dalším cílem bylo vytvořit aplikaci pro tablety, která bude určena pro hluchoslepé lidi. Tuto bakalářskou práci jsem si vybral z toho důvodu, že ve světě neexistuje mnoho aplikací, které jsou vytvořeny pro jejich pomoc s komunikací a získáváním informací. Dalším důvodem bylo to, že jsem chtěl upozornit na toto postižení.

V teoretické části mé bakalářské práce jsem popsal definice slepoty, hluchoty a hluchoslepoty. Popsal jsem alternativní komunikační systémy, se kterými komunikují hluchoslepí lidé a technologie a mobilní aplikace, které se využívají pro pomoc těmto postiženým lidem. Vyjmenoval a popsal jsem různé platformy, které se využívají pro vývoj aplikací na chytrý telefon a tablet. Myslím si, že všechny kapitoly v této části jsou srozumitelné a přehledné.

V praktické části jsem pospal, jakou platformu jsem si vybral pro vytvoření aplikace. Popsal jsem, jak se v ní dají vytvořit dané projekty a co vše se v ní nachází. Vyjmenoval a popsal jsem, jaké existují funkce pro spárování zařízení. Pak jsem popisoval mé dvě aplikace, jenž jsem rozdělil na aplikaci pro personál a aplikaci pro hluchoslepé. Rozebral jsem je tak, že jsem jako první představil co tato aplikace znamená a poté už jsem popisoval, jaký kód byl použit a zobrazení. U aplikace pro hluchoslepé jsem hlavně popsal, jak je text zobrazován, jak zvětšit text nebo jak je tato aplikace schopna sledovat pozici přiloženého prstu a detekovat pod ním zobrazený znak, který byl pak poslán do spárovaného zařízení (první aplikace). Toto splňuje několik bodů, které byly vytyčeny pro zpracování této aplikace. Samozřejmě by zde tato aplikace mohla mít i několik vylepšení do budoucna. Jedním z nich by mohlo být to, že by se mi text roztáhl na celou obrazovku, jelikož teď je text zatím daný jen na určitou výšku. Do budoucna by se k této aplikaci také mohl připecnit držák ze špejlemi, který by mohl pomoci k držení prstu na jedné řádce a také by se k tomu mohlo udělat hardwarové zařízení, které by tyto znaky přetvořilo do fyzické podoby.

Na závěr bych chtěl uvést, že práce na této aplikaci mě obohatila ve směru programování mobilních aplikací. Poprvé jsem programoval v Android studiu, a tato práce mě naučila podrobné věci a byla také velmi zajímavá. Rád bych se k tomuto programu a k této problematice v budoucnu vrátil a zabýval se jí.

Seznam použité literatury

[1] *Slepota*. URL: https://cs.wikipedia.org/wiki/Slepota (cit. 20. 01. 2023).

[2] *Hluchota*. URL: https://cs.wikipedia.org/wiki/Hluchota (cit. 23. 01. 2023).

[3] *Lorm*. URL: https://www.lorm.cz/pro-hluchoslepe/definice-hluchoslepoty/ (cit. 26. 01. 2023).

[4] *Duální senzorické onemocnění*. URL: https://www.ddmsberoun.cz/ (cit. 10. 02. 2023).

[5] *Tyfloservis*. URL: http://www.tyfloservis.cz/o-nas/ (cit. 23. 02. 2023).

[6] *SPC pro děti s vadami s řeč*. URL: https://www.alternativnikomunikace.cz/stranka-co-je-aak-9 (cit. 23. 02. 2023).

[7] *Prstová abeced*. URL: http://www.svetabeced.cz/ostatni/prstova-abeceda/ (cit. 23. 03. 2023).

[8] *Prstová abeced*. URL: http://ruce.cz/clanky/3-prstova-abeceda (cit. 23. 02. 2023).

[9] *TalkBack*. URL: https://www.samsung.com/cz/support/mobile-devices/jak-pouzivat-funkci-talkback/ (cit. 23. 03. 2023).

[10] *Galop, Breaillské zobrazovače a hmatová grafi*. URL: https://www.galop.cz/katalog.php?sk=18 (cit. 23. 03. 2023).

[11] *Sluchadla kotvená do kosti*. URL: https://www.otorinolaryngologie.cz/content/uploads/2020/02/ppp-baha.pdf (cit. 23. 03. 2023).

[12] *SCREENING Sluchu dětí ve věku 5 let*. URL: https://www.otorinolaryngologie.cz/content/uploads/2020/02/ppp-screening-sluchu-deti-5let.pdf (cit. 23. 03. 2023).

[13] *Android Studio*. URL: https://developer.android.com/ (cit. 10. 04. 2023).

[14] *Xcode*. URL: https://developer.apple.com/xcode (cit. 10. 04. 2023).

[15] *React Native*. URL: https://reactnative.dev/ (cit. 10. 04. 2023).

[16] *Flutter*. URL: https://flutter.dev/ (cit. 10. 04. 2023).

LUDÍKOVÁ, L.: Vzdělávání hluchoslepých I., Scientia, 2000., 74 s. ISBN 80-7183-225-1

Šarounová, J.: Metody alternativní a augmentativní komunikace, 2014, ISBN 978-80-262-0716-0

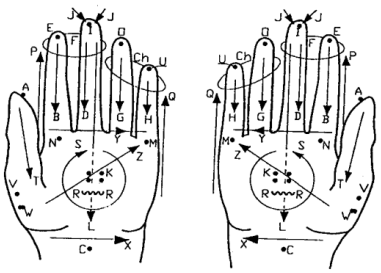
Emili C.Bouck: Assistive Technology, SAGE Publications 2015, IBSN 1483374459

Hersch, Johnson: Assistive Technology for the Hearing-impaired, Deaf and Deafblind, SpringerLink 2015, ISBN 978-1447139218

Příloha č. 1 – Lormova abeceda

POPIS ČESKÉ LORMOVY ABECEDY

Ke komunikaci se využívá dlaňová strana nejlépe levé ruky, prsty této ruky se drží poněkud napjaté a mírně roztažené. Podle potřeby je možno používat i dlaně pravé ruky. Mluvčí vyznačuje jednotlivá písma svým ukazováčkem do dlaně a na prsty ruky příjemce sdělovaných informací dle následujícího schématu:



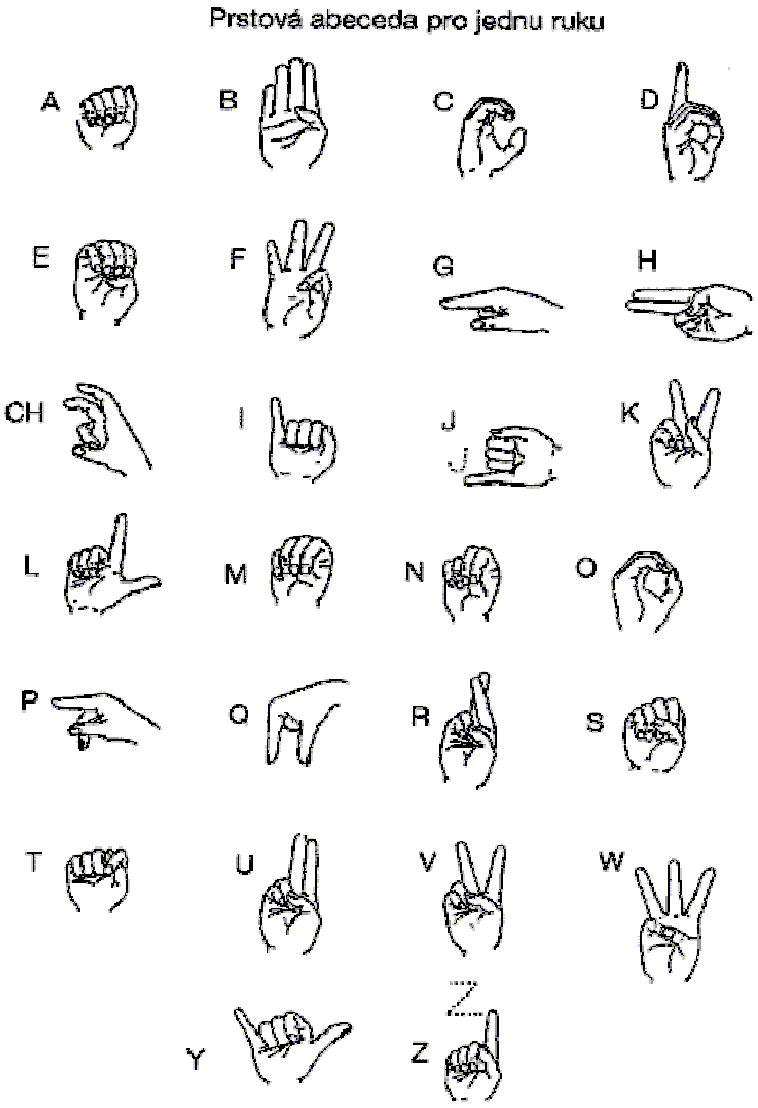
A - bod na špičce palce
B - čára po ukazováčku od špičky prstu k dlani
C - bod na zápěstí
D - čára po prostředníčku od špičky prstu k dlani
E - bod na špičce ukazováčku
F - současné stisknutí špiček ukazováčku a prostředníčku ze strany
G - čára po prsteníčku od špičky prstu k dlani
H - čára po malíčku od špičky prstu k dlani
CH - současné stisknutí špiček prsteníčku a malíčku ze strany
I - bod na špičce prostředníčku
J - stisk špičky prostředníčku ze strany
K - bod čtyř špiček prstů do dlaně
L - čára po prostředníčku od špičky prstu přes dlaň k zápěstí
M - bod pod malíčkem
N - bod pod ukazováčkem
O - bod na špičce prsteníčku
P - čára po vnější straně ukazováčku od dlaně ke špičce ukazováčku
Q - čára po vnější straně malíčku od dlaně ke špičce malíčku
R - postupné pokládání ukazováčku, prostředníčku a prsteníčku do dlaně
S - ukazováčkem kruh na dlani
T - čára po palci od špičky prstu k dlani
U - bod na špičce malíčku
V - bod pod palcem
W - dvakrát bod pod palcem
X - čára podél zápěstí zleva doprava
Y - čára pod prsty směrem od ukazováčku k malíčku
Ý - čára pod prsty směrem od ukazováčku k malíčku a pokračovat po vnější straně malíčku směrem ke špičce malíčku
Z - šikmá čára přes dlaň od palce k malíčku

DLOUHÉ SAMOHLÁSKY - krátká čárka na špičce prstu pro příslušnou samohlásku směrem nahoru
HÁČEK NAD PÍSMENY - bod mezi palcem a ukazováčkem před příslušnou hláskou
ČÍSLA - lze vyjádřit dvěma způsoby: 1) arabské číslice napsat obrysově do dlaně; označení tisíc, milión, miliarda zkratkou (tis., mil., mld.), 2) použít značení jako v Braillově písmu 1=A, 2=B, 3 =C, 4=D, 5=E, 6=F, 7=G, 8=H, 9=I, 0 =J, před číslicí použít čáru obráceného "L" od zápěstí směrem ke špičce prostředníku.
„NEROZUMÍM“ - zavřít dlaň
OMYL - lehké klepnutí do dlaně
MEZERA MEZI SLOVY - plochou ruky přejet jedenkrát po dlani
OTAZNÍK - ukazováčkem vypsat do dlaně obrys otazníku
KONEC VĚTY - plochou ruky přejet dvakrát po dlani

Příloha č.2 - Prstová abeceda

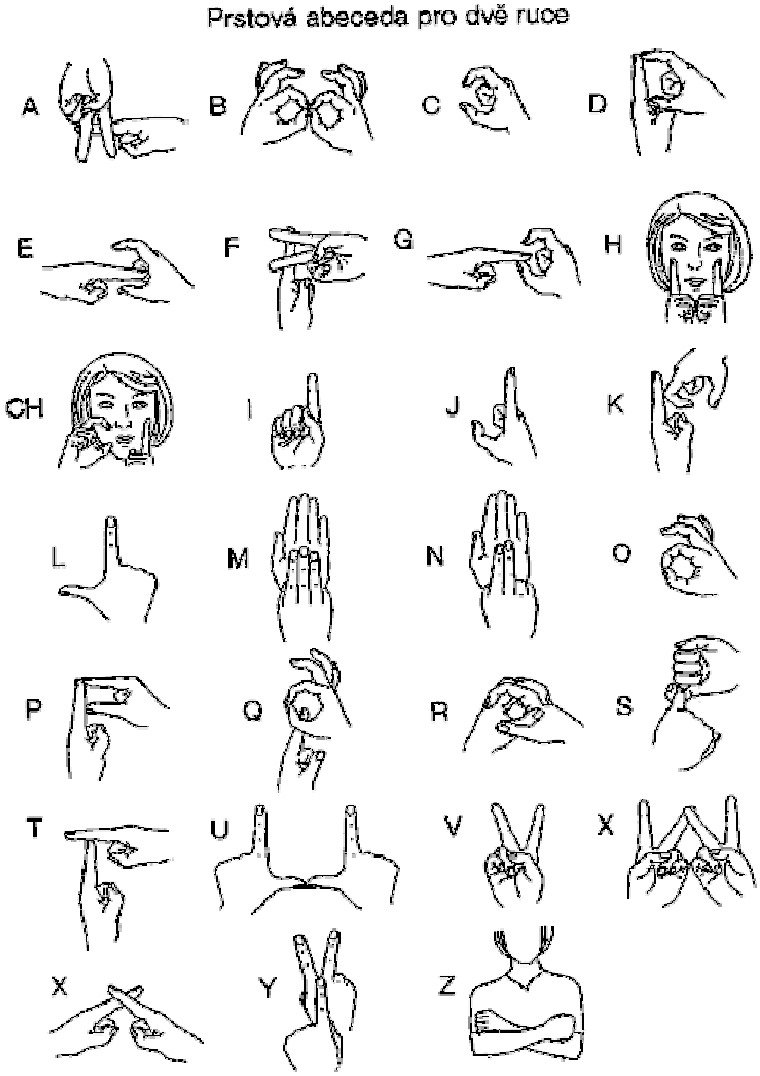
Příloha č.1.

Obrázek č.1. - Jednoruční prstová abeceda



Příloha č.1.

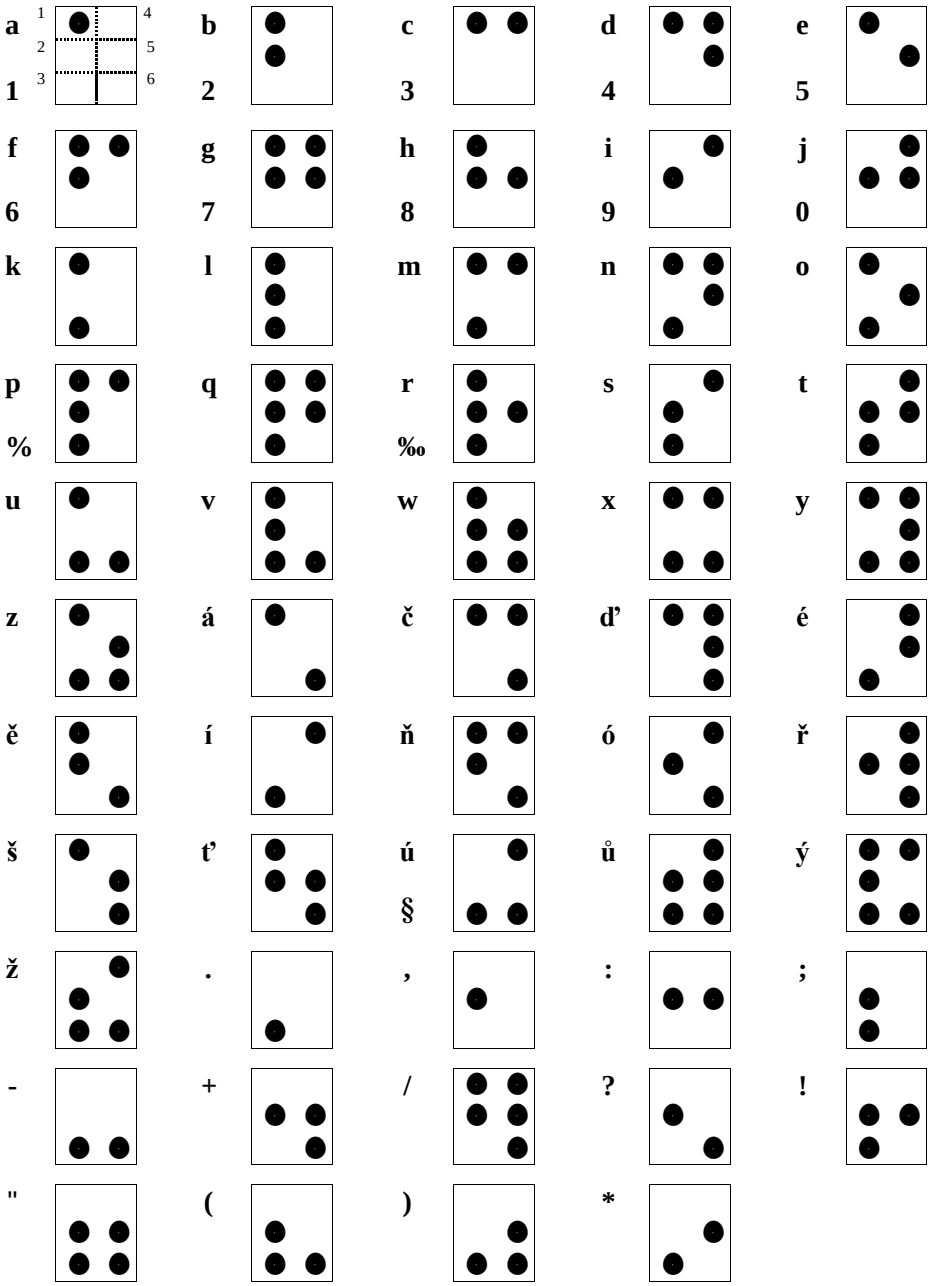
Obrázek č. 2. – Dvouruční prstová abeceda

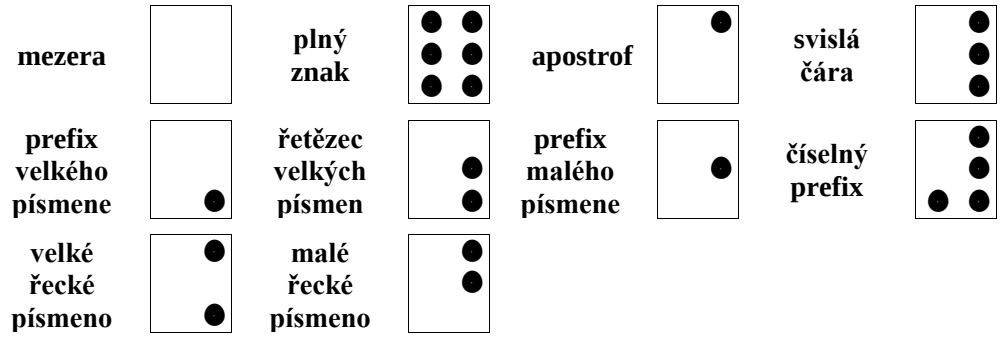


Příloha č.3 - Braillovo písmo

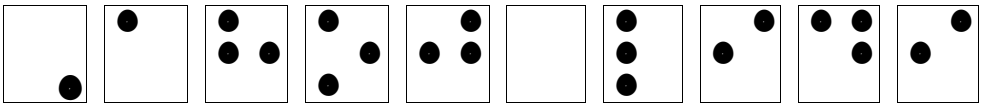
Slepecká Braillova abeceda

Pro zápis veškerých textů se používá **česká základní znaková sada Braillova písma**. Existuje 64 kombinací šesti bodů uspořádaných do dvou sloupečků a třech řad, které jsou buď vytlačené nebo nevytlačené. Pro zápis mnoha znaků 64 kombinací nestačí a proto existují tzv. prefixy, které mění význam znaku nebo skupiny znaků, před jimiž stojí.

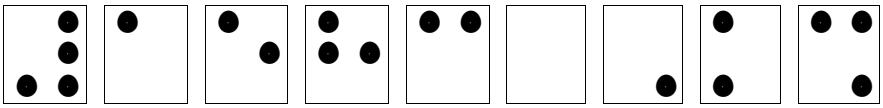




„Ahoj lidi“

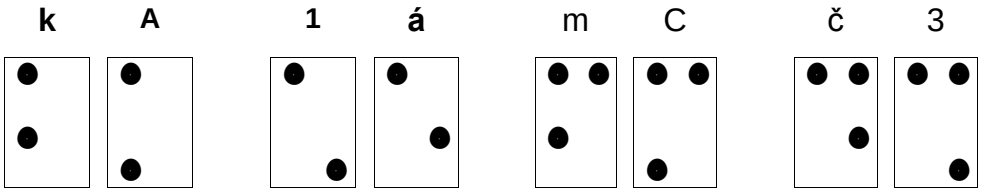


„1583 Kč“



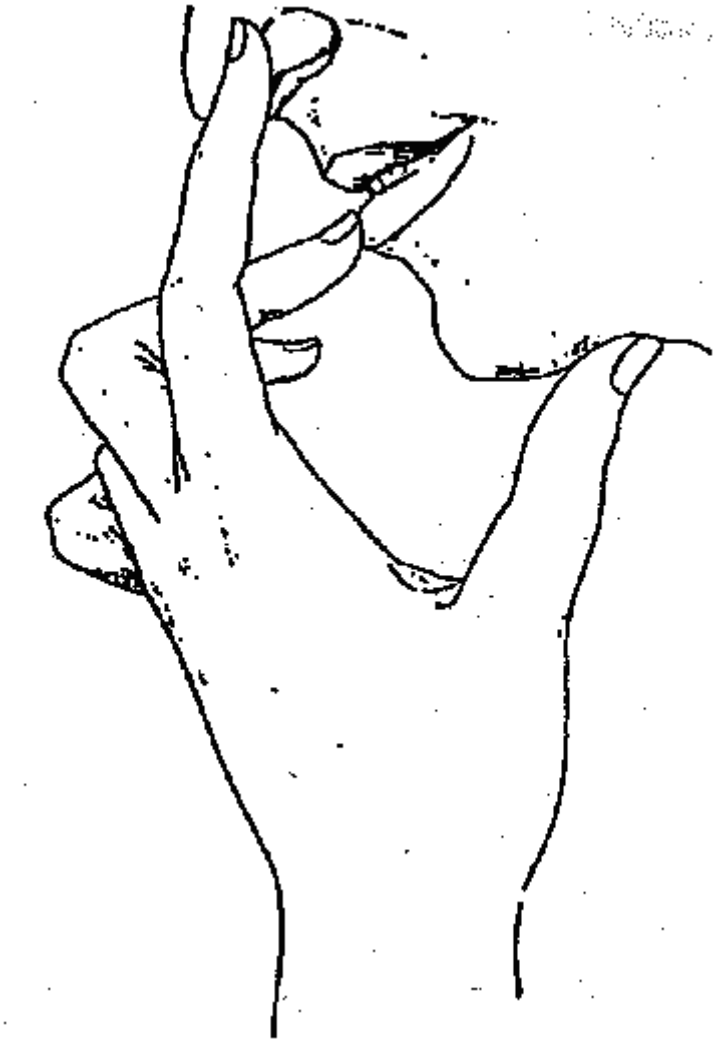
Existuje i písmo osmiznakové, které se používá v souvislosti s výpočetní technikou.

Osmibodové písmo však může při čtení hmatem činit určité obtíže:



Osmibodové písmo však umožňuje každý znak vyjádřit jednoznačnou kombinací bodů, stejně tak jako u binárního zápisu znaků v paměti PC.

Příloha č. 4 - Tadoma



Obrázek 14: Tadoma

Příloha č.5 - První Aplikace kód Java

```java
package com.example.wifi;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        FirebaseDatabase database = ...
            FirebaseDatabase.getInstance("https://wifi-7efe9-default-rtdb.europe-west1.firebase
        DatabaseReference Odeslat = database.getReference("prvni");
        DatabaseReference Prijmout = database.getReference("druhy");

        TextView ZobrazeniTextu = findViewById(R.id.textView);
        Button tlacitko = findViewById(R.id.button);
        EditText VlozitText = findViewById(R.id.edittext);

        //odeslan  textu
        tlacitko.setOnClickListener(v -> {
            String text = VlozitText.getText().toString();

            Odeslat.setValue(text);
            ZobrazeniTextu.setText("");
        });

        //prijmuti pismen
        Prijmout.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                String text = dataSnapshot.getValue(String.class);
                //ZobrazeniTextu.setText(text);
                ZobrazeniTextu.append(text);
            }
        });
    }
}
```

Příloha č.6 - První Aplikace kód XML

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/textInputLayout"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:layout_marginStart="1dp"
        android:layout_marginTop="5dp"
        android:layout_marginBottom="19dp">

        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/edittext"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:hint="@string/napi_zpr_vu"
            android:textColor="@color/black"
            android:textColorHint="@color/black"
            tools:ignore="TextContrastCheck" />

    </com.google.android.material.textfield.TextInputLayout>

    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/odeslat" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:layout_marginStart="1dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="1dp"
        android:textSize="16sp"
        android:text="@string/textview"
        android:textColor="@color/black"/>
</LinearLayout>
```

Příloha č.7 - Druhá Aplikace kód Java Aktivity1

```
1  package com.example.aplikaceprohluchoslepe;
2
3  import androidx.activity.result.ActivityResultLauncher;
4  import androidx.activity.result.contract.ActivityResultContracts;
5  import androidx.annotation.NonNull;
6  import androidx.appcompat.app.AppCompatActivity;
7
8  import android.app.Activity;
9  import android.content.Context;
10 import android.content.Intent;
11 import android.content.SharedPreferences;
12 import android.graphics.Color;
13 import android.graphics.Rect;
14 import android.os.Build;
15 import android.os.Bundle;
16 import android.os.Handler;
17 import android.os.Looper;
18 import android.os.VibrationEffect;
19 import android.os.Vibrator;
20 import android.text.Layout;
21 import android.text.Spannable;
22 import android.text.SpannableString;
23 import android.text.style.ForegroundColorSpan;
24 import android.util.Log;
25 import android.util.TypedValue;
26 import android.view.MotionEvent;
27 import android.view.ViewTreeObserver;
28 import android.widget.Button;
29 import android.widget.Scroller;
30 import android.widget.TextView;
31 import android.widget.Toast;
32
33 import com.google.firebase.database.DataSnapshot;
34 import com.google.firebase.database.DatabaseError;
35 import com.google.firebase.database.DatabaseReference;
36 import com.google.firebase.database.FirebaseDatabase;
37 import com.google.firebase.database.ValueEventListener;
38
39 import java.util.HashSet;
40 import java.util.Set;
41
42 public class MainActivity extends AppCompatActivity {
43
44     boolean Tlacitko3Zmacknuto = false;
45     boolean Tlacitko5Zmacknuto = false;
46
47     private int Souradnicey = 0;
48     private TextView textView;
49
```

```
50      FirebaseDatabase database = ...
            FirebaseDatabase.getInstance("https://wifi-7efe9-default-rtdb.europe-west1.firebasedata
51      DatabaseReference Odchozi = database.getReference("druhy");
52      DatabaseReference Prihozi = database.getReference("prvni");
53
54      public static final String VyskaTextu = "com.example.myapp.TEXT_SIZE";
55      public static final String RozsahMezery = "com.example.myapp.LETTER_SPACING";
56      public static final String RozsahHorniRadky = ...
            "com.example.myapp.EXTRA_PADDING";
57      public static final String RozsahVsechRadku = ...
            "com.example.myapp.EXTRA_LINESPACE";
58      private final Set<String> UlozeniPismen = new HashSet<>();
59      private final Handler ResetovaniUlozeni = new ...
            Handler(Looper.getMainLooper());
60
61      @Override
62      protected void onCreate(Bundle savedInstanceState) {
63          super.onCreate(savedInstanceState);
64          setContentView(R.layout.activity_main);
65
66          textView = findViewById(R.id.textView);
67          Button Tlacitko1 = findViewById(R.id.Tlacitko1);
68          Button Tlacitko3 = findViewById(R.id.Tlacitko3);
69          Button Tlacitko5 = findViewById(R.id.Tlacitko5);
70          Button Tlacitko7 = findViewById(R.id.Tlacitko7);
71
72          //u d  l   textview  Posouvac
73          textView.setScroller(new Scroller(this));
74
75          //sleduje jetli  u   je textview  n a  e t l
76          ViewTreeObserver NacteniObrazovky = textView.getViewTreeObserver();
77          NacteniObrazovky.addOnGlobalLayoutListener(new ...
                ViewTreeObserver.OnGlobalLayoutListener() {
78              @Override
79              public void onGlobalLayout() {
80                  int[] lokace = new int[2];
81                  textView.getLocationOnScreen(lokace);
82                  int y = lokace[1];
83                  Log.d("MyActivity", "Sou adnice Y: " + y);
84                  Souradnicey = y;
85                  textView.getViewTreeObserver().removeOnGlobalLayoutListener(this);
86              }
87          });
88
89          //posouvan  nahoru
90          Tlacitko1.setOnClickListener(v -> {
91              if (textView.getScrollY() > 0)
92                  textView.scrollTo(0, textView.getScrollY() - ...
                        5*textView.getLineHeight());
93          });
94
95          Tlacitko3.setOnLongClickListener(v -> {
96              Tlacitko3Zmacknuto = true;
```

```
97
98              new Handler().postDelayed(() -> Tlacitko3Zmacknuto = false, ...
                    2000); // Po dvou sekundch se to resetuje
99              ZmacknutiObouchTlacitek();
100             return true;
101         });
102
103         Tlacitko5.setOnLongClickListener(v -> {
104             Tlacitko5Zmacknuto = true;
105             new Handler().postDelayed(() -> Tlacitko5Zmacknuto = false, 2000);
106             ZmacknutiObouchTlacitek();
107             return true;
108         });
109
110         //posouvan dolu
111         Tlacitko7.setOnClickListener(v -> {
112             int MaximalniPosouvani = ...
                    textView.getLayout().getLineTop(textView.getLineCount() - 0) ...
                    - textView.getHeight();
113             if (textView.getScrollY() < MaximalniPosouvani)
114                 textView.scrollTo(0, textView.getScrollY() + ...
                        5*textView.getLineHeight());
115         });
116
117         DostaniTextu();
118         NacistPromene();
119     }
120
121     private void ZmacknutiObouchTlacitek() {
122         if (Tlacitko3Zmacknuto && Tlacitko5Zmacknuto) {
123             Intent Poslat = new Intent(this, MainActivity2.class);
124
125             TextView textView = findViewById(R.id.textView);
126             float StavajiciVyskaTextu = textView.getTextSize();
127             float StavajiciRozsahMezery = textView.getLetterSpacing();
128             int topPadding = textView.getPaddingTop();
129             float LineSpace = textView.getLineSpacingExtra();
130
131             Poslat.putExtra(VyskaTextu, StavajiciVyskaTextu);
132             Poslat.putExtra(RozsahMezery, StavajiciRozsahMezery);
133             Poslat.putExtra(RozsahHorniRadky, topPadding);
134             Poslat.putExtra(RozsahVsechRadku, LineSpace);
135
136
137             VibraceTelefonu(this, 500);
138             NavazaniSpojeniSDruhouAktivitou.launch(Poslat);
139
140
141         }
142     }
143
144     ActivityResultLauncher<Intent> NavazaniSpojeniSDruhouAktivitou = ...
            registerForActivityResult(
```

```
145             new ActivityResultContracts.StartActivityForResult(),
146             result -> {
147                 if (result.getResultCode() == Activity.RESULT_OK) {
148                     Intent Prijmout = result.getData();
149                     if (Prijmout != null) {
150                         float NovaVyskaTextu = ...
                                Prijmout.getFloatExtra(VyskaTextu, 60);
151                         float NoveRozmeziMezery = ...
                                Prijmout.getFloatExtra(RozsahMezery, 0.0f);
152                         float NovyRozmeziVsechRadku = ...
                                Prijmout.getFloatExtra(MainActivity.RozsahHorniRadky, ...
                                40);
153                         int NovyRozmeziHornihoRadku = (int) ...
                                Prijmout.getFloatExtra(MainActivity.RozsahVsechRadku, ...
                                40);
154
155                         UlozitPromene(NovaVyskaTextu, NoveRozmeziMezery, ...
                                NovyRozmeziVsechRadku, NovyRozmeziHornihoRadku);
156
157                         int LevaMezeraRadku = textView.getPaddingLeft();
158                         int PravaMezeraRadku = textView.getPaddingRight();
159                         int DolniMezeraRadku = textView.getPaddingBottom();
160
161
162                         textView.setTextSize(TypedValue.COMPLEX_UNIT_PX, ...
                                NovaVyskaTextu);
163                         textView.setLetterSpacing(NoveRozmeziMezery);
164                         textView.setPadding(LevaMezeraRadku, ...
                                NovyRozmeziHornihoRadku, PravaMezeraRadku, ...
                                DolniMezeraRadku);
165                         textView.setLineSpacing(NovyRozmeziVsechRadku, 1.0f);
166                     }
167                 }
168             });
169
170     private void UlozitPromene(float novaVyskaTextu, float noveRozmeziMezery, ...
            float novyRozmeziVsechRadku, int novyRozmeziHornihoRadku) {
171         SharedPreferences preferences = ...
                getSharedPreferences("UlozeneHodnoty", MODE_PRIVATE);
172         SharedPreferences.Editor editor = preferences.edit();
173         editor.putFloat("NovaVyskaTextu", novaVyskaTextu);
174         editor.putFloat("NoveRozmeziMezery", noveRozmeziMezery);
175         editor.putFloat("NovyRozmeziVsechRadku", novyRozmeziVsechRadku);
176         editor.putInt("NovyRozmeziHornihoRadku", novyRozmeziHornihoRadku);
177         editor.apply();
178     }
179
180     public void NacistPromene() {
181         SharedPreferences sharedPreferences = ...
                getSharedPreferences("MojeUlozeneHodnoty", MODE_PRIVATE);
182         float NovaVyskaTextu = sharedPreferences.getFloat("NovaVyskaTextu", 60);
183         float NoveRozmeziMezery = ...
                sharedPreferences.getFloat("NoveRozmeziMezery", 0.0f);
```

```
184         float NovyRozmeziVsechRadku = ...
                sharedPreferences.getFloat("NovyRozmeziVsechRadku", 60);
185         int NovyRozmeziHornihoRadku = ...
                sharedPreferences.getInt("NovyRozmeziHornihoRadku", 60);
186
187         int LevaMezeraRadku = textView.getPaddingLeft();
188         int PravaMezeraRadku = textView.getPaddingRight();
189         int DolniMezeraRadku = textView.getPaddingBottom();
190
191         textView.setTextSize(TypedValue.COMPLEX_UNIT_PX, NovaVyskaTextu);
192         textView.setLetterSpacing(NoveRozmeziMezery);
193         textView.setPadding(LevaMezeraRadku, NovyRozmeziHornihoRadku, ...
                PravaMezeraRadku, DolniMezeraRadku);
194         textView.setLineSpacing(NovyRozmeziVsechRadku, 1.0f);
195     }
196
197
198     public void VibraceTelefonu(Context lokace, long cas) {
199         Vibrator vibrace = (Vibrator) ...
                lokace.getSystemService(Context.VIBRATOR_SERVICE);
200
201         if (Build.VERSION.SDK_INT ≥ Build.VERSION_CODES.O) {
202             VibrationEffect effect = VibrationEffect.createOneShot(cas, ...
                    VibrationEffect.DEFAULT_AMPLITUDE);
203             vibrace.vibrate(effect);
204         } else {
205             vibrace.vibrate(cas);
206         }
207     }
208
209     @Override
210     public boolean onTouchEvent(MotionEvent event) {
211         switch (event.getAction()) {
212             case MotionEvent.ACTION_DOWN:
213                 float x = event.getX() - textView.getPaddingLeft();
214                 float y = event.getY() - textView.getPaddingTop();
215                 y = y - Souradnicey;
216                 KontrolaPozicePismene(x, y);
217                 break;
218             case MotionEvent.ACTION_MOVE:
219                 x = event.getX() - textView.getPaddingLeft();
220                 y = event.getY() - textView.getPaddingTop();
221                 y = y - Souradnicey;
222                 KontrolaPozicePismene(x, y);
223                 break;
224             default:
225                 break;
226         }
227         return true;
228     }
229
230     private void KontrolaPozicePismene(float x, float y) {
231         Layout rozlozeni = textView.getLayout();
```

```
232         String text = textView.getText().toString();
233         boolean PismenoJeTam = false;
234 
235         for (int i = 0; i < text.length(); i++) {
236             char Znak = text.charAt(i);
237 
238             // Z  sk  n   Sou adnic
239             int radek = rozlozeni.getLineForOffset(i);
240             float souradniceX = rozlozeni.getPrimaryHorizontal(i);
241             float souradniceY = rozlozeni.getLineBaseline(radek) - ...
                    textView.getScrollY();
242 
243 
244             // Z  sk  n     ky   a v  k y  p smen
245             Rect HranicePismene = new Rect();
246             textView.getPaint().getTextBounds(String.valueOf(Znak), 0, 1, ...
                    HranicePismene);
247             float TlouskaPisma = HranicePismene.width();
248             float VyskaPisma = HranicePismene.height();
249 
250             // P  i d  n  poloviny   ky   a v  k y  p smene k ...
                    sou adnic m X a Y
251             float souradniceXprostedku = souradniceX + TlouskaPisma / 2;
252             float souradniceYprostedku = souradniceY - VyskaPisma / 2;
253 
254             // V po et vzd lenosti mezi st edem p smene a sou adnicemi ...
                    dotyku
255             float vzd lenost = (float) Math.sqrt((souradniceXprostedku - x) ...
                    * (souradniceXprostedku - x) + (souradniceYprostedku - y) * ...
                    (souradniceYprostedku - y));
256 
257             if(Znak == 'I' || Znak == 'i' || Znak == 'j' || Znak == 'J' || ...
                    Znak == 'l' || Znak == 't') {
258                 float VetsiTlouskaPismene = TlouskaPisma * 2f;
259                 if (vzd lenost < VetsiTlouskaPismene) {
260                     String OznacenePismeno = Znak + "_" + i;
261                     if (!UlozeniPismen.contains(OznacenePismeno)) {
262                         PrejetiPresPismeno(String.valueOf(Znak));
263                         UlozeniPismen.add(OznacenePismeno);
264                         VybraceTelefonu(this, 100);
265                         ZmenaBarvyPismene(i);
266                     }
267                     PismenoJeTam = true;
268                     break;
269                 }
270             }
271 
272             if (vzd lenost < TlouskaPisma) {
273                 String OznacenePismeno = Znak + "_" + i;
274                 if (!UlozeniPismen.contains(OznacenePismeno)) {
275                     PrejetiPresPismeno(String.valueOf(Znak));
276                     UlozeniPismen.add(OznacenePismeno);
277 
```

```
278                     ZmenaBarvyPismene(i);
279                 }
280                 PismenoJeTam = true;
281                 break;
282             }
283         }
284
285         if (!PismenoJeTam) {
286             int HodnotaMezery = ZjisteniMezery(x, y);
287             if (HodnotaMezery != -1) {
288                 String OznacenaMezera = " " + "_" + HodnotaMezery;
289                 if (!UlozeniPismen.contains(OznacenaMezera)) {
290                     PrejetiPresPismeno(" ");
291                     UlozeniPismen.add(OznacenaMezera);
292                 }
293             }
294         }
295
296     }
297
298     private int ZjisteniMezery(float x, float y) {
299         Layout rozlozeni = textView.getLayout();
300         String text = textView.getText().toString();
301
302         for (int i = 0; i < text.length(); i++) {
303             char Znak = text.charAt(i);
304             int radek = rozlozeni.getLineForOffset(i);
305             float souradniceX = rozlozeni.getPrimaryHorizontal(i);
306             float souradniceY = rozlozeni.getLineBaseline(radek) - ...
                    textView.getScrollY();
307
308             if (Znak == ' ') {
309                 float DalsiPismeno;
310                 if (i < text.length() - 1) {
311                     DalsiPismeno = rozlozeni.getPrimaryHorizontal(i + 1);
312                 } else {
313                     DalsiPismeno = textView.getWidth() - ...
                            textView.getPaddingRight();
314                 }
315
316                 if (x ≥ souradniceX && x ≤ DalsiPismeno && Math.abs(y - ...
                        souradniceY) < 25) {
317                     return i;
318                 }
319             }
320         }
321         return -1;
322     }
323
324     private void PrejetiPresPismeno(String Znak) {
325         Log.d("Znak", "Znak: " + Znak);
326
327         long UnikatniCislo = System.currentTimeMillis();
```

```
328             String OznacenyZnak = Znak + "_" + UnikatniCislo;
329             if (UlozeniPismen.add(OznacenyZnak)) {
330                 Odchozi.setValue(OznacenyZnak);
331             }
332             ResetovaniUlozenychPismen();
333         }
334
335         private void ResetovaniUlozenychPismen() {
336             ResetovaniUlozeni.removeCallbacksAndMessages(null);
337             ResetovaniUlozeni.postDelayed(UlozeniPismen::clear, 5000);
338         }
339
340         private void ZmenaBarvyPismene(int Znak) {
341             Spannable Objekt = new SpannableString(textView.getText());
342             Objekt.setSpan(new ForegroundColorSpan(Color.RED), Znak, Znak + 1, ...
                    Spannable.SPAN_EXCLUSIVE_EXCLUSIVE);
343             textView.setText(Objekt);
344         }
345
346         private void DostaniTextu() {
347             Prihozi.addValueEventListener(new ValueEventListener() {
348                 @Override
349                 public void onDataChange(@NonNull DataSnapshot snapshot) {
350                     String data = snapshot.getValue(String.class);
351                     if (data != null) {
352                         TextView textView = findViewById(R.id.textView);
353                         textView.setText(data);
354
355                     }
356                 }
357
358                 @Override
359                 public void onCancelled(@NonNull DatabaseError error) {
360                     Toast.makeText(MainActivity.this, "Chyba p i   ten   dat: " ...
                            + error.getMessage(), Toast.LENGTH_SHORT).show();
361                 }
362             });
363         }
364 }
```

Příloha č.8 - Druhá Aplikace kód XML Aktivity1

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:orientation="horizontal"
```

```
8       tools:context=".MainActivity">
9
10      <TextView
11          android:id="@+id/textView"
12          android:layout_width="0dp"
13          android:layout_height="wrap_content"
14          android:layout_weight="1"
15          android:textSize="60px"
16          android:padding="60px"
17          android:textColor="@color/black"
18          android:background="@color/white"
19          android:lineSpacingExtra="60px"
20          android:maxLines="5"
21          android:text="Pro gunk nost t to aplikace je nutn  p ipojen  k ...
                internetu. (Wifi)" />
22
23      <LinearLayout
24          android:layout_width="100dp"
25          android:layout_height="match_parent"
26          android:orientation="vertical">
27
28          <Button
29              android:id="@+id/Tlacitko1"
30              android:layout_width="match_parent"
31              android:layout_height="0dp"
32              android:layout_weight="1"
33              android:text="PAGE UP"/>
34
35          <View
36              android:layout_width="match_parent"
37              android:layout_height="0dp"
38              android:layout_weight="1" />
39
40          <Button
41              android:id="@+id/Tlacitko3"
42              android:layout_width="match_parent"
43              android:layout_height="0dp"
44              android:layout_weight="1"
45              android:text="MENU"/>
46
47          <View
48              android:layout_width="match_parent"
49              android:layout_height="0dp"
50              android:layout_weight="1" />
51
52          <Button
53              android:id="@+id/Tlacitko5"
54              android:layout_width="match_parent"
55              android:layout_height="0dp"
56              android:layout_weight="1"
57              android:text="MENU"/>
58
59          <View
```

```
60              android:layout_width="match_parent"
61              android:layout_height="0dp"
62              android:layout_weight="1" />
63
64          <Button
65              android:id="@+id/Tlacitko7"
66              android:layout_width="match_parent"
67              android:layout_height="0dp"
68              android:layout_weight="1"
69              android:text="PAGE DOWN" />
70
71      </LinearLayout>
72
73  </LinearLayout>
```

Příloha č.9 - Druhá Aplikace kód Java Aktivity2

```
1  package com.example.aplikaceprohluchoslepe;
2
3  import androidx.appcompat.app.AppCompatActivity;
4
5  import android.app.Activity;
6  import android.content.Context;
7  import android.content.Intent;
8  import android.os.Build;
9  import android.os.Bundle;
10 import android.os.Handler;
11 import android.os.VibrationEffect;
12 import android.os.Vibrator;
13 import android.util.Log;
14 import android.util.TypedValue;
15 import android.widget.Button;
16 import android.widget.TextView;
17 import android.widget.Toast;
18
19 public class MainActivity2 extends AppCompatActivity {
20
21     private float UlozitNovouVelikostTextu;
22     private float UlozitNovyRozsahMezery;
23     private int UlozitNovyRozsahHornihoRadku;
24     private float UlozitNovyRozsahVsechRadku;
25
26     private float KurentniVelikostTextu;
27     private float KurentniRozsahMezery;
28     private int KurentniRozsahHornihoRadku;
29     private float KurentniRozsahVsechRadku;
30
31     private int LevaMezeraRadku;
32     private int PravaMezeraRadku;
```

```
33      private int DolniMezeraRadku;
34
35      private TextView textView1;
36
37      private boolean Tlacitko12Zmacknuto = false;
38      private boolean Tlacitko14Zmacknuto = false;
39
40      private boolean Tlacitko13Zmacknuto = false;
41
42      @Override
43      protected void onCreate(Bundle savedInstanceState) {
44          super.onCreate(savedInstanceState);
45          setContentView(R.layout.activity_main2);
46
47          textView1 = findViewById(R.id.textView1);
48          Button Tlacitko10 = findViewById(R.id.Tlacitko10);
49          Button Tlacitko11 = findViewById(R.id.Tlacitko11);
50          Button Tlacitko12 = findViewById(R.id.Tlacitko12);
51          Button Tlacitko13 = findViewById(R.id.Tlacitko13);
52          Button Tlacitko14 = findViewById(R.id.Tlacitko14);
53          Button Tlacitko15 = findViewById(R.id.Tlacitko15);
54          Button Tlacitko16 = findViewById(R.id.Tlacitko16);
55
56
57
58          //zvětšení velikosti textu
59          Tlacitko10.setOnClickListener(v -> {
60              float VelikostKurentihoTextu = textView1.getTextSize();
61              int HorniRada = textView1.getPaddingTop();
62              float VsechnyRady = textView1.getLineSpacingExtra();
63
64              int NovaHorniRada = HorniRada - 2;
65              if (NovaHorniRada ≥ 40) {
66                  textView1.setPadding(LevaMezeraRadku, NovaHorniRada, ...
                        PravaMezeraRadku, DolniMezeraRadku);
67              }
68
69              float NoveRady = VsechnyRady - 2;
70              if (NoveRady ≥ 40) {
71                  textView1.setLineSpacing(NoveRady, 1.0f);
72              }
73
74              float NovaVelikost = VelikostKurentihoTextu + 2;
75              if (NovaVelikost ≤ 80) {
76                  textView1.setTextSize(TypedValue.COMPLEX_UNIT_PX, NovaVelikost);
77              }
78              VibraceTelefonu(this, 500);
79          });
80
81          //zmenšení velikosti textu
82          Tlacitko11.setOnClickListener(v -> {
83              float VelikostKurentihoTextu = textView1.getTextSize();
84              int HorniRada = textView1.getPaddingTop();
```

```
85              float VsechnyRady = textView1.getLineSpacingExtra();
86
87              int NovaHorniRada = HorniRada + 2;
88              if (NovaHorniRada ≤ 60) {
89                  textView1.setPadding(LevaMezeraRadku, NovaHorniRada, ...
                        PravaMezeraRadku, DolniMezeraRadku);
90              }
91
92              float NoveRady = VsechnyRady + 2;
93              if (NoveRady ≤ 60) {
94                  textView1.setLineSpacing(NoveRady, 1.0f);
95              }
96
97              float NovaVelikost = VelikostKurentihoTextu - 2;
98              if (NovaVelikost ≥ 60) {
99                  textView1.setTextSize(TypedValue.COMPLEX_UNIT_PX, NovaVelikost);
100             }
101             VibraceTelefonu(this, 500);
102         });
103
104         Tlacitko12.setOnLongClickListener(v -> {
105             Tlacitko12Zmacknuto = true;
106             new Handler().postDelayed(() -> Tlacitko12Zmacknuto = false, 2000);
107             OdeslatHodnotyNazptek();
108             return true;
109         });
110
111         //Ulozeni
112         Tlacitko13.setOnLongClickListener(v -> {
113             UlozitNovouVelikostTextu = textView1.getTextSize();
114             UlozitNovyRozsahMezery = textView1.getLetterSpacing();
115             UlozitNovyRozsahHornihoRadku = textView1.getPaddingTop();
116             UlozitNovyRozsahVsechRadku = textView1.getLineSpacingExtra();
117
118             Toast.makeText(MainActivity2.this, "Hodnoty ulo eny", ...
                    Toast.LENGTH_SHORT).show();
119             Tlacitko13Zmacknuto = true;
120             VibraceTelefonu(this, 500);
121             return true;
122         });
123
124         Tlacitko14.setOnLongClickListener(v -> {
125             Tlacitko14Zmacknuto = true;
126             new Handler().postDelayed(() -> Tlacitko14Zmacknuto = false, 2000);
127             OdeslatHodnotyNazptek();
128             return true;
129         });
130
131         //Zvet en  velikosti mezery mezi p smy
132         Tlacitko15.setOnClickListener(v -> {
133             float RozsahKurentiMezery = textView1.getLetterSpacing();
134             float NovyRozsah = RozsahKurentiMezery + 0.01f;
135             if (NovyRozsah ≤ 0.2f) {
```

```
136                 textView1.setLetterSpacing(NovyRozsah);
137             }
138             VibraceTelefonu(this, 500);
139         });
140 
141         //zmen en  velikosti mezery mezi p smy
142         Tlacitko16.setOnClickListener(v -> {
143             float RozsahKurentiMezery = textView1.getLetterSpacing();
144             float NovyRozsah = RozsahKurentiMezery - 0.01f;
145             if (NovyRozsah ≥ 0.0f) {
146                 textView1.setLetterSpacing(NovyRozsah);
147             }
148             VibraceTelefonu(this, 500);
149         });
150 
151 
152         Intent Prijmout = getIntent();
153         KurentniVelikostTextu = ...
                Prijmout.getFloatExtra(MainActivity.VyskaTextu, 60);
154         KurentniRozsahMezery = ...
                Prijmout.getFloatExtra(MainActivity.RozsahMezery, 0.0f);
155         KurentniRozsahVsechRadku = ...
                Prijmout.getFloatExtra(MainActivity.RozsahHorniRadky, 40);
156         KurentniRozsahHornihoRadku = (int) ...
                Prijmout.getFloatExtra(MainActivity.RozsahVsechRadku, 40);
157 
158         LevaMezeraRadku = textView1.getPaddingLeft();
159         PravaMezeraRadku = textView1.getPaddingRight();
160         DolniMezeraRadku = textView1.getPaddingBottom();
161 
162         textView1.setTextSize(TypedValue.COMPLEX_UNIT_PX, KurentniVelikostTextu);
163         textView1.setLetterSpacing(KurentniRozsahMezery);
164         textView1.setPadding(LevaMezeraRadku, KurentniRozsahHornihoRadku, ...
                PravaMezeraRadku, DolniMezeraRadku);
165         textView1.setLineSpacing(KurentniRozsahVsechRadku, 1.0f);
166     }
167 
168     public void VibraceTelefonu(Context lokace, long cas) {
169         Vibrator vibrace = (Vibrator) ...
                lokace.getSystemService(Context.VIBRATOR_SERVICE);
170 
171         if (Build.VERSION.SDK_INT ≥ Build.VERSION_CODES.O) {
172             VibrationEffect effect = VibrationEffect.createOneShot(cas, ...
                    VibrationEffect.DEFAULT_AMPLITUDE);
173             vibrace.vibrate(effect);
174         } else {
175             vibrace.vibrate(cas);
176         }
177     }
178 
179     private void OdeslatHodnotyNazpatek() {
180         if (Tlacitko12Zmacknuto && Tlacitko14Zmacknuto) {
181             Intent Odeslat = new Intent();
```

```
182             if (Tlacitko13Zmacknuto) {
183                 Odeslat.putExtra(MainActivity.VyskaTextu, ...
                        UlozitNovouVelikostTextu);
184                 Odeslat.putExtra(MainActivity.RozsahMezery, ...
                        UlozitNovyRozsahMezery);
185                 Odeslat.putExtra(MainActivity.RozsahHorniRadky, ...
                        UlozitNovyRozsahHornihoRadku);
186                 Odeslat.putExtra(MainActivity.RozsahVsechRadku, ...
                        UlozitNovyRozsahVsechRadku);
187                 Log.d("MainActivity2", "Saved text size: " + ...
                        UlozitNovouVelikostTextu + ", saved letter spacing: " + ...
                        UlozitNovyRozsahMezery + "Saved text size: " + ...
                        UlozitNovyRozsahHornihoRadku + ", saved letter spacing: " ...
                        + UlozitNovyRozsahVsechRadku);
188             } else {
189                 Odeslat.putExtra(MainActivity.VyskaTextu, KurentniVelikostTextu);
190                 Odeslat.putExtra(MainActivity.RozsahMezery, ...
                        KurentniRozsahMezery);
191                 Odeslat.putExtra(MainActivity.RozsahHorniRadky, ...
                        KurentniRozsahHornihoRadku);
192                 Odeslat.putExtra(MainActivity.RozsahVsechRadku, ...
                        KurentniRozsahVsechRadku);
193                 Log.d("MainActivity2", "Returning original text size: " + ...
                        KurentniVelikostTextu + ", original letter spacing: " + ...
                        KurentniRozsahMezery);
194             }
195             VybraceTelefonu(this, 500);
196             setResult(Activity.RESULT_OK, Odeslat);
197             finish();
198         }
199     }
200 }
```

Příloha č.10 - Druhá Aplikace kód XML Aktivity2

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:orientation="horizontal"
8      tools:context=".MainActivity2">
9
10     <TextView
11         android:id="@+id/textView1"
12         android:layout_width="0dp"
13         android:layout_height="match_parent"
14         android:layout_weight="1"
```

```
15          android:textSize="30sp"
16          android:gravity="top"
17          android:text="@string/test_nastaveni" />
18
19      <LinearLayout
20          android:layout_width="100dp"
21          android:layout_height="match_parent"
22          android:orientation="vertical">
23
24          <Button
25              android:id="@+id/Tlacitko10"
26              android:layout_width="match_parent"
27              android:layout_height="0dp"
28              android:layout_weight="1"
29              android:text="A+" />
30
31          <Button
32              android:id="@+id/Tlacitko11"
33              android:layout_width="match_parent"
34              android:layout_height="0dp"
35              android:layout_weight="1"
36              android:text="A-" />
37
38          <Button
39              android:id="@+id/Tlacitko12"
40              android:layout_width="match_parent"
41              android:layout_height="0dp"
42              android:layout_weight="1"
43              android:text="Z P  T" />
44
45          <Button
46              android:id="@+id/Tlacitko13"
47              android:layout_width="match_parent"
48              android:layout_height="0dp"
49              android:layout_weight="1"
50              android:text="ULO IT" />
51
52          <Button
53              android:id="@+id/Tlacitko14"
54              android:layout_width="match_parent"
55              android:layout_height="0dp"
56              android:layout_weight="1"
57              android:text="Z P  T"/>
58
59          <Button
60              android:id="@+id/Tlacitko15"
61              android:layout_width="match_parent"
62              android:layout_height="0dp"
63              android:layout_weight="1"
64              android:text="___+" />
65
66          <Button
67              android:id="@+id/Tlacitko16"
```

```
68              android:layout_width="match_parent"
69              android:layout_height="0dp"
70              android:layout_weight="1"
71              android:text="___-" />
72      </LinearLayout>
73
74  </LinearLayout>
```