

TAGARELA BRAILLE – APP PARA AUXÍLIO NO APRENDIZADO AO BRAILLE

Leonardo Pereira Vieira, Prof. Dalton Solano dos Reis – Orientador

Curso de Bacharel em Ciência da Computação

Departamento de Sistemas e Computação

Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brasil

leonardopereira@furb.br, daltonreis@furb.br

Resumo: Este artigo apresenta o desenvolvimento do aplicativo para auxiliar no aprendizado do Braille inicialmente criado por Cazagrande (2016). Um dos objetivos propostos por esse artigo é tornar o aplicativo mais acessível através de áudio e interação por gestos na tela, possibilitando assim a utilização por pessoas cegas. Este aplicativo pretende também tornar mais fácil a inclusão de novos módulos para que possa ser criado mais ferramentas dentro do aplicativo e abranger outras deficiências. Para isso foi feita a migração da tecnologia utilizada inicialmente para o FLUTTER que utiliza bem o conceito de componentização para reaproveitar os componentes. Foi percebido a necessidade de otimizar as instruções do manual de uso do aplicativo, porém a utilização do aplicativo se mostrou eficiente em acessibilidade, pois cumpriu de forma prática o objetivo de auxiliar e estimular o aprendizado de pessoas com ausência total de visão e com deficiência visual.

Palavras-chave: Acessibilidade. Aplicativo. Braille. Componentização. Flutter.

1 outras áreas da educação especial.

2 com ausência parcial ou total de visão.

1 INTRODUÇÃO

Segundo pesquisa do IBGE (2017) de acordo com o censo de 2010, existe no Brasil quase 46 milhões de pessoas com deficiência, cerca de 24% da população, se declaram deficientes de uma das habilidades investigadas (enxergar, ouvir, caminhar ou subir degraus), ou possuir deficiência mental / intelectual (IBGE, 2017). Dentre esses já passaram ou ainda irão passar pelo período escolar, como é previsto pelo artigo 2º da lei nº 7.853 de 1989, que garante esse direito:

Ao Poder Público e seus órgãos cabe assegurar às pessoas portadoras de deficiência o pleno exercício de seus direitos básicos, inclusive dos direitos à educação, à saúde, ao trabalho, ao lazer, à previdência social, ao amparo à infância e à maternidade, e de outros que, decorrentes da Constituição e das leis, propiciem seu bem-estar pessoal, social e econômico. (BRASIL, 1989, p.1).

Conforme Masini e Gaspareto (2007), esse tipo de inclusão de alunos com deficiência tem se tornado alvo de muita discussão por profissionais da educação. O tema é polêmico por isso tem causado grande impacto e discussão entre professores, diretores e todos os profissionais que atuam no meio escolar (MASINI; GASpareto 2007, p.35). Uma dessas deficiências que os portadores muitas vezes sofrem é com a falta de preparo e/ou adaptação do ambiente escolar e dos profissionais de educação é a deficiência visual. Segundo a organização Marta Gil (2000), cegueira pode ser tanto adquirida como congênita. Quando o indivíduo nasce com o sentido da visão, guarda memórias visuais, facilitando na sua adaptação. Quem nasce sem a capacidade da visão, por outro lado, jamais pode formar uma memória visual, possuir lembrança visual (MARTA GIL, 2000, p.8).

Atualmente existem diversos softwares para o auxílio a pessoas com deficiência como o DOSVOX, BlindSquare, entre outros que são usados pelos próprios deficientes. Existem outros que auxiliam também pessoas sem deficiência como o Liane TTS que ensina linguagem de sinais. Mas esses softwares são encontrados em locais diferentes, fazendo com que o usuário tenha que baixar de diferentes fornecedores, criando diferentes contas e o progresso gravado em locais diferentes. Uma alternativa para não se ter vários softwares e reunir estas funcionalidades em um único aplicativo, seria utilizando o conceito de componentização para facilitar a inclusão de novos módulos. O conceito de componentização pode ser usado para criar um aplicativo com componentes que possam ser reaproveitados e facilitando a inclusão de novos módulos, tornando assim um aplicativo mais completo.

Outra possível vantagem na utilização do conceito de componentização está relacionado ao desenvolvimento de softwares que está cada vez mais complexo e detalhado por conta de usuários cada vez mais exigentes, sejam elas empresas ou usuários domésticos (RAIM, 2010). Com isso são necessários tempos mais longos para o desenvolvimento e muitas vezes é ultrapassado o tempo estimado, resultando em atrasos e reclamações por parte dos clientes. Para resolver esse problema, alguns desenvolvedores utilizam o conceito de componentização, reutilizando componentes e padronização das regras de negócio, dando mais agilidade no desenvolvimento.

Diante dos argumentos citados, este trabalho se propõe a desenvolver um aplicativo para o auxílio no ensino de pessoas com e sem deficiência visual, possibilitando uma melhor adaptação entre os deficientes visuais e os que convivem com eles. Acreditasse assim que esse projeto possa vir auxiliar a integração dos deficientes ao meio onde vivem,

3 trabalho desenvolveu um

facilitando o aprendizado e adaptação, principalmente no meio educacional. Para isso será utilizado padronização de projetos utilizando conceito de componentização para facilitar a criação de novos módulos.

2 FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo será apresentado a fundamentação teórica, onde o item 2.1 irá tratar dos conceitos usados no desenvolvimento desse trabalho, o item 2.2 será apresentar o protótipo atual desenvolvido por Lucas Cazagrande em 2016 e no item 2.3 apresenta os trabalhos correlatos com o trabalho relatado nesse artigo.

2.1 CONCEITOS

Este capítulo descreve brevemente os assuntos que fundamentarão o estudo a ser realizado: sistema Braille e desenvolvimento de aplicações utilizando componentização.

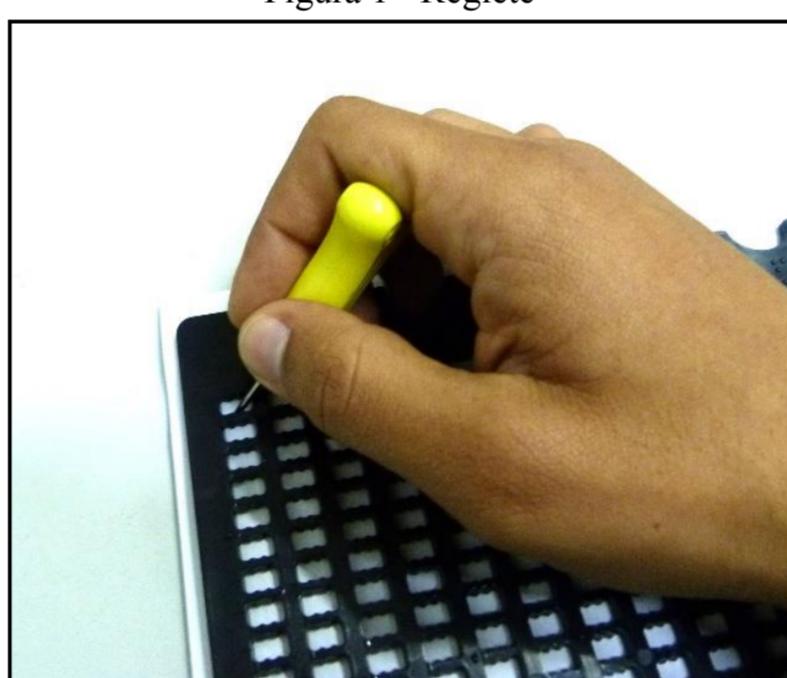
2.1.1 Braille

O Braille é um sistema de escrita e leitura tátil para as pessoas cegas e foi criado por Louis Braille que desenvolveu um sistema de leitura para deficientes visuais (INSTITUTO BENJAMIN CONSTANT, 2018) e se tornou importante para portadores de deficiência visual. O sistema Braille baseados em símbolos de alto-relevo resultantes da combinação de até seis pontos, no que se convencionou chamar de “cela braille”, dispostos em duas colunas de três pontos cada (COSTA, 2009), através desses pontos são representados 63 símbolos (OTSUKA, 2010).

O Braille foi introduzido no Brasil por José Álvares de Azevedo, que havia estudado em Paris e era cego. Em 1854 graças a ele foi fundada no Rio de Janeiro, a primeira escola para pessoas com deficiência visual da América Latina (OLIVEIRA, 2019). O Brasil utilizou o Braille na sua forma original até a década de 1940, quando teve que alterar por conta da reforma Ortográfica da Língua Portuguesa, e ficou sem regra específica até que em 1999 foi criado a Comissão Brasileira do Braille. Em 2002 foi criado a Grafia Braille para a Língua Portuguesa e passou a ser utilizado pelos territórios brasileiros e portugueses (INSTITUTO BENJAMIN CONSTANT, 2018).

Existem diversos instrumentos para a escrita Braille como o reglete (Figura 1), que é acompanhado do punção. A escrita com o reglete é feita da direita para a esquerda, porque as palavras que são lidas no alto-relevo que é formado no papel (CIVIAN, 2017). O reglete ainda é usado apesar de que hoje em dia existem algumas tecnologias com máquina de escrever em Braille, da impressora e de softwares leitores de tela (CIVIAN, 2017).

Figura 1 - Reglete



Fonte: Civian (2017).

Hoje em dia são utilizados *displays* braille que já vêm sendo utilizados em muitos países e resolvem as dificuldades de armazenamento permitindo que pessoas cegas possam ler com mais autonomia e em qualquer lugar. Contudo, o alto custo desses equipamentos ainda os torna inacessíveis para a maioria dos cidadãos (OLIVEIRA, 2019).

2.1.2 Componentização

Um conceito que será usado nesse trabalho que também é usado na indústria de jogos é a de reutilização de componentes, conceituado como a produção de componentes de software que são úteis para mais de um projeto (ROLLINGS; MORRIS, 2003). Segundo, Oliveira (2012), componentização de software é uma abordagem arquitetural baseada na divisão de sistemas de software em unidades menores, denominadas componentes, permitindo a criação de um sistema como se houvesse vários sistemas menores, diminuindo sua complexidade, fazendo com que cada componente seja focado em um conjunto de funcionalidades semelhantes ou em uma função específica, nas quais estas funções podem

1 2009). E através

2 2019 ou 2016

3 Braille. Em

4 2019 ou 2016

5 que foi usado

6 trabalho e que

7 2003 ou 1999

ser reutilizadas através do acesso ao componente (OLIVEIRA, 2012). Existem vários benefícios com uma aplicação baseada em componentes como cita Oliveira (2012).

**Arrumar
(diminuir) os
espaçamento
entre linhas da
citação direta.**

O desenvolvimento de aplicações baseado em componentes nos traz uma série de benefícios, dentre os quais podemos destacar:

- Produtividade: Pode-se economizar tempo de desenvolvimento, dependendo do portfólio de componentes já prontos;
- Robustez: Maior qualidade no produto final que utiliza componentes, pois os mesmos já foram largamente testados em um projeto dedicado à construção dos mesmos;
- Padrão de desenvolvimento: Equipe orientada a desenvolvimento nos moldes da componentização (OLIVEIRA, 2012, p. 1).

2.2 PROJETO ATUAL

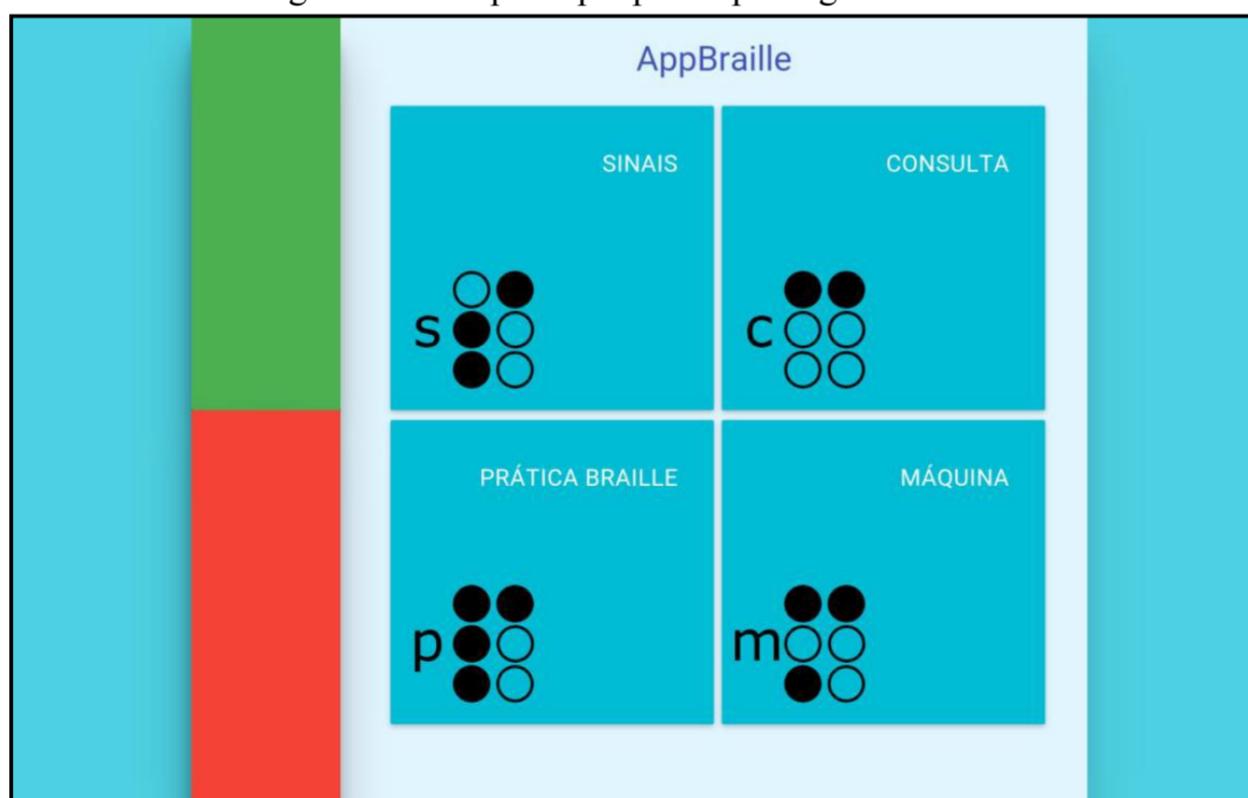
O projeto atual foi desenvolvido por Cazagrande (2016), tinha como objetivo desenvolver um módulo de jogos para o tagarela que é um aplicativo de comunicação alternativa, para auxiliar no aprendizado do sistema Braille por pessoas normovisuais. Para desenvolver o projeto foi utilizado Brackets em conjunto com o PhoneGap e utilizado as linguagens HyperText Markup Language 5 (HTML5), Cascading Style Sheets (CSS) e JavaScript para o desenvolvimento de três opções, sendo acessada a partir da tela principal como mostra a Figura 2.

A primeira opção Sinais foi criada para demonstrar as sete séries de sinais braile, que representando todas as letras e alguns sinais escritos. Para facilitar o entendimento da disposição das séries, quando o usuário clica em alguma das celas é direcionado para um exemplo com a letra ou sinal selecionado.

A opção consulta Braille representa a transição dos pontos braile. Nessa tela tem duas celas dispostas uma ao lado da outra já que um único símbolo escrito pode ser representado por mais de uma cela Braille, como por exemplo os números.

Na terceira opção Prática Braille são apresentadas imagens com a descrição que remete ao significado da imagem, uma cela Braille ao lado e abaixo do nome da imagem, uma descrição com o uso dos sinais Braille. Alguns dos sinais são substituídos por um sinal de interrogação em tinta, assim o usuário tem que informar qual o sinal Braille é referente ao sinal que está faltando.

Figura 2 - Tela principal protótipo Tagarela Braille



Fonte: Cazagrande (2016).

A quarta opção Máquina não foi implementada ficando como sugestão para extensões, que seria um simulador de máquina Braille. Um dos principais pontos fortes desse trabalho foi a divisão dos módulos que deixa bem intuitivo para o usuário e os exemplos com imagens, representando melhor e ajudando a fixar o exemplo que se quer passar. O ponto negativo foi a falta de acessibilidade, através de áudio, gestos na tela ou movimentos do dispositivo, para tornar o aplicativo mais acessível por quem é cego ou tem baixa visão.

2.3 TRABALHOS CORRELATOS

Foram escolhidos três trabalhos correlatos com propostas semelhantes a este. No Quadro 1 descreve o aplicativo Aprende Braille desenvolvido por Ugedo (2016), com exercícios para praticar e aprender Braille. No Quadro 2 relata o trabalho AbcNum Braille (AQUINO *et al.*, 2015), que tem como objetivo desenvolver uma aplicação para auxiliar na aprendizagem do alfabeto Braille para pessoas com baixa visão. Por último, no Quadro 3 o LêBraille (FAÇANHA *et al.*, 2012) um aplicativo que simula os instrumentos utilizados para o aprendizado em Braille.

Quadro 1 - Trabalho correlato Aprende Braille

Referência	Ugedo (2016)
Objetivos	O aplicativo visa auxiliar no aprendizado de Braille através de exercícios
Principais funcionalidades	<ul style="list-style-type: none"> a) alfabeto – lista com o alfabeto e seus respectivos sinais em braile; b) ejercicios de letras – exercícios para o usuário digitar a letra que é representada na célula; c) ejercicios de palabras – exercícios para o usuário tem que digitar a palavra que é representado nas células braile; d) ejercicios de palabras – módulo de exercícios para o usuário digitar o número informado na célula braile; e) ejercicios de frases – Assim como o módulo exercício de palavras, os exercícios com frases para o usuário informar qual frase representa os símbolos em braile.
Ferramentas de desenvolvimento	Não foi encontrada a ferramenta usada para o desenvolvimento.
Resultados e conclusões	O que também auxilia na acessibilidade é o <i>feedback</i> com áudio presente em algumas opções do aplicativo. O aplicativo é prático e intuitivo, com nota de 4,6 na loja de aplicativo e mais de dez mil downloads na <i>play store</i> .

Fonte: Elaborado pelo autor.

1 Ponto final.

Quadro 2 - Trabalho correlato AbcNumBraille

Referência	Aquino <i>et al</i> (2015)
Objetivos	Auxiliar deficientes visuais no processo de alfabetização, possibilitando a prática de atividades de alfabetização.
Principais funcionalidades	<ul style="list-style-type: none"> a) Feedback sonoro; b) Feedback tátil; c) Exercício de vogais; d) Exercício de consoantes; e) Exercícios de números.
Ferramentas de desenvolvimento	Não foi informado as ferramentas usadas no desenvolvimento.

Fonte: Elaborado pelo autor.

2 Ponto final.

3 Quadro 3 - Correlato LêBraille

Referência	Façanha <i>et al</i> . (2012)
Objetivos	O principal objetivo de disponibilizar uma ferramenta de auxílio na alfabetização no sistema Braille
Principais funcionalidades	<ul style="list-style-type: none"> a) O aplicativo dispõe de uma tela com 6 teclas que representam uma cela braile; b) Interação por gestos na tela; c) feedback da interação por áudio.
Ferramentas de desenvolvimento	JAVA
Resultados e conclusões	O aplicativo se mostrou acessível principalmente para pessoas que possuem um grau maior de deficiência visual, como mostrado nos resultados obtidos após a experiência com deficientes visuais (FAÇANHA <i>et al.</i> , 2012). Os pontos mais marcantes foram a naveabilidade e o conteúdo, como relatados pelo que participaram da pesquisa apesar de alguns não demonstrarem segurança no uso do sistema (FAÇANHA <i>et al.</i> , 2012).

Fonte: Elaborado pelo autor.

3 Quadro 3 - Trabalho correlato

4 Ponto final.

5 Ponto final.

6 Linguagem e programação Java.

7 pelos

3 DESCRIÇÃO

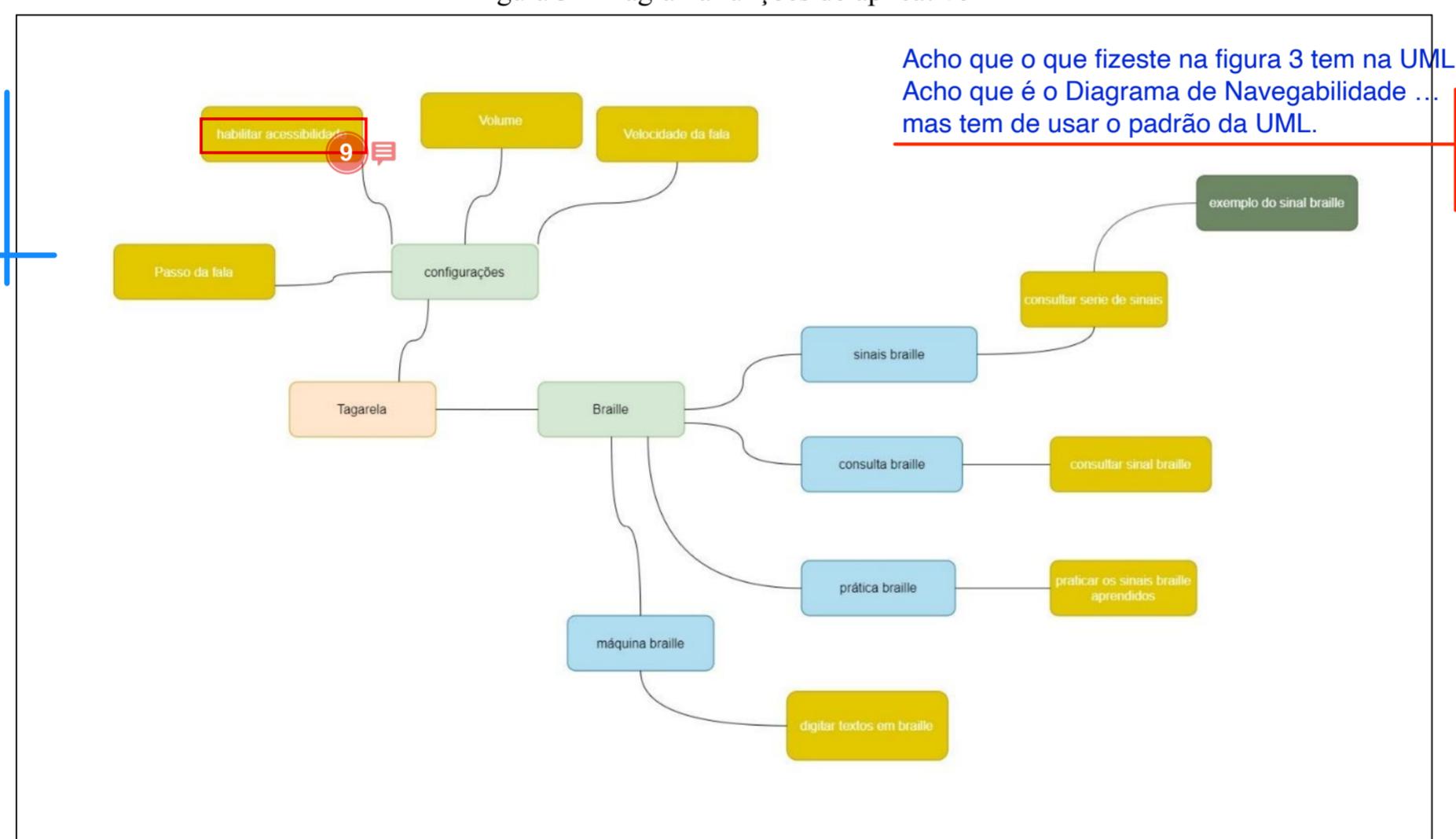
Neste capítulo será descrito detalhes sobre a implementação do aplicativo, na seção 3.1 serão descritos os aspectos mais importantes e uma visão geral do aplicativo, na seção 3.2 tem-se um detalhamento sobre a implementação do aplicativo e ferramentas utilizadas.

Atualmente só se tem o módulo Braille, mas mais módulos serão desenvolvidos, e o módulo Braille servirá como modelo para os próximos módulos.

3.1 VISÃO GERAL DO APLICATIVO

O aplicativo dispõe de uma tela inicial com a opção de Braille que dispõe das funcionalidades principais para auxiliar no aprendizado do Braille como mostra a Figura 3, como o aplicativo tem intenção de que seja desenvolvido mais módulos e o módulo Braille serve como guia para os demais a serem desenvolvidos por isso tem apenas esta opção. Ao iniciar o aplicativo um áudio informa que, se o usuário deseja acessar o módulo Braille deve-se clicar duas vezes na tela, ao entrar no módulo Braille tem-se a tela inicial do Tagarela Braille que será explicado melhor a seguir. Para um melhor entendimento essa especificação será dividida em duas, o modo acessibilidade e as funcionalidades do módulo.

Figura 3 - Diagrama funções do aplicativo



A figura 3 deve ser descrita e citar no texto.

Fonte: Elaborado pelo autor.

3.1.1 Modo acessibilidade

O modo acessibilidade pode ser configurado no menu lateral do aplicativo na opção configurações e por padrão essa configuração é ativada para que pessoas com baixa visão ou cegos possam utilizar o aplicativo sem ajuda externa, no menu existem algumas configurações que podem ser feitas com relação ao áudio do modo acessibilidade, são elas:

- acessibilidade: habilita a função de acessibilidade para uso por pessoas com baixa visão ou cegos;
- volume: essa funcionalidade, assim como as demais configurações, só é usada quando o modo acessibilidade está ativo, essa configuração serve para controlar o volume da voz do modo acessibilidade;
- velocidade: essa configuração é a velocidade com que o áudio é falado;
- passo: é a velocidade do passo das palavras que é falada pelo aplicativo .

A função de acessibilidade foi desenvolvida pensado em usuário que tem pouca visão ou são cegos, pois o feedback de áudio auxilia e possibilita que ele possa utilizar o aplicativo através de gestos na tela. Com essa opção ativa o usuário tem total controle no módulo Braille, com exceção das configurações. Para que o usuário não clique em uma opção por engano todos os botões da tela são desabilitados quando a acessibilidade estiver ativa e a interação com o aplicativo só funcionará através gestos na tela. Sempre que o usuário entrar na tela e o modo acessibilidade estiver ativo é dada as instruções por áudio para que ele possa utilizar as funcionalidades dela. Os gestos detectados pelo aplicativo seguem sempre um padrão e sempre são acompanhados de um áudio da ação tomada. Como pode ser visto no Quadro 4.

- 1 opção Braille
- 2 Braille (Figura 3). omo
- 3 tela. Ao
- 4 módulo
- 5 explicada
- 6 entendimento, a especificação do desenvolvimento deste trabalho será
- 7 duas partes, o
- 8 módulo ou aplicativo
- 9 Ruim para ler.
- 10 Fonte courier.
- 11 configurações. E por
- 12 Neste menu, de configurações, existem alguns ajustes que podem ser feitos, como exemplo: configurações .. fonte courier.
- 13 cegos
- 14 ativo. Essa
- 15 Quadro 4 que lista os gestos e ações disponíveis no aplicativo.

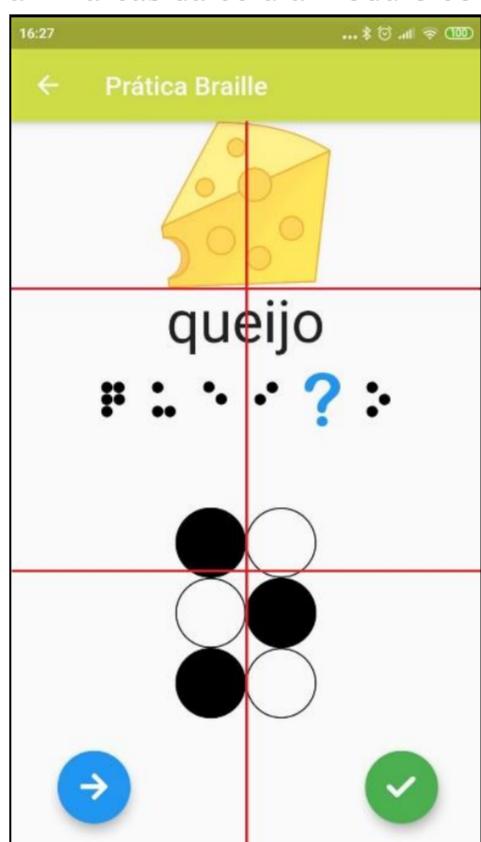
Quadro 4 - Gestos e ações

Gestos	Ações
Manter pressionado	Repete a instrução dada ao entrar na tela.
Deslizar para direita	Navega entre as opções de botões ou opções da tela, sempre seguido de dois cliques na tela para confirmar a opção selecionada.
Deslizar para esquerda	Assim como o deslizar para a direita este gesto navega entre as opções, sendo que no sentido contrário e assim navegando para a opção anterior, caso tenha mais de uma opção.
Um toque na tela	Depende da tela, conforme explicado a seguir.
Dois toques na tela	Quando navegado pelas opções disponíveis e ao clicar duas vezes executa a ação.

Fonte: Elaborado pelo autor.

Em algumas telas como a da opção **pratica** é necessário informar qual ponto **braile** tem deve estar marcado, cada parte da tela representa um ponto na célula braile como mostra a Figura 4 que mostra essa divisão.

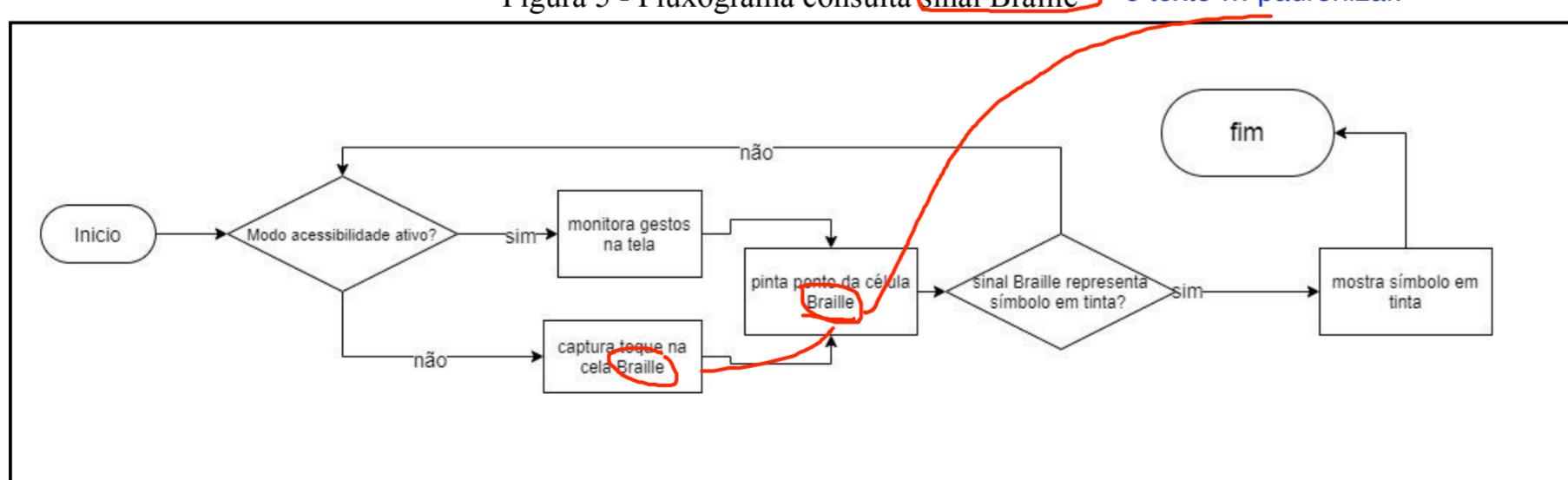
Figura 4 - áreas da célula modulo consulta



Fonte: Elaborado pelo autor.

Sempre que o usuário clica em alguma das áreas divididas na tela, o ponto braile é marcado de acordo com o local da tela clicado, sempre com o retorno de áudio para que o usuário saiba que sinal está sendo digitado, nessa tela ainda o usuário pode navegar entre as duas opções os botões pular palavra e confirmar, para navegar entre essas opções ele tem que deslizar para direita ou esquerda e confirmar com dois cliques na tela. Assim como a tela de prática a de consulta também contém celas braile, a diferença é que ao marcar o ponto a representação do caractere em tinta é mostrado na tela, a Figura 5 exibi o fluxograma da consulta de sinais braile.

Figura 5 - Fluxograma consulta sinal Braille



Fonte: Elaborado pelo autor.

1 Braille (Figura 6).

2 escrito. Cada

3 consultar. Os

4 braile, e em

5 tinta. Para

6 tela. Na

7 representando. Nem

8 consultados, pois
alguns

9 aprendido. Nessa

10 opção, a máquina

11 dois botões (confirmar
e deletar). O usuário

12 deseja usar e

13 Braille. E em símbolo

14 Nesta seção se
pretende apresentar

15 Flutter

16 iOS

17 alterado. Existem

18 outros. Mas

19 Flutter

20 frameworks. Mas

21 projeto modular

22 Apêndice A

23 Apêndice B e Apêndice
C.

24 figma, que é uma

25 móveis. O protótipo

26 As telas do protótipo
estão representadas na
Figura 7, as quais
refletiram exatamente
como ficaram as telas
na aplicação final.

3.1.2 Funcionalidades do aplicativo

O aplicativo tem quatro funcionalidades, são elas: sinais, consulta, prática e máquina Braille como mostra a Figura 6. A opção **sinais** é composto por sete séries de sinais braile, na qual cada sinal representa um símbolo escrito, cada símbolo tem um exemplo em que o usuário pode consultar, os exemplos têm uma figura e um texto escrito em braille e em letra de tinta, para navegar entre as séries de sinais é utilizado os botões na parte inferior da tela para navegar. Na opção **consulta** possui duas celas braile em que o usuário pode consultar os sinais digitando o símbolo em Braille na tela, na parte superior da tela será informada qual letra ou símbolo as células estão representando, nem todos os sinais braile podem ser consultados, pois, alguns poucos símbolos necessitam de mais de duas celas braile. Outra opção disponível é a **prática** que possibilita a prática do que foi aprendido, nessa opção é exibido uma imagem com o nome dela embaixo e uma série de símbolos abaixo com um sinal Braille faltando, para que o usuário possa digitar qual símbolo que está faltando. Na última opção **máquina Braille** é composto por duas caixas de textos, uma cela braile e dois botões, **confirmar** e **deletar**, o usuário digita na cela braile os símbolos que deseja digitar e nas caixas de texto acima é informado o texto em Braille e em símbolo na representação de tinta, o usuário pode até mesmo digitar símbolos compostos, que utilizam mais de uma cela braile, que mesmo assim será reconhecido. No botão **confirmar** o usuário confirma o símbolo que deseja digitar e o botão vermelho com “x” exclui o texto digitado.

Figura 6 - tela inicial tagarela Braille



Poderia colocar uma figura para cada uma das 4 opções para facilitar entender o que estais explicando no texto.

Fonte: Elaborado pelo autor.

3.2 IMPLEMENTAÇÃO

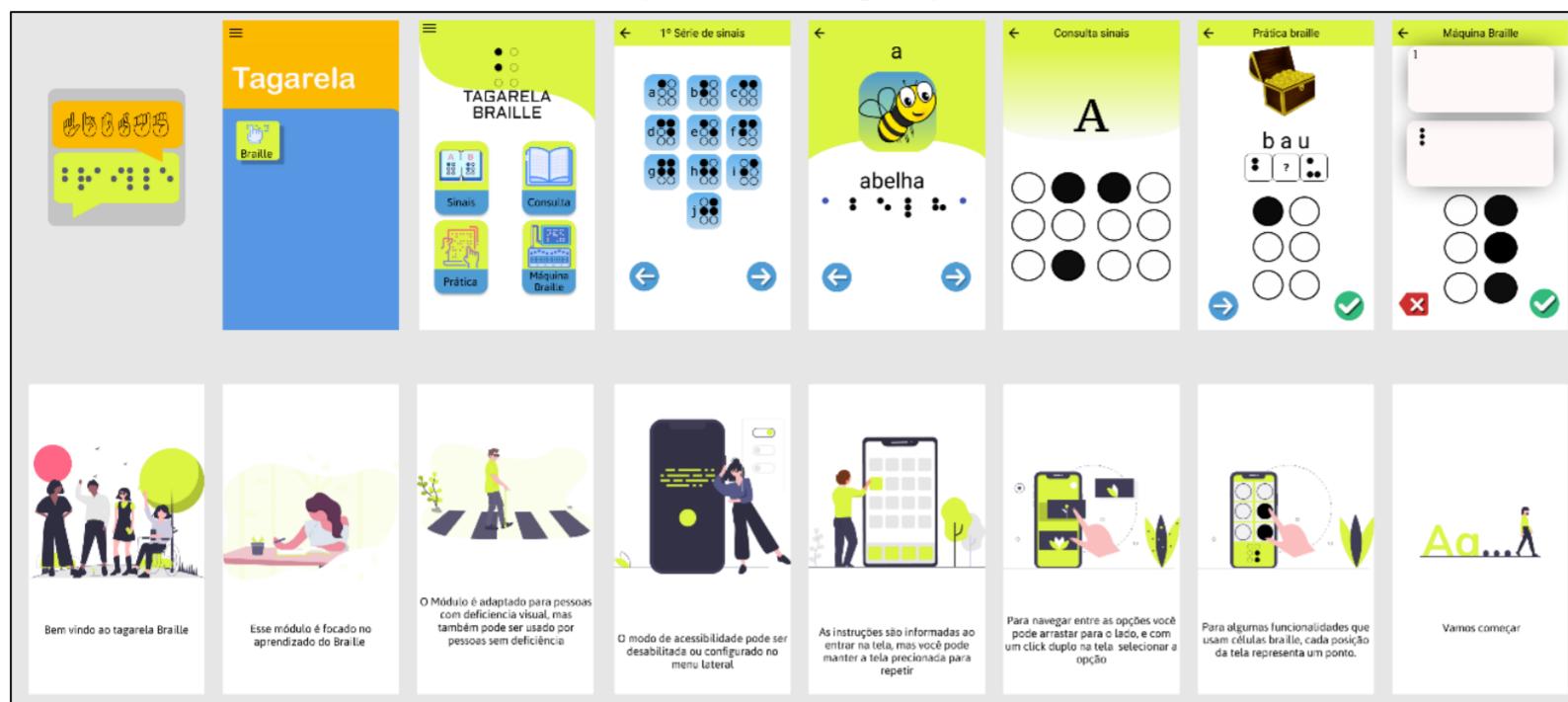
14 Neste paragrafo pretende-se apresentar os detalhes do desenvolvimento e especificações mais relevantes do aplicativo e as ferramentas utilizadas.

3.2.1 Ferramentas utilizadas

15 O projeto foi desenvolvido em **FLUTTER**, um *framework* para desenvolvimento de aplicativos para desktop, Android, **iOS** e WEB lançada em 2017 pela Google e utiliza a linguagem DART. Para a criação de um aplicativo é necessário utilizar um gerenciador de estado, que serve para atualizar a tela quando a variável tem seu valor alterado, existem diversos gerenciadores de estado como o BloC, MobX, GetX, setState, entre outros, mas para esse projeto foi utilizado o MobX. 16 O **FLUTTER** não tem uma estrutura de projetos padrão assim como outros *frameworks*, mas nesse projeto foi utilizado uma estrutura de projeto do modular que serve também para a injeção de dependência, controlar as rotas e modularização. A estrutura do projeto criada pode ser vista no **APÊNDICE A**. Para a criação dos sinais braile foi utilizado o criador de fontes calligraphr que é um criador de fontes on-line, onde todos os caracteres são desenhados e convertido em fonte de texto pela ferramenta como pode ser visto no **APÊNDICE B** e **APÊNDICE C**. Antes de iniciar o desenvolvimento do aplicativo foi criado um protótipo usando o **figma** uma ferramenta web para protótipação de sites e aplicativos mobile, o protótipo ficou como na Figura 7 o resultado ficou igual ao proposto.

O Apêndice C não fala sobre os caracteres?????

Figura 7 - Protótipo figma

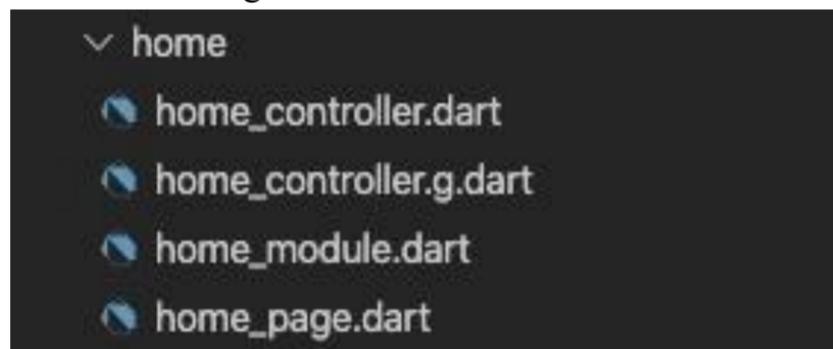


Fonte: Elaborado pelo autor.

3.2.2 Desenvolvimento

O aplicativo inicia no módulo tagarela que é o modulo principal e a home page do aplicativo, todos os módulos são compostos por quatro arquivos como mostra a Figura 8, os arquivos `home_page.dart`, `home_controller.dart`, e `home_module.dart` são padrões de uma estrutura Model View Controller (MVC) e o quarto arquivo é o `home_controller.g.dart`, é um arquivo utilizado pelo gerenciador de estado MobX para controlar a mudança de estado.

Figura 8 - Estrutura módulo.



Fonte: Elaborado pelo autor.

Segundo a divisão dos módulos e componentes ficou como representado na Figura 9, que também mostra as principais classes, essa divisão foi feita para melhorar a organização dos próximos módulos a serem implementados. O aplicativo tem duas classes principais `PalavraBraille` e `Braille`, a classe `PalavraBraille` é a classe de estrutura das palavras usadas em todo o módulo `Braille`, a classe `Braille` contém os métodos principais de conversão do sinal Braille para a letra de tinta e vice versa e todas as palavras braile usadas nas séries de sinais.

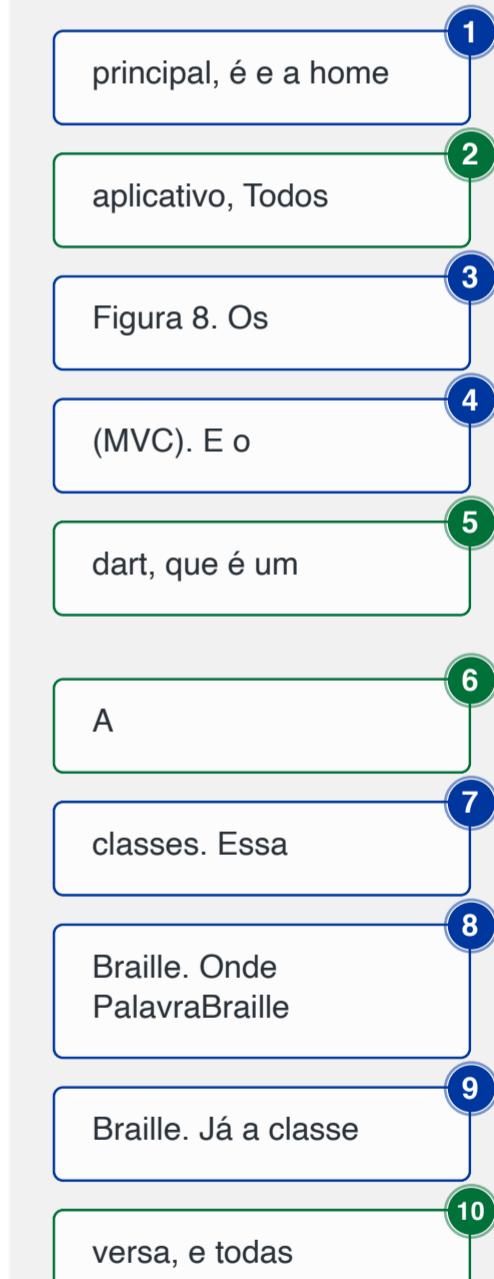
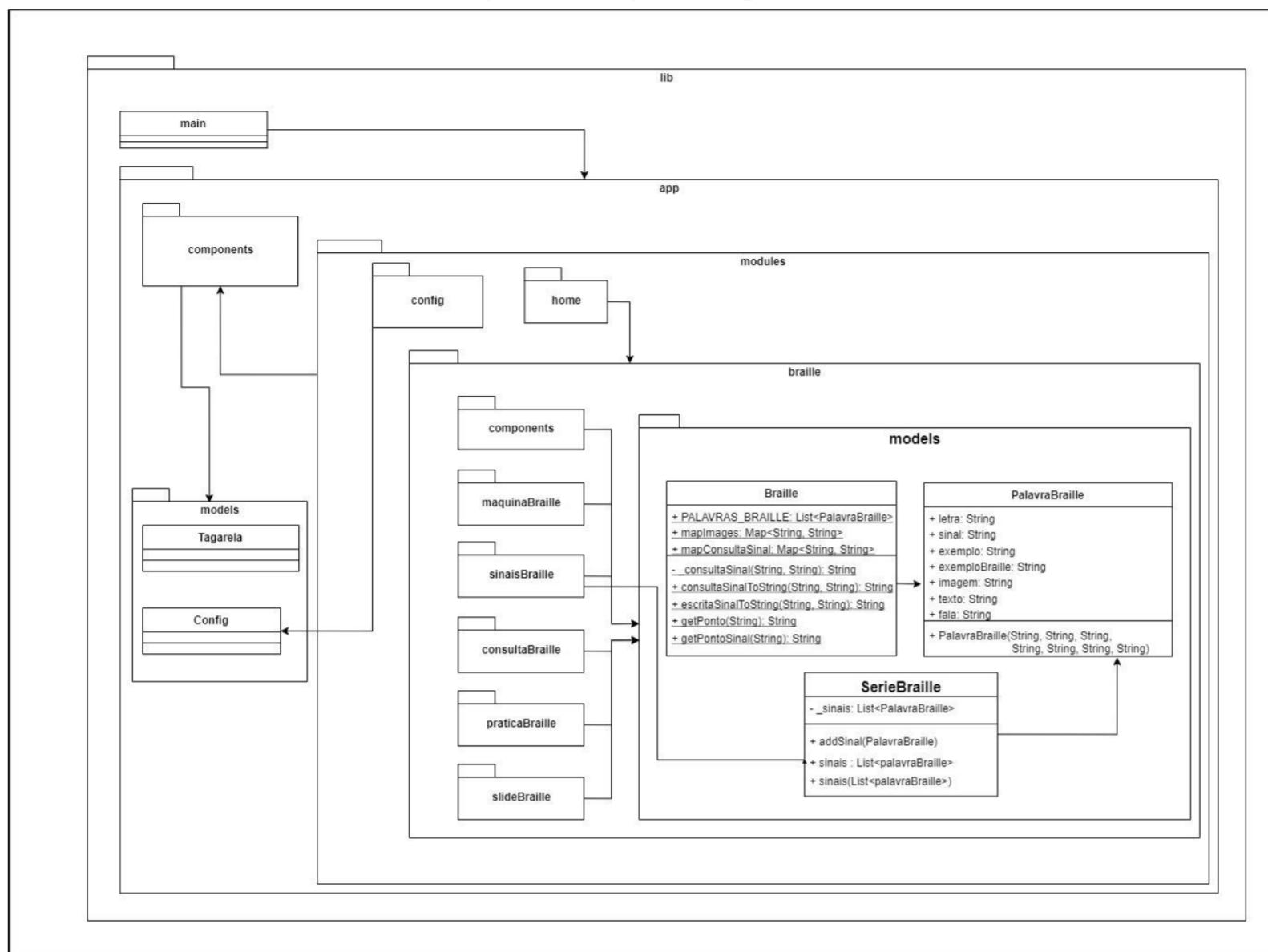


Figura 9 - Diagrama de pacotes



Fonte: Elaborado pelo autor.

O atributo `PALAVRAS_BRAILLE` é uma lista de objetos `palavraBraille` e na tela `sinais` são divididas em sete series de `sinais`, para isso na classe `SinaisBrailleController` é criado uma lista com sete `SerieBraille` e percorrida na página usando o componente `SinaisBrailleWidget` para mostrar os `sinais` e ao clicar em um destes é exibido um exemplo para aquele sinal selecionado. Para os `sinais` braile foram criadas duas fontes específicas para esse projeto como mostra no APÊNDICE B, uma fonte para as celas braile exibidos na tela de `sinais` e outra para os todos os outros `sinais` braile, isso para que tenha uma melhor visualização dos `sinais` na tela de `sinais`, como nem todos os `sinais` braile poderiam ter uma representação foi utilizado alguns caracteres especiais para representar certos `sinais` braile, isso principalmente opção máquina braile, em que o usuário pode digitar qualquer sinal e o texto em tinta é mostrado.

Os métodos `consultaSinalToString` e `escritaSinalToString` possuem a mesma funcionalidade, convertem sinal braile para um caractere, para isso usam o método privado `_consultaSinal` esse método recebe por

- 1 sinais. Para
- 2 sinais. E ao
- 3 Apêndice B. Uma
- 4 braile. Isso
- 5 sinais. Como
- 6 braile. Principalmente na opção
- 7 caractere, e para
- 8 _consultaSinal. Esse

parâmetro dois sinais braile em binário, por exemplo o símbolo “a” em braile seria o ponto 1 da cela braile marcado e todos os outros apagados como mostra a Figura 10.

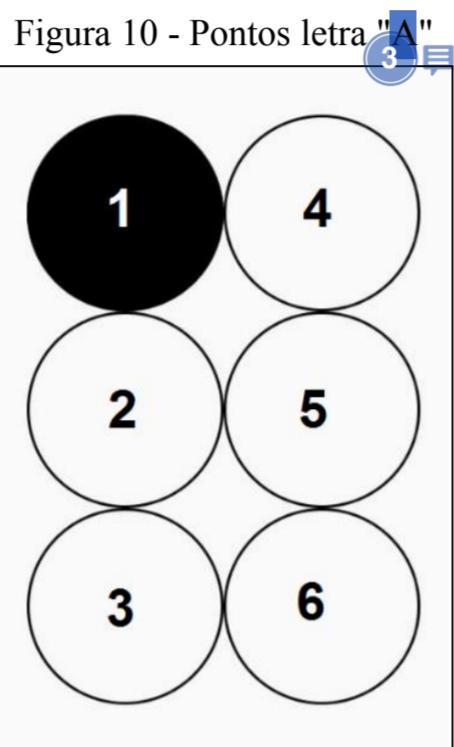


Figura 10 - Pontos letra "A"

Fonte: elaborado pelo autor.

Para isso é convertido todos os pontos em representação binária 0 ou 1, quando 1 o ponto está marcado e 0 apagado, então a letra “a” seria 100000 onde os três primeiros zeros seriam a primeira coluna da cela braile e os outros três seriam a segunda coluna, como alguns sinais precisam de mais de uma cela, então é usado duas celas para converter em braile. A conversão é feita através do map `mapConsultaSinal` na qual a chave é o sinal braile em binário e o valor é o caractere como mostra o Quadro 5.

Quadro 5 - Map Pontos Braille

```

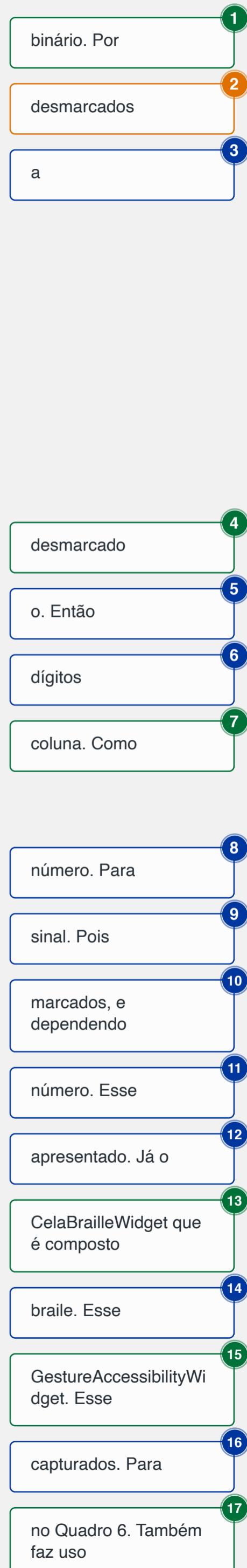
1 static Map<String, String> mapConsultaSinal = {
2     '001111' + '100000': '1',
3     '001111' + '110000': '2',
4     '001111' + '100100': '3',
5     '001111' + '100110': '4',
6     '001111' + '100010': '5',
7     '001111' + '110100': '6',
8     '001111' + '110110': '7',
9     '001111' + '110010': '8',
10    '001111' + '010100': '9',
11    '001111' + '010110': '0',
12    '100000' + '000000': 'a',
13    '110000' + '000000': 'b',
14    '100100' + '000000': 'c',

```

Fonte: Elaborado pelo autor.

A diferença entre os métodos `consultaSinalToString` e `escritaSinalToString` é que o `consultaSinalToString` trata sinais que não são escritos como, por exemplo o sinal que indica um número, para isso são usadas constantes que em todo o aplicativo representam esse sinal, pois para representar um número é utilizado uma cela com os pontos 3,4,5 e 6 marcados e dependendo da próxima cela representar o número, esse método é utilizado na tela de consulta em que deve mostrar o símbolo independente do sinal apresentado, o `escritaSinalToString` é usado na tela de prática e máquina Braille. Para utilizar esses métodos foi criado um componente `CelaBrailleWidget` esse componente é composto por seis pontos formando uma cela braille, esse componente retorna em binários quais pontos estão marcados, e assim podem ser usados pelos métodos de conversão descritos acima.

O componente principal para acessibilidade do aplicativo é o `GestureAccessibilityWidget`, esse componente captura os gestos na tela e executa a ação dependendo dos gestos capturados, para isso ele utiliza do componente `GestureTagarelaWidget` que através dos pontos x e y da posição dos gestos capturados na tela, retorna um enum que representa a posição como pode ser visto no trecho de código no Quadro 6, para isso faz-se uso do componente `GestureDetector`, que executa um método sempre que um toque na tela é detectado e retorna os pontos x e y.



Quadro 6 - Método getPosition

```
1 PositionTap getPosition(double x, y) {
2     Size size = MediaQuery.of(context).size;
3     if (x < size.width * .5 && y < size.height * .33) {
4         return PositionTap.leftTop;
5     } else if (x > size.width * .5 && y < size.height * .33) {
6         return PositionTap.rightTop;
7     } else if (x < size.width * .5 && y < size.height * .66) {
8         return PositionTap.leftCenter;
9     } else if (x > size.width * .5 && y < size.height * .66) {
10        return PositionTap.rightCenter;
11    } else if (x < size.width * .5) {
12        return PositionTap.leftButton;
13    } else
14        return PositionTap.rightButton;
15 }
```

Fonte: Elaborado pelo autor.

Para reproduzir os textos de forma dinâmica é usado a classe `Tagarela` que utiliza de um pacote `flutter_tts` que ler e reproduz os textos nativamente, alguns métodos foram criados para padronizar e reutilizar essa funcionalidade em todo o aplicativo, através do método `speak`¹ os textos são lidos e reproduzidos, todos os controles de reprodução como controle do volume ² são feitos na própria classe.

A acessibilidade do aplicativo foi possível usando o componente GestureAccessibilityWidget e a classe Tagarela, as instruções iniciais e as ações a serem tomadas são passados por parâmetros no momento de criação do componente (Quadro 7), internamente o componente utiliza o método speak para reproduzir os textos passados por parâmetros ou os retornados ao executar o método onTap e assim retornando o texto a ser reproduzido, as ações tomadas ao deslizar na tela são passadas por parâmetros através de uma lista de objetos OptionGesture que contém a ação a ser executada e o texto a ser reproduzido.

Quadro 7 - Uso componentes acessibilidade

```
1 GestureAccessibilityWidget(
2   active: Tagarela.config.accessible,
3   onTap: (position) {
4     String pontos1 = Braille.getPontoSinal(controller.sinal1);
5     String pontos2 = Braille.getPontoSinal(controller.sinal2);
6     return '${(pontos1 != '' ? 'Pontos primeira célula, ' + pontos1 : '')} ${(pontos2 != '' ? 'Pontos segunda célula, ' + pontos2 : '')}' ${(controller.letra != '' ? ', Letra: ' + controller.letra : '')';
7   },
8   options: [
9     OptionGesture(
10       action: () {
11         cela = 1;
12       },
13       speak: 'Célula Braille 1, click duas vezes para confirmar',
14     OptionGesture(
15       action: () {
16         cela = 2;
17       },
18       speak: 'Célula Braille 2, click duas vezes para confirmar'
19     ],
20   primarySpeak:
21     'Consulta sinais braille, \n click na tela para marcar um ponto, \n
22     pontos 1 e 4 na parte superior, \n' +
23     ' pontos 2 e 5 no centro e pontos 3 e 6 na parte inferior. \n
      para navegar entre as células Braille arraste para o lado',
24 )
25 
```

Fonte: Elaborado pelo autor.

Para agilizar o desenvolvimento de novas opções e funcionalidades foi usado a estrutura de módulos, em que cada módulo tem seus módulos filhos e componentes padrões apenas para o módulo específico ou para o filho, como é o caso do CardLetraBrailleWidget que foi feito apenas para o módulo Braille, mas existem também componentes que podem ser usados em todo o aplicativo como é o caso BoxTextWidget que é o componente de caixa de texto existente na tela de máquina Braille, isso foi feito para que os componentes possam ser reaproveitados e assim possibilitem uma padronização e agilizem o desenvolvimento de novos módulos para o aplicativo. Os componentes criados são sempre dinâmicos possibilitando assim o seu reuso, assim podendo ser utilizado.

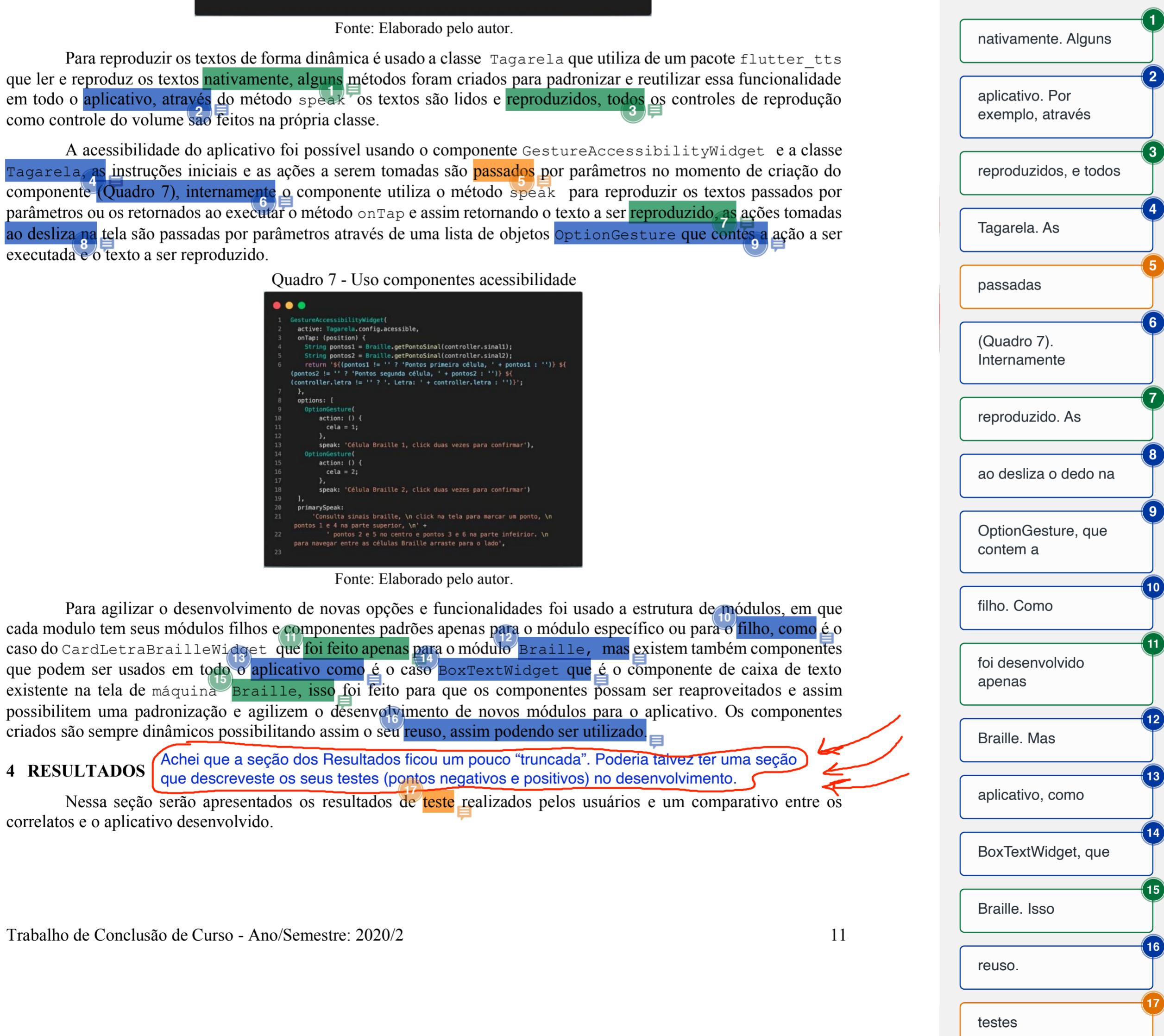
4 RESULTADOS

Achei que a seção dos Resultados ficou um pouco “truncada”. Poderia talvez ter uma seção que descreveste os seus testes (pontos negativos e positivos) no desenvolvimento.

Nessa seção serão apresentados os resultados de teste realizados pelos usuários e um comparativo entre os correlatos e o aplicativo desenvolvido.

Trabalho de Conclusão de Curso - Ano/Semestre: 2020/2

11



4.1 COMPARATIVO COM CORRELATOS

Em comparação com os trabalhos correlatos, o aplicativo se mostrou mais completo e ainda sendo multiplataforma, proporcionando o uso por um número maior de usuários. Como pode ser observado no Quadro 8.

Quadro 8 - Comparativo com os correlatos

	Aprende Braille (UGEDO, 2016)	AbcNum (AQUINO, W. et al. 2015)	LêBraille (FAÇANHA et al. 2012)	Tagarela Braille
Plataforma	Android	Android	Android	Android, iOS
Feedback com áudio	Sim	Sim	Sim	Sim
Captura de gestos	Não	Não	Sim	Sim
Ensino de escrita	Não	Sim	Sim	Sim
Ensino de Leitura	Sim	Sim	Não	Sim
Consultar sinais Braille	Sim	Não	Não	Sim

Aumenta um pouco a largura desta coluna até a última linha ocupar só uma linha.

Fonte: Elaborado pelo autor.

Diminui um pouco a largura destas duas colunas até que as citações na primeira linha fiquem na segunda linha.

4.2 TESTE DE UTILIZAÇÃO

Para coleta de dados foi utilizado um formulário, que pode ser visto no APÊNDICE D, onde os usuários teriam que instalar o aplicativo e executar funções básicas, para usuários sem deficiência visual foi encontrado grande dificuldade de executar as funções com o modo acessibilidade ativo, então todos as respostas foram com o modo acessibilidade desabilitados, foram testados com 10 usuário com faixa etária entre 19 e 57 anos, na qual 90% desses conheciam, mas não sabiam ler braille e 10% nunca ouviu falar, 80% consideraram que o aplicativo tem boa usabilidade e 10% considerou a usabilidade regular ou péssima tanto para pessoas com deficiência como para pessoas sem deficiência, acreditasse que essa nota baixa devesse ao fato de que o modo acessibilidade é ativado por padrão e alguns usuários não conseguiram usar o aplicativo normalmente, isso poderia ser solucionado com uma melhor instrução aos usuários, por exemplo um guia de primeiros passos.

Se remover o Apêndice C, este será o C.

5 CONCLUSÕES

O presente trabalho apresentou um aplicativo para o auxílio no aprendizado do braille utilizando o conceito de componentização para incentivar a criação de novos módulos, ao longo deste artigo foi descrito as funcionalidades, implementação e ferramentas utilizadas.

O Aplicativo desenvolvido para dispositivos Android e iOS foi desenvolvida com FLUTTER e foi postado nas lojas de aplicativos app store e play store, ambas as lojas de aplicativos foram aprovados e já estão disponíveis para downloads. Inicialmente foi pensado em utilizar os mesmos métodos utilizados no projeto do Lucas Cazagrande em 2016, como o de utilização de imagens para representar os sinais braille, mas tendo em vista que a utilização de imagens aumentaria o tamanho do aplicativo e o intuito é que sejam adicionados novos módulos o tamanho do aplicativo pode aumentar muito, então foi criado as próprias fontes de texto, pois com elas além de tornar o aplicativo mais leve deixou mais fácil e rápido o desenvolvimento. Assim como as fontes os áudios também foi pensado na possibilidade de usar áudios pré-gravados, mas seria muito mais difícil de fazer com que os textos dinâmicos tivessem o resultado esperado, além disso também foi pensado em usar API's externas para realizar a leitura dos textos, mas assim o modo acessibilidade só funcionaria on-line, então a melhor solução foi usar o pacote flutter_tts, a desvantagem desse pacote é que a versão mínima do android suportado é a 5.0, mas como de acordo com a própria Google developer (2020) 94,1% dos usuários usam a versão 5.0 ou superior, isso não se tornou um grande problema.

Uma das desvantagens de utilizar o FLUTTER é que por ser um framework relativamente novo, apesar de ser multiplataforma ainda não está com a versão para web disponível para desenvolvimento em produção, mas isso não impede que futuramente o aplicativo possa estar disponível em versões web, e assim tornando ainda mais acessível.

5.1 EXTENSÕES E MELHORIAS

Como extensões e melhorias para trabalhos futuros sugere-se:

- criar *login* para o aplicativo salvar o progresso dos usuários;
- incluir novos módulos para o aplicativo tagarela;
- criar funcionalidade de conversão de texto do alfabeto escrito para o Braille;
- migração para da versão web;
- criação de novos exercícios para a prática Braille;
- tornar as configurações do aplicativo acessíveis;

Este conteúdo descrito na conclusão é o que deverias descrever nos Resultados na parte que faz os comentários como desenvolvedor. A conclusão tem de ser um resumo (sintetize) de tudo o que foi descrito nos Resultados.

iOS
formulário que
Apêndice D. Este formulário instruía os usuários a instalarem o básicas. Para os desabilitados. Os testes foram realizados com usuários na falar. Destes usuários 80% consideraram deficiência. Acreditasse péssima, tanto normalmente. Isso aplicativo que pode auxiliar módulos. Ao longo utilizadas no desenvolvimento deste aplicativo iOS usando o framework Flutter, e foi Aplicativo foi desenvolvido para os dispositivos App Store e Play Store. Ambas lojas já aprovaram o aplicativo, e o mesmo em 2016 (ver seção 2.2). braile. Mas aplicativo, e o intuito módulos, o tamanho muito. Então além de diminuir o tamanho do aplicativo, rápido o seu desenvolvimento esperado. Além textos. Mas modo de flutter_tts. A desvantagem Android superior, e assim isso Flutter Itálico Confirmar com TCC2, mas não se cria seção para extensões ... é iunto com as Tagarela para versão Ponto final.

REFERÊNCIAS

AQUINO, Wermeson *et al.* AbcNum Braille: Proposta de um Aplicativo para Auxiliar no Aprendizado do Alfabeto Braille para Pessoas com Baixa Visão. In: Simpósio Brasileiro de Informática na Educação - SBIE, XXVI. 2015, **Anais...** Disponível em: <https://br-ie.org/pub/index.php/sbie/article/view/5372/3733>. Acesso em: 27 nov. 2020.

BRASIL. Lei nº 7853, de 24 de outubro de 1989. Dispõe sobre o apoio às pessoas portadoras de deficiência, sua integração social, sobre a Coordenadoria Nacional para Integração da Pessoa Portadora de Deficiência. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/L7853.htm. Acesso em: 27 nov. 2020.

CAZAGRANDA, Lucas. **aprendendo braille:** o ensino do sistema braille com o uso do tagarela. 2016. 58 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

CIVIAN. **Você sabe o que é reglete?**, [2017?]. Disponível em: <https://www.civiam.com.br/blog/voce-sabe-o-que-e-reglete/>. Acesso em: 27 nov. 2020.

COSTA, Renata. **Como funciona o sistema Braille?**, [2009]. Disponível em: <https://novaescola.org.br/conteudo/397/como-funciona-sistema-braille>. Acesso em: 27 nov. 2020.

FAÇANHA, Agebson *et al.* Auxiliando o Processo de Ensino-Aprendizagem do Braille Através de Dispositivos Touch Screen. **Informática na educação: teoria & prática**. Porto Alegre, v.15, n.2, jul./dez. 2012

IBGE (Org.). **Conheça o Brasil - População: PESSOAS COM DEFICIÊNCIA**. [2017?]. Disponível em: <https://educa.ibge.gov.br/jovens/conheca-o-brasil/populacao/20551-pessoas-com-deficiencia.html>. Acesso em: 28 nov. 2020.

INSTITUTO BENJAMIN CONSTANT. **O Sistema Braille**. [2018]. Disponível em: http://www.ibc.gov.br/index.php?option=com_content&view=article&id=675:o-sistema-braille&catid=121&Itemid=373. Acesso em: 27 nov. 2020.

MARTA GIL(Org.). **Caderno da tv escola: deficiência visual**. Brasília: Secretaria de Educação, 2000. 80 p.

MASINI, Elcie F. Salzano; GASPERETTO, Maria Elisabete Rodrigues Freire (Org.). **Visão subnormal: Um enfoque educacional**. São Paulo: Votor, 2007.

OLIVEIRA, Regina. **Braille nos dias de hoje: objeto de vitrine ou ferramenta indispensável?**. 2016. Disponível em: <https://www.fundacaodorina.org.br/blog/braille-nos-dias-de-hoje-objeto-de-vitrine-ou-ferramenta-indispensavel/>. Acesso em: 28 nov. 2020.

OLIVEIRA, Renato Gonçalves de (Ed.). **Desenvolvimento baseado em componentes**: Revista Java Magazine 110. 2012. Disponível em: <https://www.devmedia.com.br/desenvolvimento-baseado-em-componentes-revista-java-magazine-110/26550>. Acesso em: 08 set. 2019.

OTSUKA, Daniela. **Braille** 2010. Disponível em: <https://www.infoescola.com/portugues/braile/>. Acesso em: 29 nov. 2020.

RAIM, Samyr Abdo Nunes. **Os 5 grandes desafios no processo de Desenvolvimento de Software**. 2010. Disponível em: https://www.oficinadanet.com.br/artigo/desenvolvimento/os_desafios_no_processo_de_desenvolvimento_de_software. Acesso em: 29 nov. 2020.

ROLLINGS, Andrew; MORRIS, Dave. **Game Architecture and Design**: A New Edition. Indiana: New Riders, 1999. p. 742

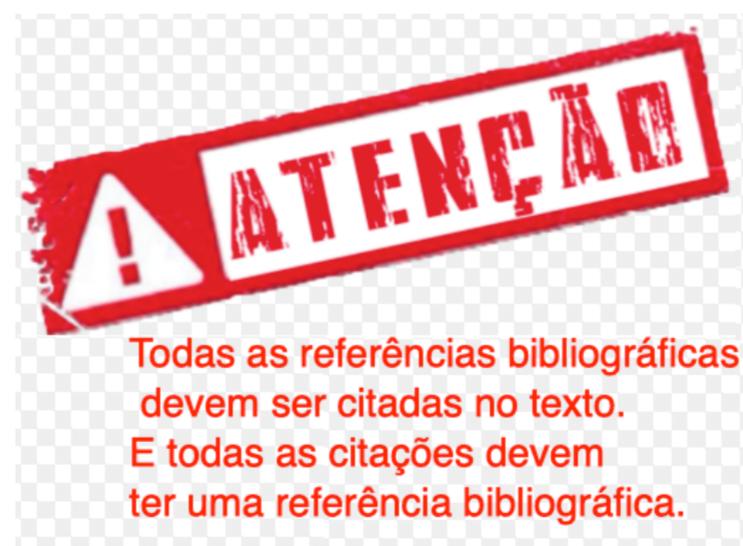
UGEDO, Mario. **Aprende Braille**. 2016. Disponível em: <https://play.google.com/store/apps/details?id=com.comoelagua.android.braille>. Acesso em: 17 nov. 2020.

Google Developers. **Painel de distribuição** [2020]. Disponível em: <https://developer.android.com/about/versions/android-5.0.html>. Acesso em: 29 nov. 2020

As referências
devem estar em
ordem alfabética.



Na versão final do artigo verificar se
quadros, figuras, tabelas, legendas, fontes
não se separam nas quebras de páginas

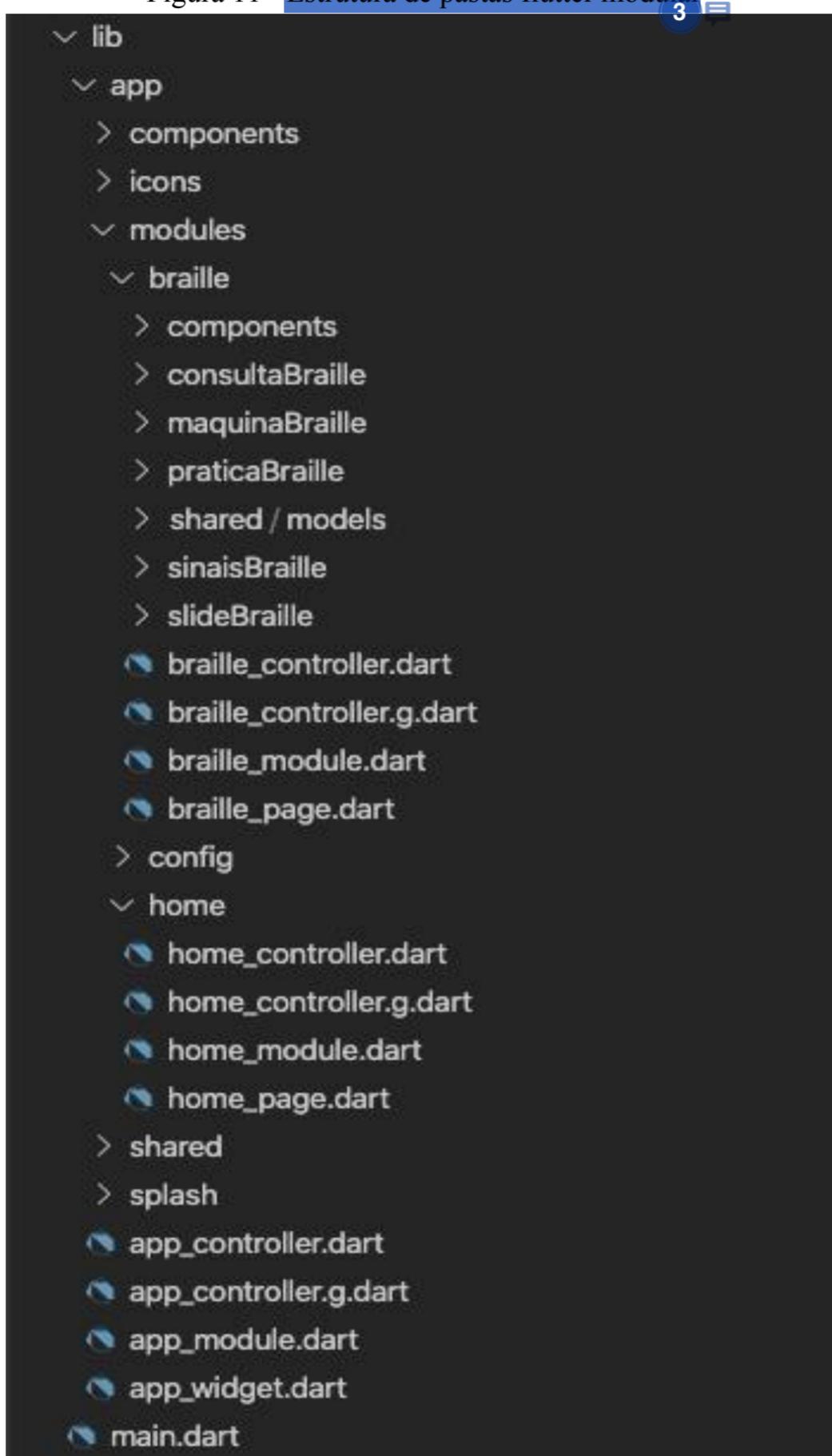


Todas as referências bibliográficas
devem ser citadas no texto.
E todas as citações devem
ter uma referência bibliográfica.

APÊNDICE A – ESTRUTURA DE PASTAS FLUTTER MODULAR

Na Figura 11 apresenta a estrutura de pastas utilizada pelo FLUTTER modular.

Figura 11 - Estrutura de pastas flutter modular



ESTRUTURA
MODULAR DE
PASTAS DO FLUTTER

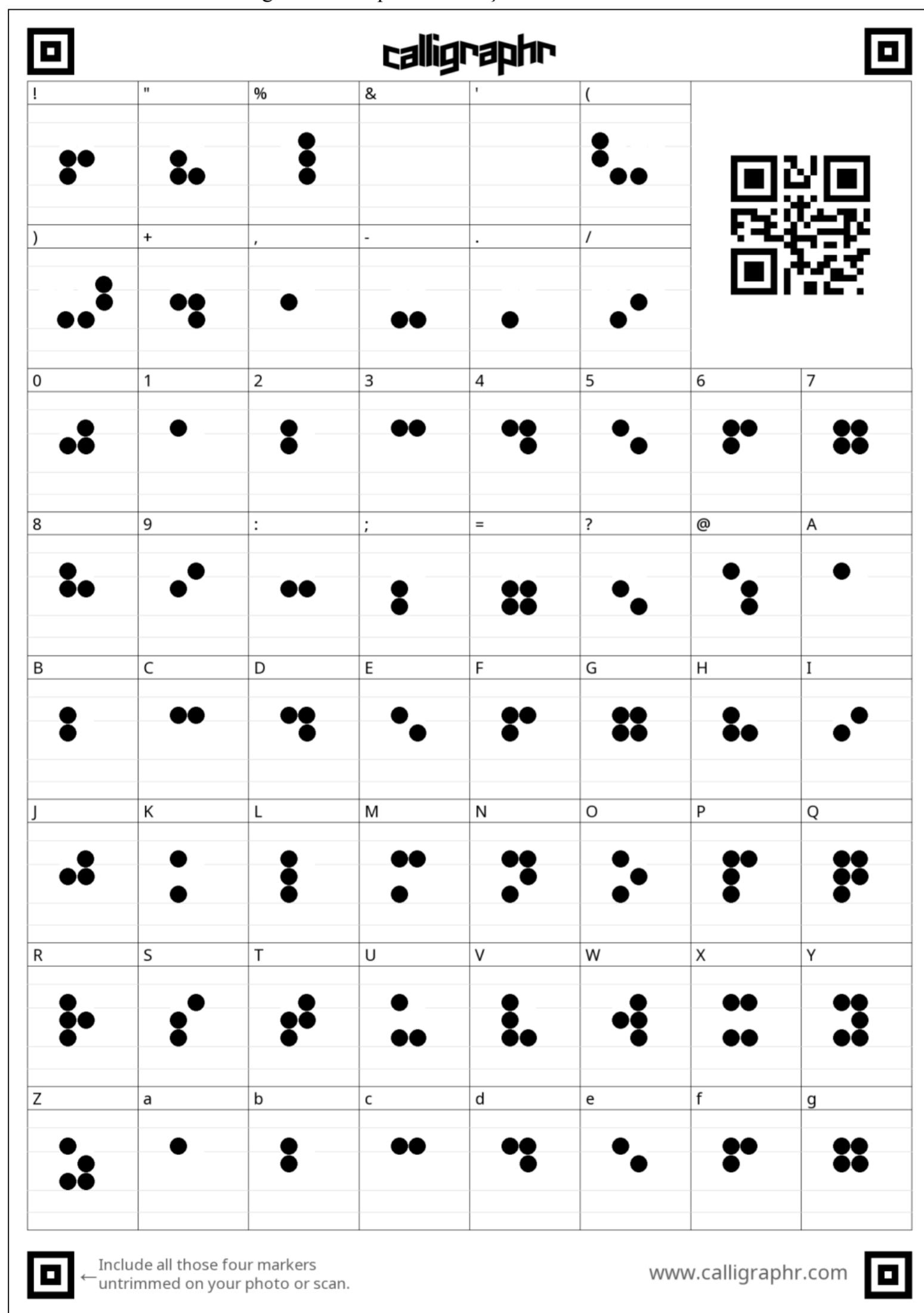
apresenta a estrutura
modular de pastas
utilizada pelo
FLUTTER.

Estrutura modular de
pastas do flutter

APÊNDICE B – ARQUIVO FONTES CALLIGRAPHR

Neste apêndice são apresentados alguns dos arquivos de fontes utilizados para criação de fontes sinais Braille. Na Figura 12 mostra o arquivo que foi utilizado para criar a fonte sem borda utilizada em todo o aplicativo e na Figura 13 o arquivo utilizado para criação da fonte utilizada na tela de sinal.

Figura 12 - Arquivo de criação de fonte sem borda



Fonte: Elaborado pelo autor.

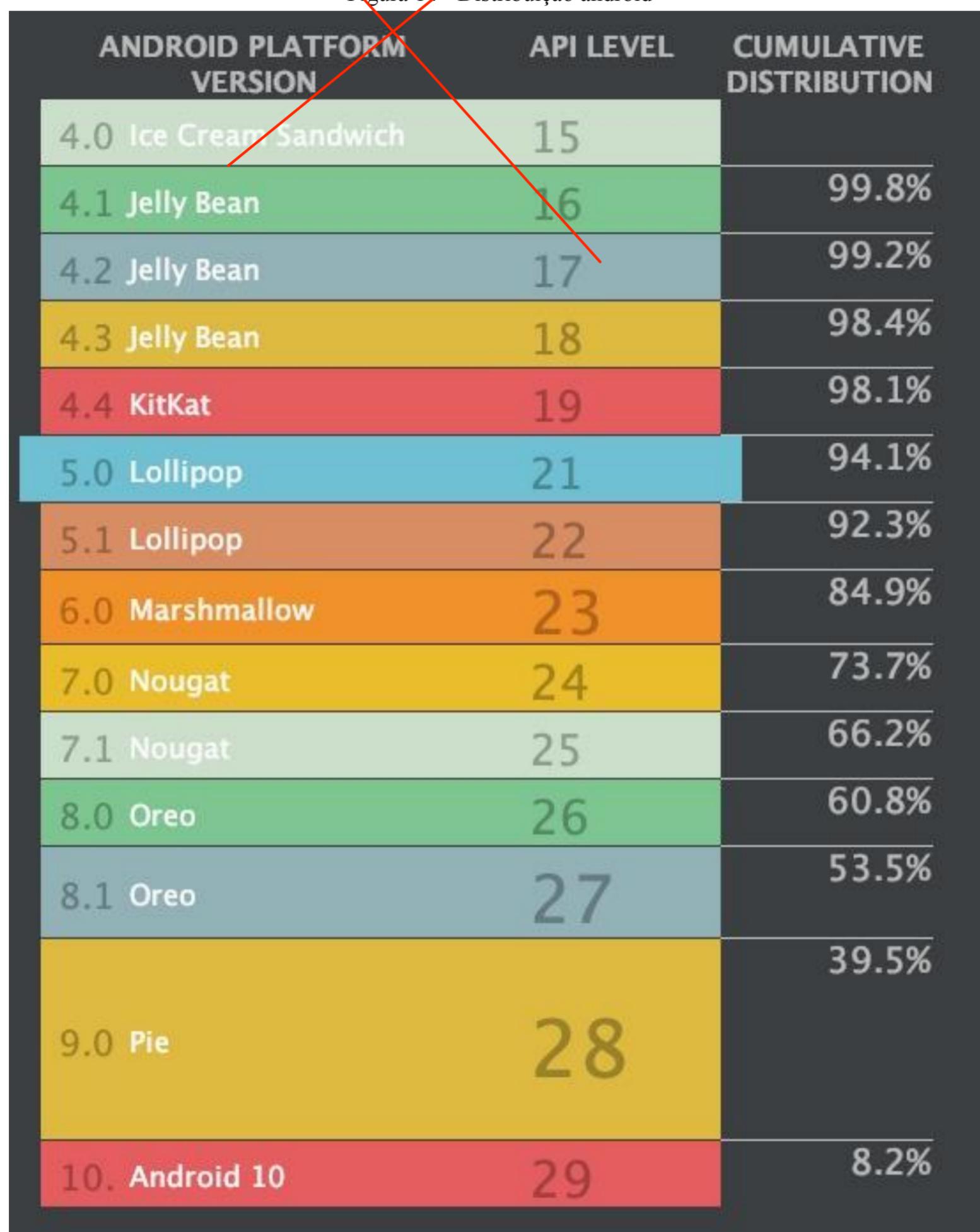
- 1 dos arquivos utilizados para
- 2 das fontes dos sinais
- 3 Braille. A
- 4 aplicativo. Já a Figura
- 5 fonte com borda utilizada

Acho que não faz sentido ter este Apêndice!!!

APÊNDICE C - DISTRIBUIÇÃO ANDROID

As estatísticas de distribuição do android na Figura 14, estão disponíveis apenas pelo android studio e mostra a quantidade de dispositivos que usam determinada versão do android.

Figura 14 - Distribuição android



Fonte: Google Developers (2020).

APÊNDICE D – QUESTIONÁRIO*Se remover o Apêndice C este passa a ser o C*

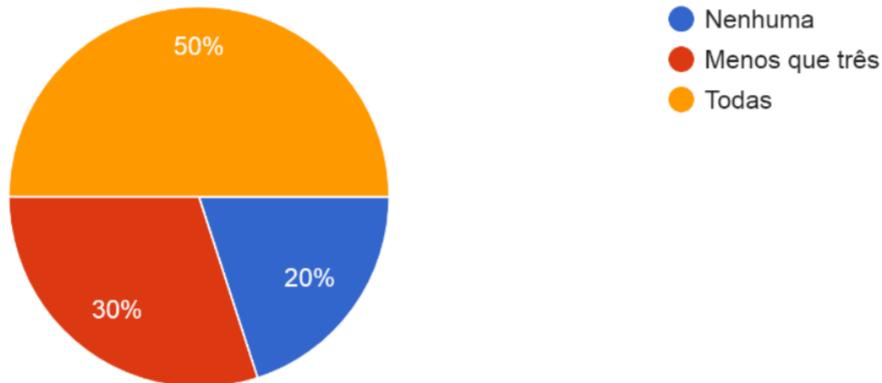
Estas são os pontos mais importantes do questionário, dos usuários que testaram o aplicativo 90% se diziam ter ouvido falar sobre o braile e 10% nunca ouviu falar, como mostra o gráfico na Figura 16. Foi solicitado aos usuários que executassem algumas ações no aplicativo e na Figura 15 mostra quantos usuários conseguiram concluir todas essas atividades, posteriormente foi solicitado que classificassem a usabilidade do aplicativo, para isso foi usada uma escala de 1 a 5 onde 1 é péssimo e 5 excelente, as respostas podem ser observadas na Figura 17. Foi solicitado também que classificassem o grau de estímulo que o aplicativo proporcionava ao aprendizado do braile por pessoas com e sem deficiência visual. Figura 18.

Figura 15 - Tarefas concluídas

Atenção: arrumar este texto.

Quantas tarefas você concluiu sem NENHUM auxílio externo?

10 respostas



Fonte: Elaborado pelo autor.

Figura 16 - Conhecimento braile

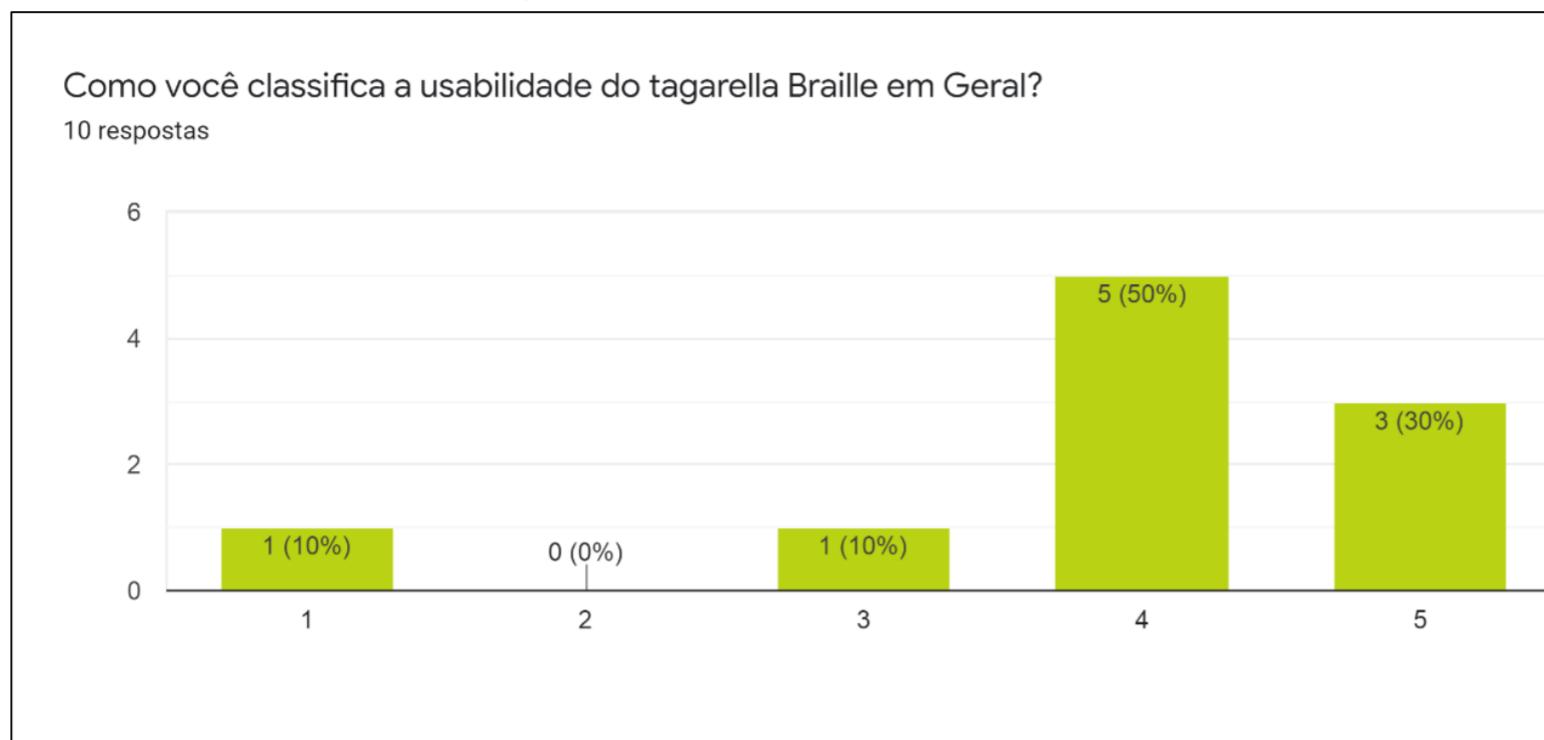
Você conhece Braille?

10 respostas



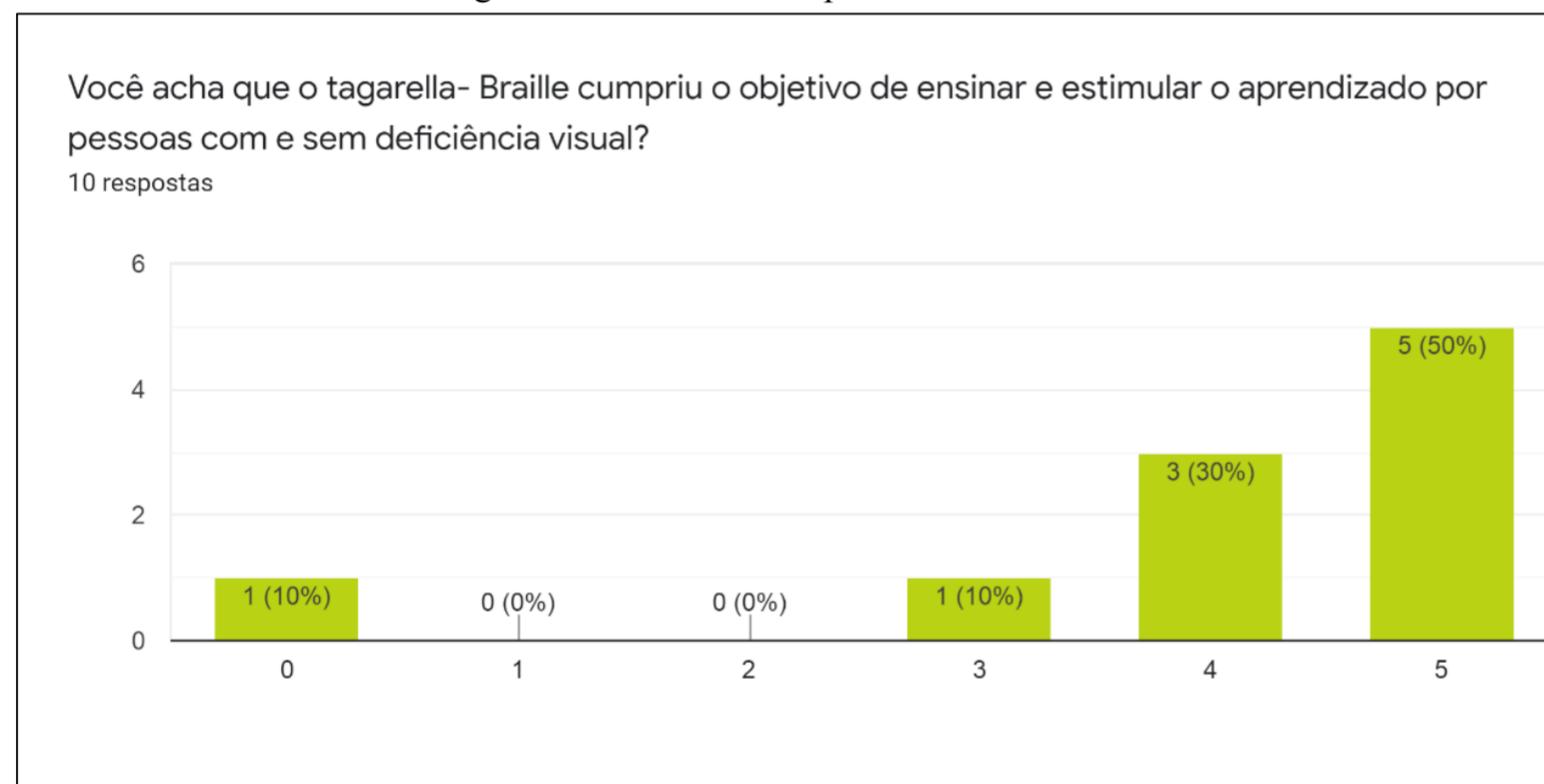
Fonte: Elaborado pelo autor.

Figura 17 - Classificação de usabilidade



Fonte: Elaborado pelo autor.

Figura 18 - Estímulo ao aprendizado do braile



Fonte: Elaborado pelo autor.

