

## Développement Logiciel Libre 2023

Je souhaitais d'abord commencer par des issues en rapport avec le Développement Mobile, et plus précisément en JAVA. Parmi les gros projets de développement mobile, j'ai choisi Simple Mobile Tools ([Simple FlashLight](#) , [Simple Clock](#)) et [Bike Computer](#) ainsi que [CoinTrend](#).

### Conventions et règles de contribution:

Pour toutes les issues, il fallait respecter des règles et conventions de contribution. C'est-à-dire pour qu'une pull request ou merge request soit acceptée, il fallait par exemple dans le cas de ma contribution au projet **CoinTrend** :

Voici une capture d'écran du message du développeur avant la fusion finale, ci-dessous :


There are some changes that are still to be made. Check them out and submit the final pull request. Please, make sure to check the Android Studio options inside the commit dialog before committing: "Cleanup code", "Rearrange code", "Optimize imports", "Analyze code" so that the code follows the Kotlin formatting standards.

- Clean Code / Optimize imports / Rearrange code / Analyze code

Donc supprimer les importations de package ou de module inutiles, vérifier que toutes les indentations soient correctes que le code soit bien lisible, mais aussi vérifier le nom des variables et des fonctions. Dans ce cas précis, le langage de programmation était Kotlin, alors vérifier si cela correspond au formatage Kotlin, qui fait référence à la manière dont le code source Kotlin est présenté visuellement, c'est-à-dire l'indentation, l'espacement et l'agencement des éléments de code tels que les instructions, les fonctions, les classes, etc. Le formatage Kotlin est important, car il peut rendre le code plus lisible et plus facile à comprendre pour les développeurs qui travaillent sur le projet.

### Les Features Request

#### Projet Bike Computer

Bike Computer	Bike Computer
<a href="https://f-droid.org/fr/packages/de.nulide.bikecomputer/">https://f-droid.org/fr/packages/de.nulide.bikecomputer/</a>	Le Projet
<a href="https://gitlab.com/Nulide/BikeComputer">https://gitlab.com/Nulide/BikeComputer</a>	Le Code Source
<a href="https://gitlab.com/Nulide/BikeComputer/-/issues/9">https://gitlab.com/Nulide/BikeComputer/-/issues/9</a>	L'Issue
<a href="https://gitlab.com/Nulide/BikeComputer/-/merge_requests/2">https://gitlab.com/Nulide/BikeComputer/-/merge_requests/2</a>	La Merge Request Acceptée 

Le but était d'ajouter une fonctionnalité qui permettrait par la suite de transformer son smartphone en sonnerie pour vélo. L'application analyse la distance, le temps parcouru et mesure la vitesse du cycliste en temps réel pendant son trajet. L'idée était donc de rajouter un bouton avec une icône de cloche, tout en respectant les couleurs et le thème principale de l'application, qui lorsqu'il est pressé, lance un audio de son de cloche. (Cela peut être utile s'il on a pas de sonnerie, peut être utilisé avec le haut-parleur d'un téléphone ou connecté à une enceinte Bluetooth.)

#### En matière de code --> JAVA + XML:

- J'ai commencé par créer un bouton dans le fichier xml (qui sert pour le design).

```
<com.google.android.material.floatingactionbutton.FloatingActionButton
    android:id="@+id/btnBell"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="end|bottom"
    android:src="@drawable/ic_bell_bycicle"
    app:backgroundTint="@android:color/darker_gray"
    app:fabCustomSize="64dp"
    app:maxImageSize="32dp"
    android:layout_margin="20dp" />
```

- Puis importer les modules nécessaires, pour lire un fichier audio et pour le bouton qui est Floating

```
import android.media.MediaPlayer;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
```

- Déclarer 2 variables pour une pour un objet FloatingActionButton que je nome btnBell et une autre pour un objet MediaPlayer pour la lecture du fichier audio

```
private FloatingActionButton btnBell;
private MediaPlayer mediaPlayer;
```

- Ensuite récupérer avec la méthode findViewById, l'id du bouton que l'on a déclaré dans le fichier XML. Puis avec setOnClickListener, lorsque le bouton est presser on lance le fichier audio bicycle\_bell avec mediaPlayer.start(); qui se trouve dans un dossier raw

```
btnBell = findViewById(R.id.btnBell);
btnBell.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        mediaPlayer = MediaPlayer.create(getApplicationContext(), R.raw.bicycle_bell);
        mediaPlayer.start();
    }
});
```

<b>Bike Computer</b>	<b>Amélioration perso en</b> +
<a href="https://gitlab.com/Nulide/BikeComputer/-/merge_requests/3">https://gitlab.com/Nulide/BikeComputer/-/merge_requests/3</a>	La Merge Request Acceptée 

Added\_French\_Translation

!3 · created 1 month ago by Zine Eddine Hadj Rabah

MERGED 8 Approved 6

updated 2 minutes ago

Après avoir contribué une première fois au projet, j'ai tenté une deuxième merge request pour améliorer le projet. J'ai opté pour la prise en charge de plusieurs langues, car l'application est disponible seulement en anglais. Il ne s'agissait pas de traduire l'application sur un site web, mais de coder à l'intérieur de l'application, en créant des fichiers XML (qui vont nous servir de ressources pour le projet). J'y ai ajouté la prise en charge de la langue française et j'ai créé les fichiers nécessaires pour les différentes langues tels que l'espagnol, le portugais, l'ukrainien, le russe, le suédois, le norvégien, l'italien, le polonais, le chinois, et l'algérien. Cependant, la traduction ne peut s'arrêter à Google Traduction. Il faudrait donc créer une autre Feature Request "Need Translation"

#### En matière de code --> JAVA + XML:

- J'ai commencé par mettre toutes les chaînes de caractères de l'application dans un fichier string.xml qui se trouve dans un dossier /ressources, afin que les textes de l'application ne soient plus codés en dur.

Par exemple:

android:text="Distance: " est un littéral de chaîne qui spécifie directement le texte, tandis que android:text="@string/distance" est une ressource de chaîne qui fait référence à une chaîne stockée dans le strings.xml

```
android:text="Distance: "
android:text="@string/distance"
```

- Voici comment le mot "Distance :" est codé dans le fichier xml.

```
<string name="distance">Distance:</string>
```

- Puis j'ai créé plusieurs fichiers de ce type pour chaque langue (Italien, Portugais, Français, ...), ainsi lorsque l'utilisateur lance l'application son téléphone est en portugais alors l'application se traduira automatiquement en portugais. Il y a une consigne

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <!-- ***** Translate Into Italian ***** -->

    <!-- Setting page Text -->
    <string name="settings">Settings</string>
    <string name="metric">Metric:</string>
    <string name="minimal_view">Minimal View</string>
    <string name="dark_after">Dark after:</string>
    <string name="background">Background</string>
```

```

<string name="run_for">Run for:</string>
<string name="import_data">Import Data</string>
<string name="export_data">Export Data</string>

<!-- Overall Stat page Text -->
<string name="overall_stats">Overall Stats</string>
<string name="distance">Distance:</string>
<string name="max_speed">Max. Speed:</string>
<string name="avg_speed">Avg. Speed:</string>
<string name="travel_time">Travel Time:</string>

<!-- Trip Stat page Text just title -->
<string name="trip_stats">Trip Stats</string>
</resources>

```

- Pour finaliser, le développeur voulait me donner un accès au référentiel ou le droit de publier l'app, afin d'améliorer le projet encore plus. J'accepterais et mettrais le projet sur github pour donner accès aux ressources à d'autres développeurs et pour que des dev sur github qui n'ont pas accès a gitlab, puisse contribuer.

8 Nulide a approuvé cette demande de fusion il y a 6 minutes

Nulide @Nulide · il y a 4 minutes Propriétaire

Je dois donc mentionner que le projet est interrompu, j'accepte toujours les demandes de fusion mais je ne consacrerai malheureusement pas beaucoup de temps à ce projet.

Mais c'est agréable de voir vos efforts et que les gens se soucient toujours de l'application 😊

Nulide @Nulide · il y a 4 minutes Propriétaire

Je suis désolé d'avoir répondu si tard, je n'ai pas eu accès à mon ordinateur portable au cours des deux derniers mois.

Nulide @Nulide · il y a 2 minutes Propriétaire

Si vous souhaitez continuer le développement de l'application, je peux vous donner un accès en écriture au référentiel ou vous le bifurquez et le publiez par vous-même si vous le souhaitez.

faites le moi savoir :)

## Projet SimpleMobileTools - Simple Clock

Projet SimpleMobileTools	Simple Clock
<a href="https://f-droid.org/packages/com.simplemobiletools.clock/">https://f-droid.org/packages/com.simplemobiletools.clock/</a>	Le Projet
<a href="https://github.com/SimpleMobileTools/Simple-Clock">https://github.com/SimpleMobileTools/Simple-Clock</a>	Le Code Source
<a href="https://github.com/SimpleMobileTools/Simple-Clock/issues/414">https://github.com/SimpleMobileTools/Simple-Clock/issues/414</a>	L'Issue
En cours	La Merge Request 🔄


Le but était de créer un Widget, avec un Timer configurable par l'utilisateur. Le membre a validé ma feature, mais le développeur m'a dit de m'y prendre d'une autre manière. Après plusieurs échanges avec le développeur, il m'a conseillé de faire un widget configurable depuis une activité, dans laquelle je lirais la base de données

des timers et que l'utilisateur, choisira à ce moment le timer qui figurera dans le widget sur l'écran d'accueil.

## Projet SimpleMobileTools - Simple-Flashlight

--> En binôme avec Bruno DaSilva:

Projet SimpleMobileTools	Simple Flashlight
<a href="https://f-droid.org/fr/packages/com.simplemobiletools.flashlight/">https://f-droid.org/fr/packages/com.simplemobiletools.flashlight/</a>	Le Projet
<a href="https://github.com/SimpleMobileTools/Simple-Flashlight">https://github.com/SimpleMobileTools/Simple-Flashlight</a>	Le Code Source
<a href="https://github.com/bsae19/Simple-Flashlight">https://github.com/bsae19/Simple-Flashlight</a>	Fork du Projet
<a href="https://github.com/SimpleMobileTools/Simple-Flashlight/issues/23">https://github.com/SimpleMobileTools/Simple-Flashlight/issues/23</a>	L'Issue
<a href="https://github.com/SimpleMobileTools/Simple-Flashlight/pull/192">https://github.com/SimpleMobileTools/Simple-Flashlight/pull/192</a>	La Merge Request fini , en attente de validation ☺☺

 add of morse code entry in the flashlight  
#192 opened last month by bsae19

2

Le but était d'implémenter une fonction qui traduit un message texte en code morse avec le flash du smartphone, nous avons commencé par coder la fonction qui prend un texte (User Input) et le transforme en code morse par exemple le mot sos devient ... -- ... Puis après avoir su comment utiliser le flash, nous avons dans une boucle qui itère sur la phrase caractère par caractère, transformer le message texte en morse flash.

En matière de code --> KOTLIN:

- J'ai commencé par créer une image clickable, un bouton avec une icône sous licence CC0, qui représente le langage morse, pour que lorsqu'elle est cliquée, on ouvre une nouvelle page (MorseFlashActivity.kt)

```
<ImageView
android:id="@+id/morse_btn"
android:layout_width="@dimen/smaller_button_size"
android:layout_height="@dimen/smaller_button_size"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.25"
app:layout_constraintStart_toEndOf="@+id/sos_btn"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.628"
app:srcCompat="@drawable/ic_morse_vector" />
```

- Ensuite, on a déclaré mapOf est une fonction qui crée une carte en lecture seule à partir des paires clé-valeur spécifiée, pour la

traduction du message en morse , \_u = 200L va correspondre à la durée du flash, pause ou active.

```
var _u = 200L
val MSG = arrayListOf(_u)
val editText_UserInput = findViewById<EditText>(R.id.editText_UserInput)
val morseAlphabet = mapOf(
    'a' to ".-.",
    'b' to "-....",
    'c' to "-.-.",
    'd' to "-....",
    'e' to "..",
    'f' to "..-.",
    'g' to "--.",
    .... jusqu'à la fin
)
```

- L'algorithme, il s'agissait alors de récupérer le texte de l'utilisateur dans un editText, puis de le transformer en langage morse, par la suite avec le flash transformer le langage morse en flash

```
btn_morseFlash.setOnClickListener{
    val text_user = editText_UserInput.text

    System.out.println("Saisie Utilisateur ====> " + text_user)

    resultMorseCode = text_user.map { if (it == ' ') "/" else morseAlphabet[it]
}.joinToString(" ")
    textView_translation.setText(resultMorseCode)

    System.out.println("Traduction MORSE ====> " + resultMorseCode)

    MSG.clear()

    System.out.println("MSG vide ====> " + MSG)

    for (i in resultMorseCode.indices) {
        if (resultMorseCode[i] == '/'){
            cameraManager.setTorchMode(cameraId, false)
            System.out.println("flash OFF")
            MSG.add(_u*7)
            Thread.sleep(_u*7)
        }
        else if (resultMorseCode[i]!=' '){
            cameraManager.setTorchMode(cameraId, false)
            System.out.println("flash OFF")
            if(resultMorseCode[i+1]!='/' && resultMorseCode[i-1]!='/'){
                Thread.sleep(_u*3)
                MSG.add(_u*3)
            }
        }
    }
}
```

```

        else if (resultMorseCode[i]=='-'){
            cameraManager.setTorchMode(cameraId, true)
            System.out.println("flash ON")
            MSG.add(_u*3)
            Thread.sleep(_u*3)
            cameraManager.setTorchMode(cameraId, false)
            System.out.println("flash Off")
            if(i+1<resultMorseCode.length){
                if(resultMorseCode[i+1]!=' '){
                    Thread.sleep(_u)
                    MSG.add(_u)
                }
            }

        }

        else if (resultMorseCode[i]=='.'){
            cameraManager.setTorchMode(cameraId, true)
            System.out.println("flash ON")
            MSG.add(_u)
            Thread.sleep(_u)
            cameraManager.setTorchMode(cameraId, false)
            System.out.println("flash OFF")
            if(i+1<resultMorseCode.length){
                if(resultMorseCode[i+1]!=' '){
                    Thread.sleep(_u)
                    MSG.add(_u)
                }
            }

        }
    }
    MSG.removeLast()
    MSG.add(_u*7)
    System.out.println("MESSAGE FLASH =====> " + MSG)
}

```

- Pour l'utilisation du flash, il a fallu utiliser la méthode `getSystemService()` pour obtenir une instance du `CameraManagerservice` système. `CAMERA_SERVICE` est argument passé à `getSystemService()` qui indique que l'on veut récupérer le `CameraManagerservice`, pour l'obtention du flash.

```

val cameraManager = getSystemService(CAMERA_SERVICE) as CameraManager
val cameraId = cameraManager.cameraIdList[0]


```

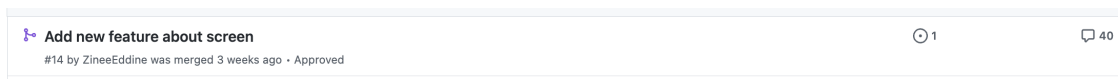
## Projet CoinTrend

CoinTrend	CoinTrend
<a href="https://f-droid.org/fr/packages/com.cointrend/">https://f-droid.org/fr/packages/com.cointrend/</a>	Le Projet
<a href="https://github.com/CoinTrend/CoinTrend">https://github.com/CoinTrend/CoinTrend</a>	Le Code Source
<a href="https://github.com/CoinTrend/CoinTrend/issues/9">https://github.com/CoinTrend/CoinTrend/issues/9</a>	L'issue
<a href="https://github.com/CoinTrend/CoinTrend/pull/10">https://github.com/CoinTrend/CoinTrend/pull/10</a>	La Merge Request, rediriger par le développeur 

La feature a été validé par l'auteur mais pas le développeur, du coups le developpeur à crée une nouvelle issue, et m'a orienté vers une nouvelle méthode

---> En bînome avec Daouda Kanouté:

Redirection du développeur (CoinTrend)	issue/pull
<a href="https://github.com/CoinTrend/CoinTrend/issues/12">https://github.com/CoinTrend/CoinTrend/issues/12</a>	L'issue ouvert par le développeur
<a href="https://github.com/CoinTrend/CoinTrend/pull/14">https://github.com/CoinTrend/CoinTrend/pull/14</a>	La Merge Request Acceptée 



Le But était de créer un bouton "donate" qui ouvre un lien internet, pour faire des dons depuis l'application au développeur. J'ai expliqué au membre que ce genre de bouton se trouve depuis une section ou rubrique "About / À propos" plus habituellement. Il m'a conseillé de le mettre en haut à droite de l'application, ce que j'ai fait avant de proposer une Pull Request. Le développeur a répondu la même chose et a dit qu'il fera dans une prochaine mise à jour, une rubrique "Settings" dans laquelle il codera le bouton "donate", car il n'a pas lieu d'être en haut à droite. Du coups l'auteur de la feature à aimer le résultat, mais le développeur n'a pas accepter. Il a ensuite ouvert une autre issue pour que l'on crée la page "About" , About Screen, puis il nous a guidé ( moi et Daouda) jusqu'à la fin de l'issue et a accepté notre pull request.

En matière de code --> KOTLIN + Jetpack Compose:

- Le langage de programmation est Kotlin et nous avons découvert Jetpack Compose qui est un kit de développement pour android, pour la conception d'User Interface de type native, 0 fichier XML. Toute l'application en un code le design et la logique.
- Importations nécessaire au développement.

```
package com.cointrend.presentation.ui.about

import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.alpha
import androidx.compose.ui.draw.clip
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
```



```
import androidx.compose.ui.text.style.TextOverflow
import androidx.compose.ui.unit.dp
import com.cointrend.presentation.R
import com.cointrend.presentation.commoncomposables.*
import com.cointrend.presentation.models.Screen
import com.cointrend.presentation.theme.*
import dev.olshevski.navigation.reimagined.NavController
import dev.olshevski.navigation.reimagined.pop
```

- Ensuite nous avons étudié les différentes pages de l'application pour, adapter notre code: qui prend une fonction fun AboutScreen, qui dans le Scaffold() va prendre tout les boutons, textes, et images. Ce type de code ressemble énormément à FLUTTER. Dans le Scaffold, il y a la TopAppBar() dans laquelle on y place les textes ou boutons que l'on veut qui figurera sur la barre tout en haut de l'application.

```
private val defaultHorizontalPadding = 16.dp

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun AboutScreen(
    navController: NavController<Screen>,
) {

    Scaffold(
        modifier = Modifier,
        topBar = {
            TopAppBar(
                title = {},
                navigationIcon = {
                    IconButton(onClick = {
                        navController.pop()
                    }) {
                        Icon(
                            painter = painterResource(id =
R.drawable.ic_arrow_back_ios),
                            contentDescription = "Return to previous screen",
                            modifier = Modifier.padding(start = 12.dp)
                        )
                    }
                },
            )
        }
    )
}
```

- Puis nous déclarons une LazyColumn qui prendra tous les éléments essentiels à la page "About", comme nous l'a demandé le développeur dans un design bien précis, une colonne contenant, le logo, qui est placé dans un item de la LazyColumn.

```
) { padding ->
```

```

LazyColumn(
    Modifier
        .fillMaxSize()
        .padding(padding)
) {

```

davidepanidev marked **this** conversation **as** resolved.

```

    item {
        CoinTrend(modifier = Modifier.fillMaxWidth())
    }

    item {
        Spacer(modifier = Modifier.size(16.dp))
    }

    item {
        SectionTitle(
            title = "Source Code", modifier = Modifier.padding(
                defaultHorizontalPadding
            )
        )
    }

    item {
        Column(
            modifier = Modifier
                .padding(horizontal = 8.dp)
                .background(
                    color = StocksDarkSelectedChip,
                    shape = MaterialTheme.shapes.large
                ),
        ) {
            SectionInfoItemAbout(
                name = "GitHub",
                info = "https://github.com/CoinTrend/CoinTrend",
                image = R.drawable.ic_github,
                showDivider = true
            )
            SectionInfoItemAbout(
                name = "Changelog",
                info = "https://github.com/CoinTrend/CoinTrend/releases",
                image = R.drawable.ic_github,
                showDivider = false
            )
        }
    }

    item {
        Spacer(modifier = Modifier.size(16.dp))
    }

    item {

```

```

        SectionTitle(
            title = "Contact", modifier = Modifier.padding(
                defaultHorizontalPadding
            )
        )
    }

    item {
        Column(
            modifier = Modifier
                .padding(horizontal = 8.dp)
                .background(
                    color = StocksDarkSelectedChip,
                    shape = MaterialTheme.shapes.large
                ),
        ) {
            SectionInfoItemAbout(
                name = "Email",
                info = "cointrend.info@gmail.com",
                image = R.drawable.ic_mail,
                showDivider = true
            )
            SectionInfoItemAbout(
                name = "GitHub Issues",
                info = "https://github.com/CoinTrend/CoinTrend/issues",
                image = R.drawable.ic_github,
                showDivider = false
            )
        }
    }

    item {
        Spacer(modifier = Modifier.size(16.dp))
    }

    item {
        SectionTitle(
            title = "Support", modifier = Modifier.padding(
                defaultHorizontalPadding
            )
        )
    }

    item {
        Column(
            modifier = Modifier
                .padding(horizontal = 8.dp)
                .background(
                    color = StocksDarkSelectedChip,
                    shape = MaterialTheme.shapes.large
                ),
        ) {

```

```

        SectionInfoItemAbout(
            name = "Donate",
            info = "https://github.com/CoinTrend#support",
            image = R.drawable.ic_donate,
            showDivider = true
        )
        SectionInfoItemAbout(
            name = "Rate on Google Play",
            info = "",
            image = R.drawable.ic_google_play,
            showDivider = false
        )
    }
}

item {
    Spacer(modifier = Modifier.size(32.dp))
}

}
}
}

```

- Il a fallu adapter notre code à celui de développeur comme il a fait pour des question de lisibilité et d'optimisation. On a codé des @Composable, qui est un composant qui une fois coder peut être appelé plusieurs fois partout dans l'application. private fun CoinTrend, prend une Column à l'intérieur une Image et un Texte, que l'on rappelle dans l'Item de la LazyColumn.

```

@Composable
private fun CoinTrend(
    modifier: Modifier = Modifier
) {
    Column(
        modifier = modifier,
        verticalArrangement = Arrangement.Center,
        horizontalAlignment = Alignment.CenterHorizontally,
    ) {
        Image(
            modifier = Modifier
                .requiredHeight(120.dp)
                .padding(top = 2.dp),
            contentDescription = "CoinTrend logo icon",
            contentScale = ContentScale.Fit,
            painter = painterResource(id = R.drawable.ic_launcher),
        )
        Text(
            modifier = Modifier.padding(top = 2.dp),
            text = "CoinTrend",
            fontSize = MaterialTheme.typography.bodyLarge.lineHeight,
            color = MaterialTheme.colorScheme.onBackground,
            style = MaterialTheme.typography.titleLarge,
        )
    }
}

```

```

        textAlign = TextAlign.Center,
        fontWeight = FontWeight.Bold,
    )
}

@Composable
fun SectionInfoItemAbout(
    name: String,
    info: String,
    image: Int,
    showDivider: Boolean,
) {
    Row(
        modifier = Modifier
            .padding(horizontal = 16.dp, vertical = 8.dp)
            .fillMaxWidth(),
        horizontalArrangement = Arrangement.spacedBy(16.dp),
        verticalAlignment = Alignment.CenterVertically,
    ) {
        Image(
            modifier = Modifier
                .size(size = 30.dp)
                .clip(shape = MaterialTheme.shapes.medium),
            painter = painterResource(id = image),
            contentDescription = null,
            contentScale = ContentScale.Crop
        )
        Column {
            Text(
                text = name,
                fontWeight = FontWeight.Bold,
                overflow = TextOverflow.Ellipsis,
                color = MaterialTheme.colorScheme.onPrimaryContainer,
                maxLines = 1
            )
            Spacer(modifier = Modifier.size(2.dp))
            Text(
                text = info,
                style = MaterialTheme.typography.bodyMedium,
                color = MaterialTheme.colorScheme.onSecondaryContainer,
                fontWeight = FontWeight.Medium,
                maxLines = 1,
                overflow = TextOverflow.Ellipsis
            )
        }
    }
}

if (showDivider) {
    Divider(
        modifier = Modifier
            .padding(horizontal = 8.dp)
            .alpha(.2f),
    )
}

```

```

        color = Color.White
    )
}
}

```


## Crédits

### Bike Computer

BikeComputer

[DISCONTINUED]

A FOSS BikeComputer



Features

- Measures your speed.
- Keep track of your stats. (Distance, Maximum Speed, TravelTime, ...)


Contributors

Zine Eddine Hadj Rabah [@zhrdev354](#)

Open Source Library/Assets

[See Licenses](#)

Donate



### CoinTrend

Credits


Contributors


- [ZineEddine](#) & [beastboym](#): thanks for working on the AboutScreen.


Libraries

- [Philipp Lackner](#): he inspired the LineChart with his [StockChart](#). Check out his YouTube channel for great videos about Android Development!
- [Zach Klippenstein](#): he inspired the SegmentedControl with his [Composable](#).
- [nicolashaan](#): for the [Resultat](#) library.
- [olshevski](#): for the [Compose Navigation Reimagined](#) library.
- [mxalbert1996](#): for the [Compose Shared Element](#) library.
- [aclassen](#): for the [ComposeReorderable](#) library.

Contributors 3

 [davidpanidev](#) Davide

 [beastboym](#) Dsouda

 [ZineEddine](#) Zine Dev