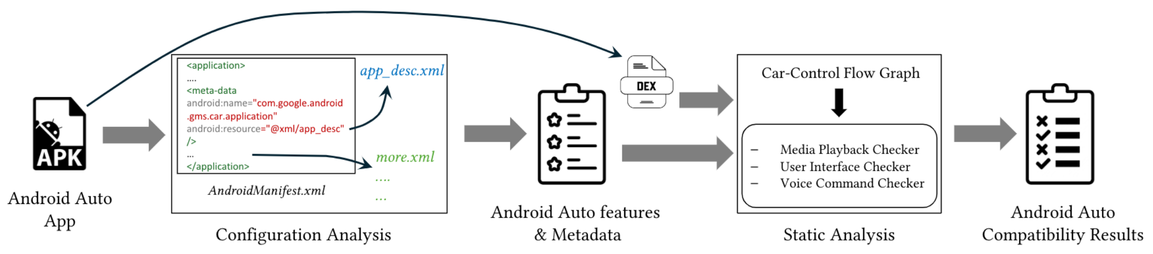
# Table of Contents

---

# CarCompat Introduction

> CarCompat is the static analysis tool we introduce in our paper "Analyzing and Detecting Compatibility Issues in Android Auto Apps".

We introduce `CarCompat`, a static analysis framework that detects compatibility problems in Android Auto apps. `CarCompat` constructs Car-Control Flow Graph (CCFG) to capture interactions between app components, lifecycle methods, and platform-specific callbacks. It applies specialized checkers to detect UI violations, media playback errors, and issues with voice command handling. We evaluated `CarCompat` on a dataset of 44 Android Auto apps and detected 27 new issues, 9 of which have been confirmed by developers, and 3 of the confirmed issues have already been fixed. The results show that `CarCompat` helps developers identify and fix compatibility issues, improving the in-vehicle experience.



This artifact contains the apks, python scripts we used to retrieve apks and repos with android auto support from F-Droid and GitHub for our study and evaluation, the dataset of apps, and also a binary of CarCompat. It also contains reports generated by CarCompat for each of these apks in Corpus-G.

## Artifacts

### Python Scripts

- fdroid_crawler.py - Python script we used for collecting Free and Open Source Software android APKs from f-droid. Running this script required downloading the Selenium ChromeDriver.
- helpers.py - Utility script for functions we used in the `fdroid_crawler.py` script.
- github_code_search.py - Python script we used for searching GitHub repositories for android auto patterns.
- github_repo_details.py - Python script we used for extracting detailed metadata from the GitHub repositories we retrieved.
- car_feat.py - Python binary analysis tool used to perform Android Auto specific feature checks.

### Data Files

- Corpus-L - dataset of 4,387 apps used in our formative study.
- Corpus-R - dataset of 44 apps used in our formative study.
- Corpus-G - dataset of 44 apps used in our evaluation of CarCompat.

### Application Resources

- apks - directory containing the apks in *Corpus-G* used for compatibility testing and analysis.
- carcompat-0.3.0.jar - our tool for android auto compatibility testing.

### Research Output

- RQs - directory containing evaluation results organized by research questions. Includes statistical analysis, test outputs, and performance metrics used to validate project hypotheses.

## Artifact Directory Tree

```
/
├── RQs/
│   ├── RQ1/
│   │   ├── Corpus-L.csv
│   │   └── README.md
│   ├── RQ2/
│   │   ├── Corpus-R.csv
│   │   ├── Issues.csv
│   │   └── README.md
│   ├── RQ3/
│   │   ├── CorpusG - Detailed Analysis Report/
│   │   ├── Corpus-G.csv
│   │   ├── README.md
│   │   └── RQ3.csv
│   └── RQ4/
│       ├── RQ4 - Detailed Issues Report/
│       ├── README.md
│       └── RQ4.csv
├── RQ5/README.md
├── RQ6/README.md
├── RQ7/README.md
├── apks/
├── LICENSE
├── README.md
├── car_feat.py
├── carcompat-0.3.0.jar
├── fdroid_app_categories.csv
├── fdroid_crawler.py
├── github_code_search.py
├── github_repo_details.py
├── helpers.py
└── setup.sh
```

## Requirements

- Java 17
- Python 3

## Download

To run the analysis using the binary, you need to have Java 17 and Python3 installed. Then follow the instructions

1. Git clone the GitHub repo or Download the repository from here.

2. Open your terminal and navigate to the root of the project.

3. Ensure you follow the setup instructions provided below before running the analysis. You must: - clone the `android-platforms` in the root of the project - run the `setup.sh` script

4. Download the archived apks from here, unzip and place them in the `/apks` folder.

## Setup Instructions

```
$ cd carcompat-0503
$ git clone https://github.com/Sable/android-platforms
$ chmod +x setup.sh
$ ./setup.sh # Install python dependencies
```

## Run

All apks used for the evaluation section of our paper should be downloaded from the provided link and placed here.

```
# example usage
$ java -jar carcompat-0.3.0.jar -a apks/retromusicplayer.ap
```

## Evaluation Results

The reports for our research are available in the RQs folder. Detailed reports for each apk analysis has been saved and made available here and here.

## License

---

# RQ 1: Landscape

> Which apps are attempting to incorporate Android Auto features?
> What are the common categories?

We collected all available apps from F-Droid, resulting in a dataset of 4,387 Android apps, which we refer to as Corpus-L.

**Table: Table: Category distribution of Android Auto Apps discovered in *Corpus-L***

| Categories | Description | # of apps |
|---|---|---|
| Media | Allows users to access audio content like music, radio, and audiobooks. | 23 (52.3%) |
| Messaging | Enables message notifications, text-to-speech, and voice-activated replies. | 18 (40.9%) |
| Navigation | Offer turn-by-turn navigation to guide drivers to their destinations. | 2 (4.5%) |
| POI | Apps for discovering and navigating to points of interest, including parking. | 1 (2.3%) |

### Answer

Out of 4,387 apps in Corpus-L, only 44 apps were found that support Android Auto. As expected, the majority of these apps belong to the media category, reflecting the dominant role that car infotainment systems play in media consumption. The messaging apps account for 40.9% of the total, which are useful for the communication needs during driving. In contrast, navigation and POI apps are the least represented categories, indicating a limited development focus in these areas.

---

## RQ 2: Common Issues and Causes

> What are the basic types of Android Auto compatibility issues? Are there any common causes?

To investigate the issues in more detail, we focus on the subset of apps that have explicitly added support for Android Auto and have reported issues in their repositories. This refined dataset is referred to as Corpus-R.

**Table: Breakdown of Android Auto App Issues in Different Categories for *Corpus-R*.**

| Issue Category | Number of Issues |
|---|---|
| Media Playback functionality | 104 (70.7%) |
| User Interface problems | 36 (24.4%) |
| Voice Command problems | 7 (4.8%) |
| Total | 147 |

### Answer

We manually examine each Android Auto issue and classify them based on their manifestation. Following analysis and discussion, all authors reached a consensus on three primary categories: `T1 - Media Playback`, `T2 - User Interface`, and `T3 - Voice Commands`.

A common theme among these three issues is an oversight by developers when implement- ing all the necessary requirements defined by the car app quality guidelines, depending on the app category, when adding support for Android Auto. The absence of automated testing tools further exacerbates the prevalence of these issues, compounding the chal- lenges in both the development and deployment phases of Android Auto-compatible apps

---

## RQ 3: Applicability

> Can CarCompat be applied to real-world apps and find issues?

To evaluate the applicability of CarCompat, we applied it to Corpus-G, a dataset of 44 open-source Android Auto apps spanning diverse categories and development practices. The objective was to assess whether CarCompat could effectively detect compatibility issues across various real-world apps and remain computationally efficient for large-scale analysis.

This directory contains the complete results of our evaluation of `CarCompat` on all apps in **Corpus-G**.

Apps in Corpus-G are made available in the `/apks` folder here.

**Table: CarCompat's performance on open-source Android Auto applications in *Corpus-G***

| CarCompat Evaluation Metrics | Corpus-G |
|---|---|
| # of Apks | 44 |
| # of Apps w Issues | 15 |
| # of Issues Detected | 27 |
| GeoMean Time(s) | 5.6 |
| LoC | 6.5m |

### Answer

CarCompat is applicable to real-world Android Auto apps and achieves low runtime, enabling efficient detection of compatibility issues in real-world Android Auto apps. Its scalability makes it a practical tool for identifying and addressing compatibility challenges across diverse codebases.

---

## RQ 4: Effectiveness of CarCompat

> Can CarCompat effectively and precisely detect Android Auto compatibility issues?

To evaluate the effectiveness of CarCompat, we assessed its accuracy on 15 Android Auto apps where either Android Lint or CarCompat detected at least one compatibility issue in RQ3.

Details of the detected issues are available here.

RQ4.csv contains the table summarizing these issues.

### Answer

CarCompat achieves high accuracy in detecting Android Auto compatibility issues, correctly identifying 27 issues with no false positives or false negatives. These results demonstrate that CarCompat provides reliable checks on real-world Android Auto apps.

---

## RQ 5: Comparison with Existing Tools

> How does CarCompat compare with existing Android Auto analysis tools in terms of issue coverage and accuracy?

To evaluate CarCompat against existing Android Auto analysis tools, we compare it with Android Lint, which, to the best of our knowledge, is the only tool that includes built-in checks for Android Auto compatibility violations.

**Table: Performance comparison between CarCompat and Android Lint on the 15 apps in Corpus-G with detected issues**

| Evaluation Metrics | CarCompat | Android Lint |
|---|---|---|
| # of Issues Detected | 27 | 2 |
| True Positives | 27 | 2 |
| False Positives | 0 | 0 |
| GeoMean Time(s) | 4.2 | 11.1 |

Details of the detected issues are available here.

RQ4.csv contains the table summarizing these issues.

### Answer

CarCompat significantly outperforms Android Lint by detecting 27 issues compared to only 2, while both tools report zero false positives. Additionally, CarCompat runs more than 2X faster, making it a more comprehensive and efficient solution for Android Auto compatibility analysis.

---

## RQ6: Distribution

> What is the distribution of compatibility issues across these categories in real-world Android Auto apps?

**Table: Distribution of Android Auto Compatibility Issues by Category**

| Issue category | # of Issues | # of Apps |
|---|---|---|
| Media Playback | 7 | 5 |
| User Interface | 10 | 5 |
| Voice Command | 10 | 10 |

### Answer

The most common Android Auto compatibility issues are voice command integration (37%), UI issues (37%), and media playback issues (26%). These findings confirm that the categories identified in our study (Sec. 3) apply to real-world apps.

---

## RQ7 (Usefulness)

> Do developers agree with the compatibility issues detected by CarCompat, and how often are these issues acknowledged or addressed?

**Table: Summary of Android Auto compatibility issues detected by CarCompat and developer response**

| | |
|---|---|
| # of Issues Detected | 27 |
| # of Issues Confirmed | 9 |
| # of Issues Fixed | 3 |

**Table: Detailed breakdown of reported issues with developer feedback (gotten so far)**

| App | Issue Count | Issue # | Issue Type | Status |
|---|---|---|---|---|
| App 1 | 2 | 503 | UI | Fixed |
| App 2 | 1 | 72 | Voice | Fixed |
| App 3 | 1 | 971 | Voice | Confirmed |
| App 4 | 2 | 1127 | Voice | Confirmed |
| App 5 | 3 | 253 | Media | Confirmed |

Details of the detected issues are available here.

RQ4.csv contains the table summarizing these issues.

## Conclusion

The issues reported by CarCompat were valuable to developers, with nine acknowledged and three already fixed. Additionally, insights gained from our empirical study can be generalized to help improve the software quality of other Android Auto apps.