Poster: xg.glass: Develop AI Glasses Applications in 10 Lines of Code and Easily Deploy onto Different Glasses

Jiayang Xu*, Yufan Zhuang*, Zijie Li, Jun Zhang, and Zili Meng
Hong Kong University of Science and Technology

ACM Reference Format:
Jiayang Xu*, Yufan Zhuang*, Zijie Li, Jun Zhang, and Zili Meng. 2026. Poster: xg.glass: Develop AI Glasses Applications in 10 Lines of Code and Easily Deploy onto Different Glasses. In *The 27th International Workshop on Mobile Computing Systems and Applications (HotMobile '26), February 25–26, 2026, Atlanta, GA, USA.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3789514.3796251

1 INTRODUCTION

Since the release of Rokid Glasses and Meta Rayban Display, smart AI glasses are becoming increasingly popular in the recent days. With the capability of camera and display, people use smart glasses for real-time translation, teleprompting, navigation, and personal assistant [1]. It is estimated that smart glasses market is already valued at USD 1 billion and is increasing by 10% every year [4].

However, the growth of smart glasses heavily relies on the potential killer and groundbreaking applications, which won't work without the efforts from the community. Yet despite the rapid emergence of new devices, developers of the smart glasses applications face two key challenges:

- Scalability. The glasses models nowadays are extremely diverse (e.g., 50+ at CES 2026). Among them, the SDKs which define the APIs are vendor-specific, making it hard to migrate existing applications to other glasses. Moreover, the differences between SDKs are not simply different function names but also runtime models – whether the glasses run as a standalone Android device. The connectivity also varies across different glasses (e.g., WiFi vs. Bluetooth).
- Friendliness. To figure out the new scenarios of smart glasses, quickly testing the preliminary ideas with minimal development overhead is necessary. Yet, the provided glasses usually ask the developers to start from low-level SDK interfaces (e.g., Bluetooth and WiFi connectivity). It is often challenging for solo developers to master both low-level programming and high-level algorithm.

Despite community efforts to standardize development environments, current options are still not ideal for AI glasses applications. For example, MentraOS [3] provides OS-level support tightly coupled with specific hardware and drivers, which is not scalable. Meanwhile, OpenXR [2] is highly customized for AR/XR that makes the application overcomplicated. To address these issues, this paper introduces a universal glasses SDK, `xg.glass`, that provides a unified development interface across multiple AI glasses. We find that the core functionality of the SDK can be reduced to only four features:

- `capturePhoto()`: the input from the glasses camera.
- `audioRecord()`: the input from the glasses microphone.
- `displayText()`: the output to the glasses display.
- `audioPlay()`: the output to the glasses speaker.

By analyzing the proposed applications from different developer forums of Rokid, Frame, Halliday and Omi, we find that all applications can be supported using the only four APIs above. We expose only these four interfaces to developers; everything else is handled internally, enabling developers to prototype cross-glasses applications with only a few lines of code.

In the meantime, `xg.glass` also provides a command-line tool (CLI) that lets developers generate a runnable mobile app by writing a single file containing the application logic and then invoking the CLI to build and package the project, which later we will demonstrate with an example. `xg.glass` further provides a simulator for the glasses so that developers can easily test with pre-recorded local video or webcam and seamlessly deploy on real glasses with the same codebase. The SDK is available at https://github.com/hkust-spark/xg-glass-sdk and the documentation is at https://xg.glass.

2 EXAMPLE

Using `xg.glass`, developers can implement a simple app that captures a photo with the glasses, translates the text, and displays the result – in about 10 lines of code:

```
override suspend fun run(ctx: UniversalAppContext): Result<Unit> {
    val img = ctx.client.capturePhoto().getOrThrow()
    val b64 = Base64.getEncoder().encodeToString(img.jpegBytes)
    val req = chatCompletionRequest {
        model = ModelId("gpt-4o-mini")
        messages { user
            { content { text("Translate the text in this image to Chinese. Output
            only the translation."); image("$b64") } } }
    }
    val text = openAI.chatCompletion(req
        ).choices.firstOrNull()?.message?.content.orEmpty().ifBlank { "No text" }
    return ctx.client.display(text, DisplayOptions())
}
```

Developers then simply run the CLI to build the app:

```
xg-glass run SingleFile.kt
```

Finally, users only need to install the generated app on the phone and launch it to run on any glasses supported by `xg.glass`. We currently support three types of glasses: Rokid, Frame, and RayNeo, and are expanding support to more glasses.

REFERENCES

[1] Kai Essig, Benjamin Strenge, and Thomas Schack. 2016. ADAMAAS: towards smart glasses for mobile and personalized action assistance. In *Proceedings of the 9th ACM International Conference on PErvasive Technologies Related to Assistive Environments*. 1–4.
[2] The Khronos Group. 2025. OpenXR - High-performance access to AR and VR platforms and devices. (2025). https://www.khronos.org/OpenXR/
[3] Mentra Labs, Inc. 2026. MentraOS. (2026). https://mentraglass.com/os
[4] Rohan Jadhav. 2025. AI Smart Glasses Market Size, Share & Growth Report 2035. (2025). https://www.snsinsider.com/reports/ai-smart-glasses-market-7330

*Equal contribution. Zili Meng is the corresponding author. This work has been supported by Guangdong Basic and Applied Basic Research Fund (No. 2025A1515010460).

HotMobile '26, February 25–26, 2026, Atlanta, GA, USA

ACM ISBN 979-8-4007-2471-8/26/02.
https://doi.org/10.1145/3789514.3796251