SmartBioStream

Installation and deployment manual

Contents

1. Introduction

SmartBioStream is a Wear OS application that enables the collection of biometric data through affordable, user-friendly smartwatches. In addition, we provide a web platform, ServerBioStream, to receive, store, and download the data sent by the smartwatches. The combination of both applications supports conducting biometric studies without needing expensive and sophisticated research-oriented devices. SmartBioStream can also transmit biometrics to other systems for their analysis, such as educational and workplace platforms.

SmartBioStream' s user interface was designed to be simple and intuitive, especially the views of the end-users. While administrators and researchers may have more experience with the application, end-users, such as case study participants, probably do not. Consequently, we focused on handling complexities within the application code to provide a better user experience. This application integrates three main functionalities:

- **Options**. This functionality enables users to adjust settings related to communication with the server. These settings should be set by researchers or administrators, not by the end-users. This configuration is stored on the smartwatch and remains saved even if the application is closed. The views of this functionality enable the selection of the server's IP address, the port, the protocol (HTTP or HTTPS), whether the application should verify the HTTPS certificate, and the authentication method (username and password or identifier).

- **Connection test**. This view checks the connectivity between the application and the server. Before starting an experiment, a researcher should verify the server's availability.

- **Data collection**. These views collect and send biometrics from the end-users. Depending on the application settings, this functionality may require an experiment identifier or a username and password. After registration, the users select from the available sensors, including heart rate, accelerometer, gyroscope, and temperature. Then, the users access the recording view, where they can temporarily pause the data collection, return to the sensor menu, or navigate back to the main menu.

2. Android Studio

The first step to deploy SmartBioStream is installing Android Studio. Android Studio is an integrated development environment (IDE) designed for creating Android applications, including those for Wear OS smartwatches. Using this IDE, we can install SmartBioStream on a smartwatch or run it in an emulator for testing.

To download Android Studio, we visit the official website [1]. Android Studio is available for Linux, MacOS, and Windows operating systems. We recommend choosing the standard installation, during the installation we must accept the license agreements and terms of use.

Once the IDE is installed, we launch Android Studio for the first time. In the welcome menu, we select "Open," navigate to the decompressed SmartBioStream folder, and open the application file (the SmartBioStream file with the green Android logo), as illustrated in Figure 1. After importing the project, we would need to wait until all the configuration files are loaded (Figure 2). Figure 3 displays the project view, it is important to verify the "play" button is in green.

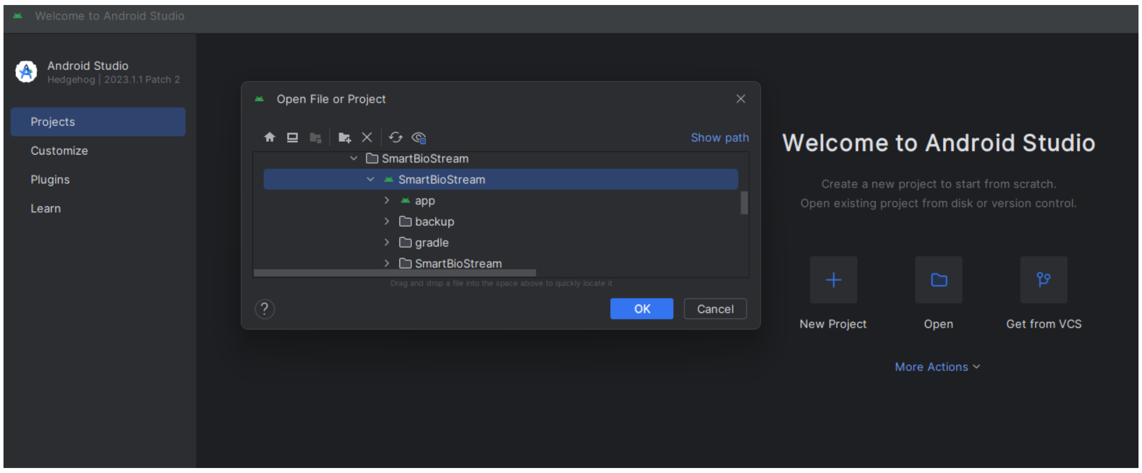

Welcome to Android Studio
Android Studio
Hedgehog | 2023.1.1 Patch 2
Projects
Customize
Plugins
Learn
Open File or Project
Show path
SmartBioStream
SmartBioStream
app
backup
gradle
SmartBioStream
Drag and drop a file into the space above to quickly locate it
OK
Cancel
Welcome to Android Studio
Create a new project to start from scratch.
Open existing project from disk or version control.
New Project
Open
Get from VCS
More Actions

Figure 1. Import the SmartBioStream project in Android Studio

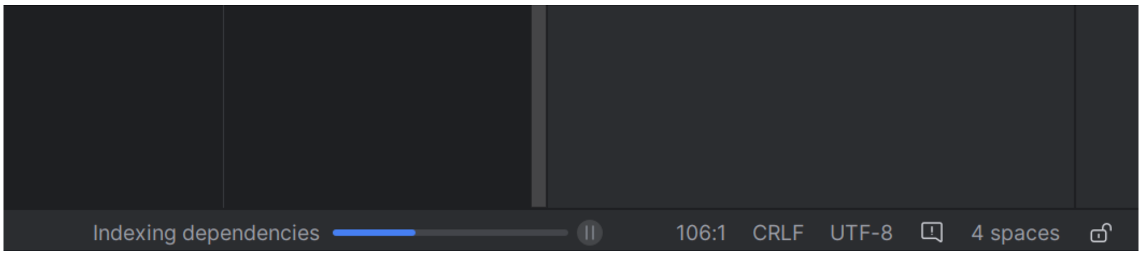


Figure 2. Progress bar indicating the import of a project in Android Studio

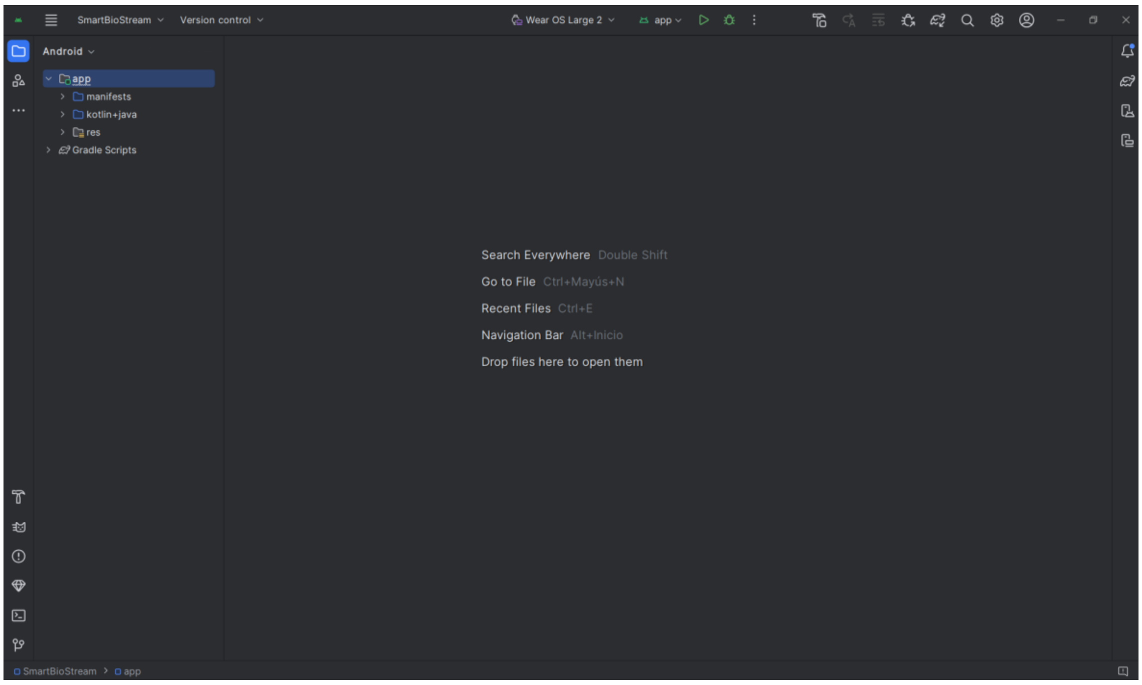


Figure 3. Project menu in Android Studio

3. Test SmartBioStream in an emulator

In this section, we explain how to utilize SmartBioStream with the emulators available in Android Studio. Additionally, there are official tutorials available in [2] and [3]. Starting from the project view, we open the Device Manager menu by clicking the third button in the right panel (Figure 4). Then, we select the "+" button to add a new emulator, and we click on "Create Virtual Device."

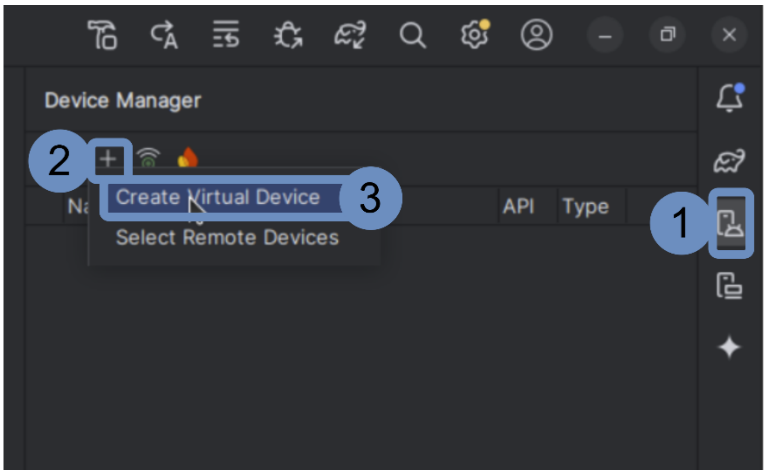
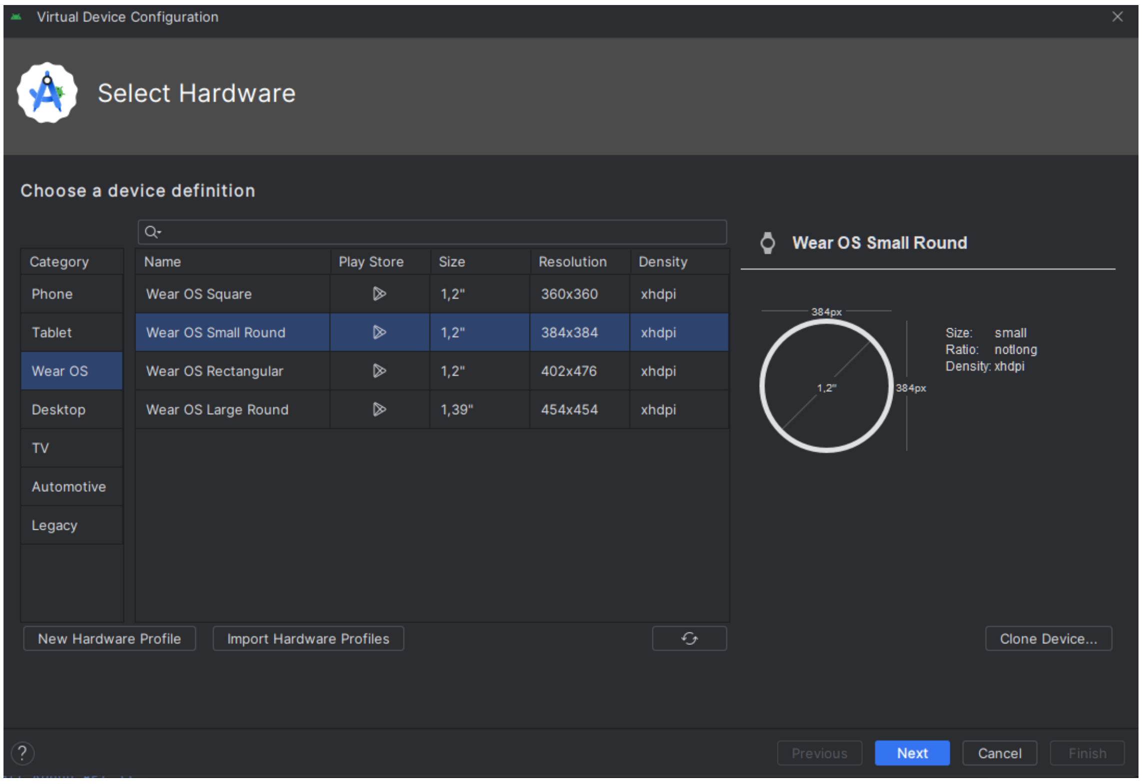


Figure 4. Accessing the creation of a smartwatch emulator in Android Studio

Next, we will create a smartwatch emulator. To do this, in the menu shown in Figure 5, we select "Wear OS." Within this category, we choose "Wear OS Small Round." After clicking the "Next" button, we need to select an operating system image. We recommend using the "Android 13 Tiramisu" version. Before selecting it, we must first download the image by clicking the download icon next to the desired version. Once the download is complete, we can proceed with the selection operating system image. This process is depicted in Figure 6. In the next window, no additional configuration changes are needed; we simply click the "Finish" button.


Virtual Device Configuration

Select Hardware

Choose a device definition

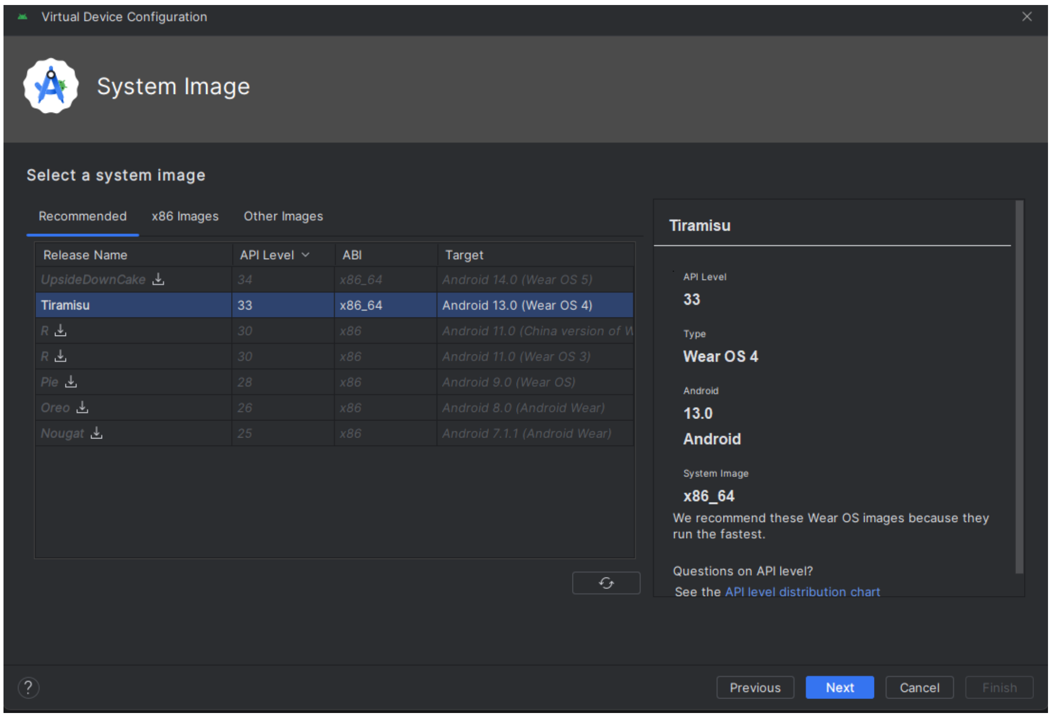
Category
Phone
Tablet
Wear OS
Desktop
TV
Automotive
Legacy

Name	Play Store	Size	Resolution	Density
Wear OS Square	▷	1,2"	360x360	xhdpi
Wear OS Small Round	▷	1,2"	384x384	xhdpi
Wear OS Rectangular	▷	1,2"	402x476	xhdpi
Wear OS Large Round	▷	1,39"	454x454	xhdpi

Wear OS Small Round

Size: small
Ratio: notlong
Density: xhdpi

Figure 5. Device configuration for the smartwatch emulator


Virtual Device Configuration

System Image

Select a system image

Release Name	API Level	ABI	Target
UpsideDownCake	34	x86_64	Android 14.0 (Wear OS 5)
Tiramisu	33	x86_64	Android 13.0 (Wear OS 4)
R	30	x86	Android 11.0 (China version of W
R	30	x86	Android 11.0 (Wear OS 3)
Pie	28	x86	Android 9.0 (Wear OS)
Oreo	26	x86	Android 8.0 (Android Wear)
Nougat	25	x86	Android 7.1.1 (Android Wear)

Tiramisu

API Level
33

Type
Wear OS 4

Android
13.0
Android

System Image
x86_64

We recommend these Wear OS images because they run the fastest.

Questions on API level?
See the API level distribution chart

Figure 6. System image configuration for the smartwatch emulator

Figure 7 shows the newly added device in the Device Manager. Finally, we click the “Start” button, which automatically opens the Running Devices window and displays the smartwatch, as shown in Figure 8.

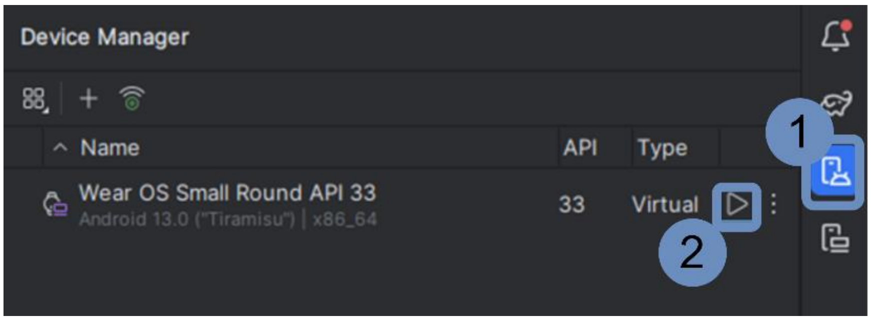

Device Manager
Name
API
Type
Wear OS Small Round API 33
Android 13.0 ("Tiramisu") | x86_64
33
Virtual
1
2

Figure 7. Launching an emulator through Device Manager

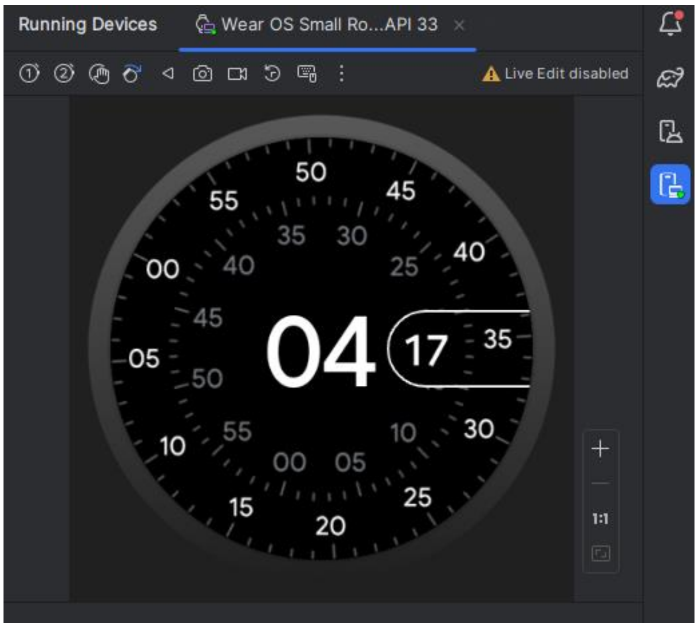


Figure 8. Emulator without an application in the Running Devices window

Once the emulator is running, we click the "play" button in the upper menu, and SmartBioStream will be launched on the emulator, as Figure 9 depicts.

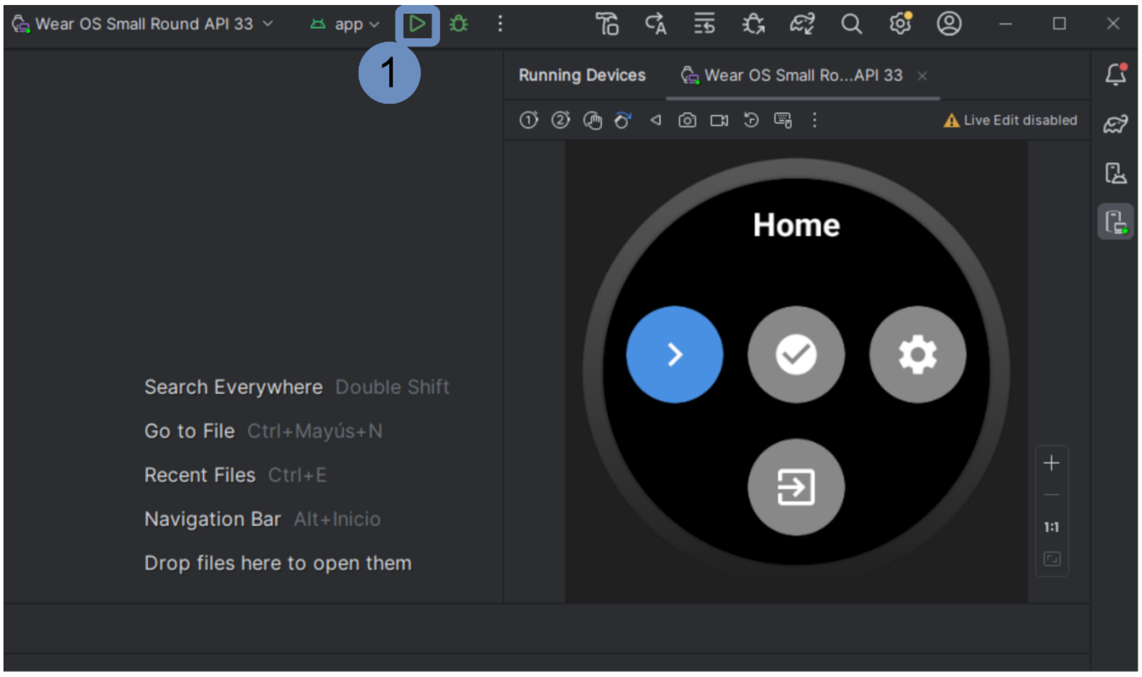

Wear OS Small Round API 33
app
1
Running Devices
Wear OS Small Ro...API 33
Live Edit disabled
Search Everywhere Double Shift
Go to File Ctrl+Mayús+N
Recent Files Ctrl+E
Navigation Bar Alt+Inicio
Drop files here to open them
Home
1:1

Figure 9. Running SmartBioStream on the smartwatch emulator

Finally, we adjust the values of the emulated smartwatch sensors through the "Extended Controls" button, as Figure 10 illustrates. It is worth noting that some sensors do not work if they are not adjusted from their default values, especially the temperature and heart rate sensors. Furthermore, there are some bugs in the emulators that sometimes cause the set values to be ignored. We recommend avoiding the use of mouse scrolling, as there is a bug in the emulators that can cause them to crash. Emulating the latest version of Wear OS may trigger bug messages such as "Clock keeps stopping." Users can choose to close the application, as these issues are unrelated to SmartBioStream. Nevertheless, we recommend using "Android 13 Tiramisu," as indicated in Figure 6, to avoid this issue.

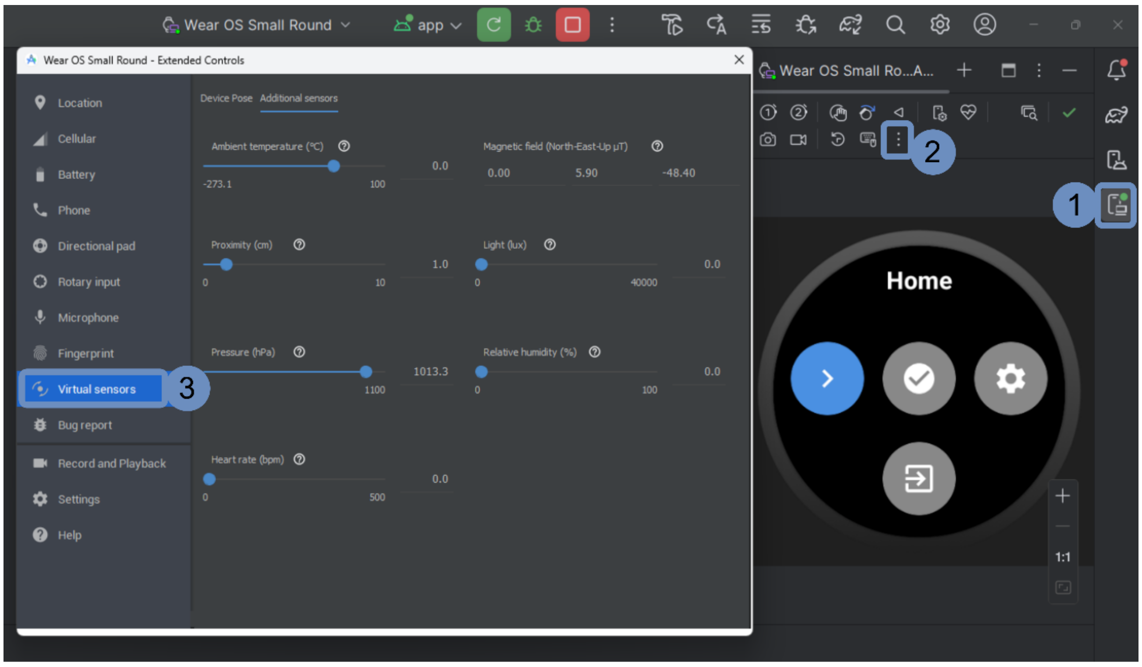


Figure 10. Setting the values of the smartwatch sensors

4. Lauch SmartBioStream on a smartwatch

This section explains how to launch SmartBioStream on a Wear OS smartwatch. More details are available in [3]. Before transferring the application, the smartwatch and the computer on which Android Studio is running must be connected to the same Wi-Fi network. For this section, we have used a Samsung Galaxy Watch 6 as an example, but the steps are similar for other smartwatches with Wear OS.

4.1. Activate developer mode on Wear OS

To activate developer mode, we must access the Wear OS settings menu. This menu can be accessed in different ways, for example, through the application list menu or the quick access menu (accessible by swiping up or down from the watch face). From the settings menu, we must navigate to "About watch," then to "Software information," and tap three times on "Software version." At this point, developer mode has been activated. Figure 11 summarizes this process.

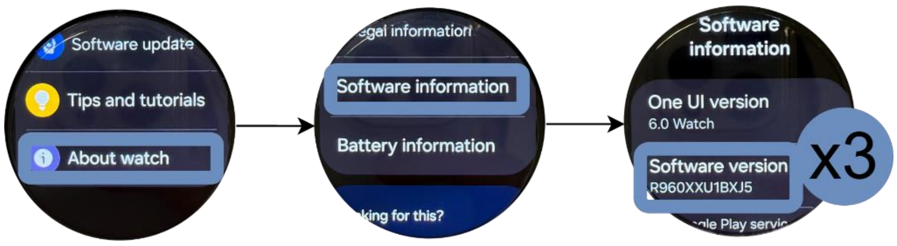


Figure 11. Enable the developer mode on Wear OS

4.2. Send SmartBioStream to a smartwatch

To send the application to the smartwatch, we need to return to the settings menu. From there, we navigate to the newly available "Developer options" and enable "ADB debugging." Next, we enter "Wireless debugging" and confirm the prompt, as shown in Figure 12.

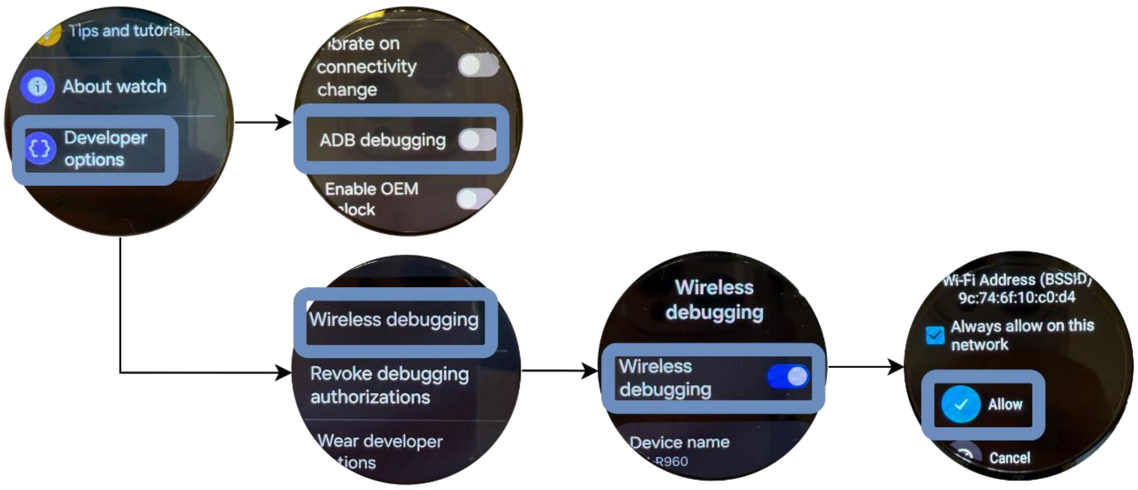


Figure 12. Connecting a Wear OS device to Android Studio

In Android Studio, with the project already imported, we access the Device Manager through the third button in the right-side menu. Inside the Device Manager, we click the button displaying a plus symbol with waves, as shown in Figure 13.

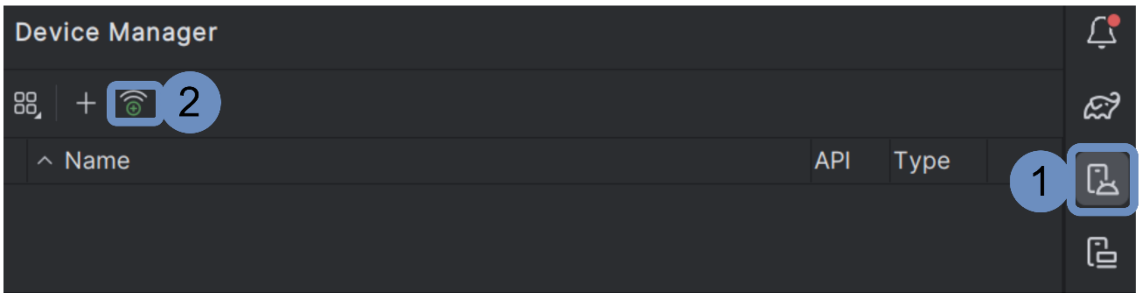
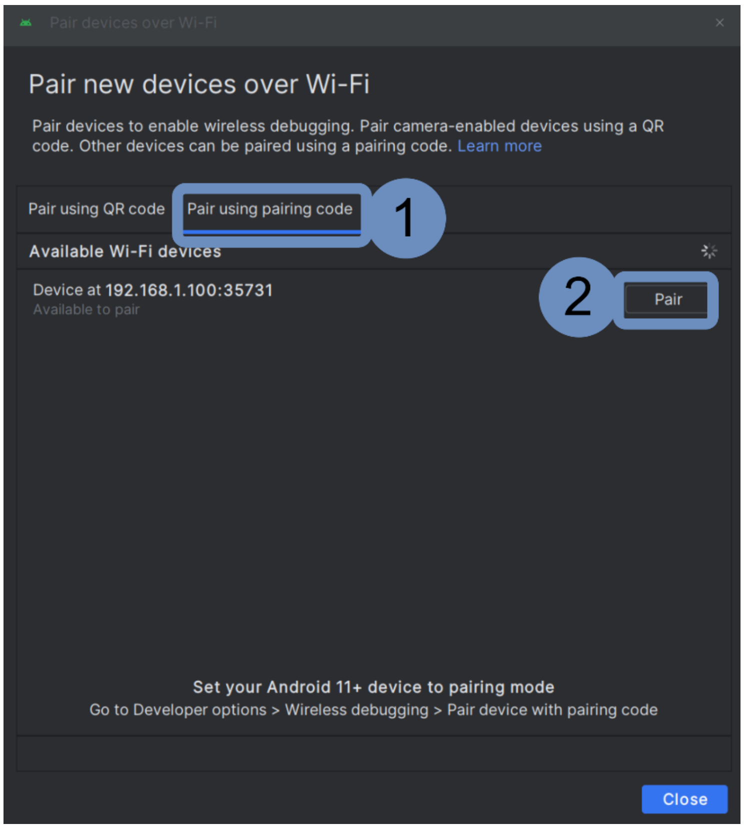


Figure 13. Access to "Pair devices over Wi-Fi" in Android Studio

Within the Pair devices over Wi-Fi menu, we select "Pair using pairing code." As shown in Figure 14, the smartwatch should be visible. We click "Pair" and enter the authentication code displayed on the smartwatch screen.


Pair devices over Wi-Fi

Pair new devices over Wi-Fi

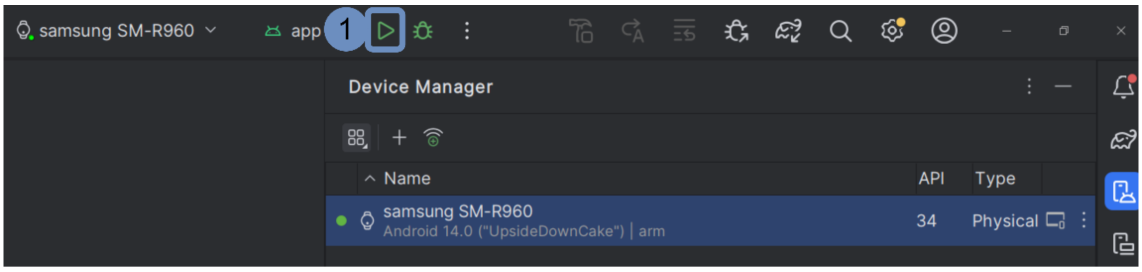
Pair devices to enable wireless debugging. Pair camera-enabled devices using a QR code. Other devices can be paired using a pairing code. Learn more

Device at 192.168.1.100:35731
Available to pair

Set your Android 11+ device to pairing mode
Go to Developer options > Wireless debugging > Pair device with pairing code

Figure 14. Pairing a new smartwatch in Android Studio

Once the device is paired, we close the “Pair devices over Wi-Fi” menu. After pairing the device, it should appear within Device Manager, as shown in Figure 15. In the top menu of Android Studio, we click the green “play” button, which will send the application to the smartwatch. From this moment, the application will be stored on the smartwatch and accessible alongside other applications from the applications inventory.



Name	API	Type
samsung SM-R960 Android 14.0 ("UpsideDownCake") | arm	34	Physical

Figure 15. Launching SmartBioStream on a smartwatch

5. Deploy ServerBioStream

We provide an additional manual on how to deploy ServerBioStream. In this section, we summarize the essential features and steps for its deployment. ServerBioStream is a Django web application that receives and stores the data collected from SmartBioStream. We have developed this tool to provide access to the data gathered during the various experiments of a case study. This platform is intended for researchers and administrators rather than the participants of the experiments. Consequently, it provides the following functionalities:

- **JSON API**. This API receives the data emitted by one or multiple smartwatches. The messages are processed, and each measurement is stored in the SQLite database. The API is developed considering the first use case where the end-users are participants of a case study and only provide their experiment identifier.

- **User management**. ServerBioStream considers two roles: researchers and administrators. Researchers can connect to the platform and download the stored data. Administrators are the only ones who can create new users. Both of them have to authenticate with a username and password before accessing the web platform.

- **Data monitoring**. Researchers and administrators can visualize the data received and stored for each experiment. This feature also helps ensure the proper collection of the biometrics during the experiments. It is worth noting that SmartBioStream queues several measurements in the same data transmission, which means that there may be a delay of less than a minute in the data display.

- **Data export**. The users of this platform can download the data collected from each experiment in CSV, XLSX, and PDF formats. This functionality enables the researchers to develop their specific analyses with the necessary software, such as Excel and Python.

The first step to deploy ServerBioStream is to install Python and the required dependencies. We provide a requirements file to facilitate the installation of the Python libraries used by ServerBioStream. Therefore, it is only necessary to run *pip install -r requirements.txt* in the project folder. Once the installation is finished, we can deploy ServerBioStream using the command *python manage.py runserver ip_address:port*. By default, the *ip_address* should be the IP of the machine where ServerBioStream is running. For the port, we recommend using 8000 for HTTP connections and 8080 for HTTPS.

On Windows, the machine's IP address can be obtained by opening a terminal, running the command *ipconfig*, and looking for the IPv4 address under the active connection (Wi-Fi or Ethernet). Alternatively, it can be found visually in the "Control Panel." On Ubuntu, the IP address can be retrieved by opening a terminal and running *ifconfig*, then checking the IP associated with *eth0* (wired connection) or *wlan0* (Wi-Fi). It can also be found through the Settings menu under Network. On MacOS, the IP address can be obtained by opening a terminal, running *ifconfig*, and looking for the IP under *en0* (Wi-Fi) or *en1*

(Ethernet). It is also possible to visualize the IP graphically by going to "System Settings," selecting "Network," and choosing the active connection.

Finally, we can access ServerBioStream through a web browser using the URL "http://ip_address:port". By default, the login menu will be displayed. The default database includes a user with the username "admin" and password "admin." We strongly recommend changing this password for security reasons.

6. Configuration of SmartBioStream

Once ServerBioStream is launched, we need to configure SmartBioStream to connect to it. These settings should be configured by a researcher or administrator, not by the end user. Furthermore, these settings are stored, so they only need to be changed if the ServerBioStream configuration changes.

First, we must identify the IP address and port where ServerBioStream is accessible. By default, the IP corresponds to the server or computer's IP address, and the port is set to 8000. In SmartBioStream, we access the settings menu, as shown in Figure 16. Then, we set the IP and port of the server. The next view asks for the protocol to communicate with the server. By default, ServerBioStream is launched with HTTP, but through NGINX or by providing SSL certificates, it can support HTTPS. If the certificates are self-signed and not provided by a reputable source, we must deactivate CA verification.

Finally, we should select the authentication method. For ServerBioStream, we should select "identifier (ID)" instead of "username and password." To use SmartBioStream with another platform where smartwatch users must authenticate, we need to set the authentication method to "username and password". We can then check the connection with the server through the second button on the home screen.

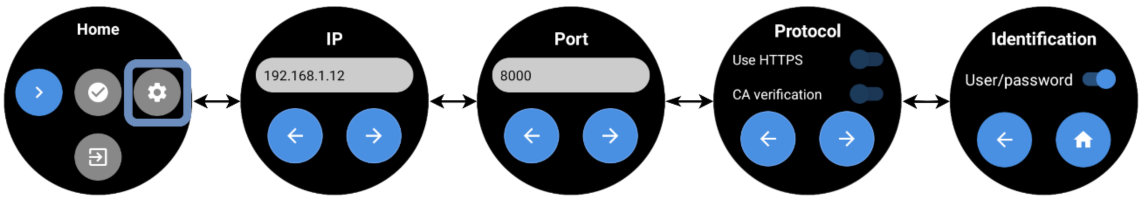


Figure 16. SmartBioStream settings

7. User manual

To start data collection, we press the first button in the home menu. In the next view, we need to enter the experiment identifier, which must be provided by a researcher. This identifier will be used to group the data in ServerBioStream. If username and password authentication has been selected in the settings, a login view will appear for entering these credentials. The next view corresponds to data collection. By default, data collection is paused and must be started by pressing the green "play" button. From this view, you can return to the main menu or change the sensor selection. Figure 17 shows the data collection views in SmartBioStream.

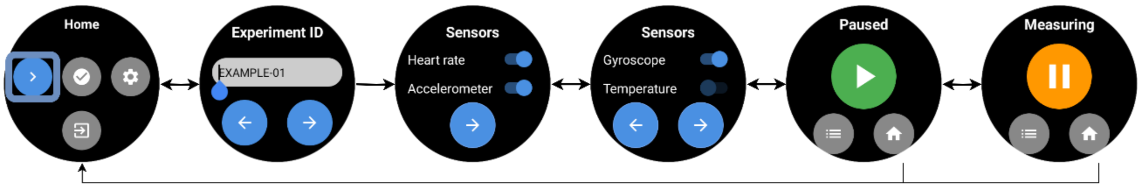
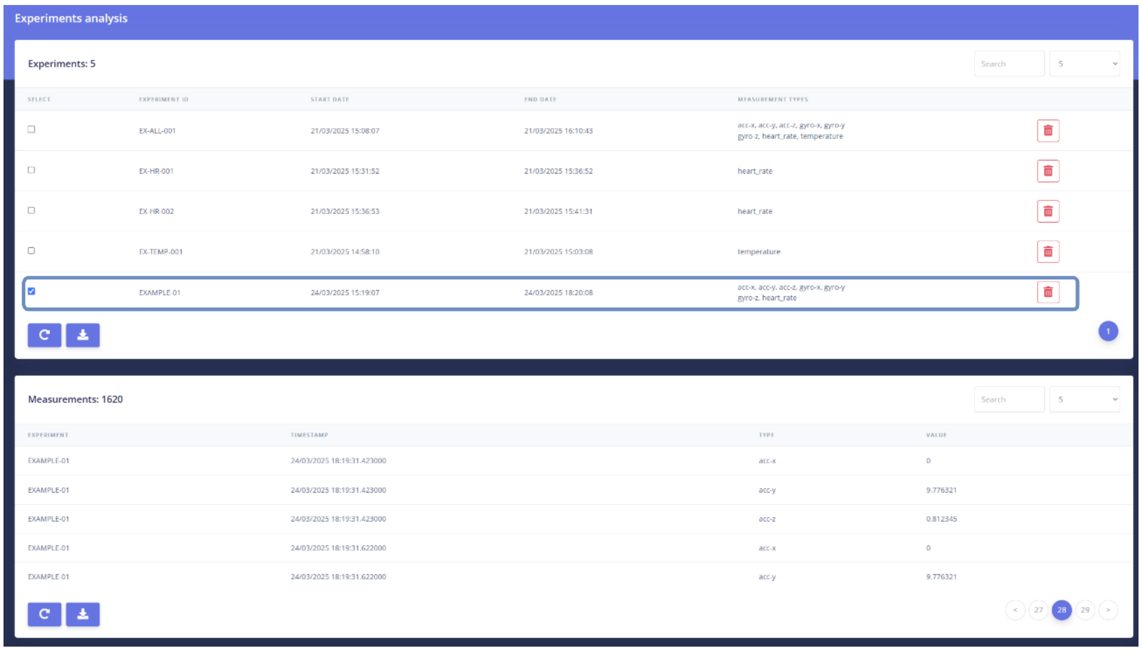


Figure 17. Data collection through SmartBioStream

After logging into ServerBioStream, we are redirected to the experiment view (Figure 18). This view displays two tables: the first summarizes all the experiments, and the second shows the collected data for the experiments selected in the first table. We can refresh and export each table using the buttons located at the end of the tables. It is worth noting that SmartBioStream queues several measurements in the same data transmission, which means there may be a delay of less than a minute in the data display. If the user is testing the SmartBioStream through an emulator, we adjust the values of the emulated smartwatch sensors using the "Extended Controls" button, as illustrated in Figure 10.



Experiments analysis

Experiments: 5

SELECT	EXPERIMENT ID	START DATE	END DATE	MEASUREMENT TYPES
☐	EX-ALL-001	21/03/2025 15:08:07	21/03/2025 16:10:43	acc-x, acc-y, acc-z, gyro-x, gyro-y gyro-z, heart_rate, temperature
☐	EX-HR-001	21/03/2025 15:31:52	21/03/2025 15:36:52	heart_rate
☐	EX-HR-002	21/03/2025 15:36:53	21/03/2025 15:41:31	heart_rate
☐	EX-TEMP-001	21/03/2025 14:58:10	21/03/2025 15:03:08	temperature
☑	EXAMPLE-01	24/03/2025 15:19:07	24/03/2025 18:20:08	acc-x, acc-y, acc-z, gyro-x, gyro-y gyro-z, heart_rate

Measurements: 1620

EXPERIMENT	TIMESTAMP	TYPE	VALUE
EXAMPLE-01	24/03/2025 18:19:31.423000	acc-x	0
EXAMPLE-01	24/03/2025 18:19:31.423000	acc-y	9.776321
EXAMPLE-01	24/03/2025 18:19:31.423000	acc-z	0.812345
EXAMPLE-01	24/03/2025 18:19:31.622000	acc-x	0
EXAMPLE-01	24/03/2025 18:19:31.622000	acc-y	9.776321

Figure 18. Data analysis through ServerBioStream

8. References

[1] Android Developers Team, “Android Studio,” https://developer.android.com/studio (accessed: March 25, 2025).

[2] Android Developers Team, “Managing AVDs,” https://developer.android.com/studio/run/managing-avds (accessed: March 25, 2025).

[3] Android Developers Team, “Creating a Wear OS App,” https://developer.android.com/training/wearables/get-started/creating (accessed: March 25, 2025).