



GUÍA COMPLETA: Health Connect Export App

Resumen del Proyecto

Objetivo: Exportar automáticamente datos de salud desde Samsung Health (vía Health Connect) a Google Drive en formato JSON.

Repositorio: <https://github.com/lw2die/health-connect-export>

Tags importantes:

- v1.0-export-auto - Export automático básico
- v1.2.1-fix-compilation - Fix de errores de compilación
- v1.3.0-refactor-serializer - Refactoring con HealthDataSerializer
- v1.4.0-vo2max - Soporte de VO2 Max

Funcionalidades Actuales

Export Automático

- Ejecuta cada 30 minutos (5 AM - 9 PM)
- Solo exporta cambios incrementales (diferencial)
- Guarda en Downloads/HealthConnectExports/
- Formato: health_data_SAMSUNG_AUTO_DIFF_YYYY-MM-DD_HH-mm.json

Export Manual

- Botón en la pantalla principal
- Exporta últimos 30 días completos
- Abre Google Drive para guardar
- Formato: health_data_SAMSUNG_MANUAL_YYYY-MM-DD_HH-mm.json

Datos Exportados

- **Weight** (Peso)
- **Exercise Sessions** (Sesiones de ejercicio con métricas detalladas)
- **Sleep Sessions** (Sesiones de sueño con stages)
- **VO2 Max** (Capacidad aeróbica)

Todos filtrados por com.sec.android.app.shealth para evitar duplicados.

Cómo Agregar una Nueva Métrica

Cada vez que quieras agregar una nueva métrica (Blood Pressure, Body Temperature, Heart Rate Variability, etc.), seguí estos 6 pasos:

1. AndroidManifest.xml

Agregá los permisos de lectura/escritura:



xml

```
<uses-permission android:name="android.permission.health.READ_[NOMBRE_METRICA]"/>
<uses-permission android:name="android.permission.health.WRITE_[NOMBRE_METRICA]"/>
```

Ejemplo Blood Pressure:



xml

```
<uses-permission android:name="android.permission.health.READ_BLOOD_PRESSURE"/>
<uses-permission android:name="android.permission.health.WRITE_BLOOD_PRESSURE"/>
```

2. InputReadings ViewModel.kt

Agregá el import y el permiso:



kotlin

```
import androidx.health.connect.client.records.[NombreRecord]

val permissions = setOf(
    // ... permisos existentes ...
    HealthPermission.getReadPermission([NombreRecord]::class)
)
```

Ejemplo Blood Pressure:



kotlin

```
import androidx.health.connect.client.records.BloodPressureRecord
```

```
val permissions = setOf(  
    // ... otros permisos ...  
    HealthPermission.getReadPermission(BloodPressureRecord::class)  
)
```

3. HealthConnectManager.kt

Agregá función para leer la métrica **ANTES** de la última llave:



kotlin

```
suspend fun read[Nombre]Records(start: Instant, end: Instant): List<[NombreRecord]> {  
    val request = ReadRecordsRequest(  
        recordType = [NombreRecord]::class,  
        timeRangeFilter = TimeRangeFilter.between(start, end)  
    )  
    val response = healthConnectClient.readRecords(request)  
    return response.records  
}
```

Ejemplo Blood Pressure:



kotlin

```
suspend fun readBloodPressureRecords(start: Instant, end: Instant): List<BloodPressureRecord> {  
    val request = ReadRecordsRequest(  
        recordType = BloodPressureRecord::class,  
        timeRangeFilter = TimeRangeFilter.between(start, end)  
    )  
    val response = healthConnectClient.readRecords(request)  
    return response.records  
}
```

4. HealthDataSerializer.kt

A) Agregá función de serialización (privada):



kotlin

```
private fun serialize[Nombre]Data(records: List<[NombreRecord]>): String {
    if(records.isEmpty()) return "[]"
    return buildString {
        append("[\n")
        records.forEachIndexed { index, record ->
            append("    {\n")
            append("        \"timestamp\": \"${record.time}\",\n")
            // Agregá los campos específicos de la métrica
            append("        \"campo1\": ${record.campo1},\n")
            append("        \"campo2\": ${record.campo2},\n")
            append("        \"source\": \"${record.metadata.dataOrigin.packageName}\"\n")
            append("    }")
            if(index < records.size - 1) append(",")
            append("\n")
        }
        append("]")
    }
}
```

B) Modificá generateHealthJSON para incluir el nuevo parámetro:



kotlin

```

fun generateHealthJSON(
    weightRecords: List<WeightData>,
    exerciseData: List<Map<String, Any?>>,
    sleepData: List<Map<String, Any?>>,
    vo2MaxRecords: List<Vo2MaxRecord>,
    [nombreMetrica]Records: List<[NombreRecord]>, // AGREGAR
    exportType: String
): String {
    return buildString {
        // ... código existente ...

        // AGREGAR nueva sección
        append("  \"[nombre_metrica]_records\": {\n")
        append("    \"count\": ${[nombreMetrica]Records.size},\n")
        append("    \"data\": ")
        append(serialize[Nombre]Data([nombreMetrica]Records))
        append("\n  }\n")

        append("{}")
    }
}

```

5. InputReadingsScreen.kt

En la función `shareWeightDataToGoogleDrive()`, agrega:



kotlin

// Leer y filtrar datos

```
val [nombreMetrica]Records = if (healthConnectManager != null) {
    healthConnectManager.read[Nombre]Records(startTime, endTime).filter {
        it.metadata.dataOrigin.packageName == "com.sec.android.app.shealth"
    }
} else {
    emptyList()
}
```

// Actualizar llamada al serializer

```
val jsonContent = HealthDataSerializer.generateHealthJSON(
    filteredWeightRecords,
    exerciseData,
    sleepData,
    vo2MaxRecords,
    [nombreMetrica]Records, // AGREGAR
    "MANUAL_SAMSUNG_ONLY"
)
```

// Actualizar Toast

```
Toast.makeText(
    context,
    "Export: ... + ${[nombreMetrica]Records.size} [Nombre]",
    Toast.LENGTH_LONG
).show()
```

6. JsonExportWorker.kt

A) Agregá función de lectura filtrada:



kotlin

```
private suspend fun read[Nombre]RecordsFiltered(
    startTime: Instant,
    endTime: Instant
): List<[Nombre]Record> {
    val all[Nombre]Records = healthConnectManager.read[Nombre]Records(startTime, endTime)
    return all[Nombre]Records.filter {
        it.metadata.dataOrigin.packageName == "com.sec.android.app.shealth"
    }
}
```

B) En doWork(), agregá lectura:



kotlin

```
val [nombreMetrica]Data = read[Nombre]RecordsFiltered(startTime, endTime)
```

C) Actualizar llamada al serializer:



kotlin

```
val jsonContent = HealthDataSerializer.generateHealthJSON(  
    weightData,  
    exerciseData,  
    sleepData,  
    vo2MaxData,  
    [nombreMetrica]Data, // AGREGAR  
    "AUTO_SAMSUNG_ONLY_WORKER"  
)
```

Checklist Rápida

Para cualquier métrica nueva:

- ☐ **AndroidManifest.xml** → Permisos READ/WRITE
- ☐ **InputReadingsViewModel.kt** → Import + Permiso
- ☐ **HealthConnectManager.kt** → Función read[Nombre]Records()
- ☐ **HealthDataSerializer.kt** → Función serialize + Modificar generateHealthJSON
- ☐ **InputReadingsScreen.kt** → Leer + Filtrar + Pasar al serializer
- ☐ **JsonExportWorker.kt** → Función filtrada + Pasar al serializer

Guía de Git

Configuración Inicial (Una sola vez)



bash

Configurar nombre y email

```
git config --global user.name "Tu Nombre"
```

```
git config --global user.email "tu@email.com"
```

Verificar configuración

```
git config --list
```

Comandos Básicos del Día a Día

Ver estado del proyecto



bash

```
git status
```

Agregar cambios al staging



bash

Agregar todos los archivos modificados

```
git add .
```

Agregar un archivo específico

```
git add nombre_archivo.kt
```

Hacer commit



bash

```
git commit -m "Descripción clara del cambio"
```

Ejemplos de buenos commits:



bash


```
git commit -m "feat: Add VO2 Max support"
git commit -m "fix: Resolve compilation error in HealthDataSerializer"
git commit -m "refactor: Extract serialization logic to separate object"
git commit -m "docs: Update README with installation instructions"
```

Subir cambios a GitHub



bash

Primera vez

```
git push -u origin main
```

Después

```
git push
```

Trabajar con Tags (Versiones)

Crear tag



bash

```
git tag v1.4.0-vo2max
```

Ver todos los tags



bash

```
git tag
```

Subir tag a GitHub



bash

```
git push origin v1.4.0-vo2max
```

Subir todos los tags



bash

```
git push --tags
```

Comandos Útiles

Ver historial de commits



bash

```
git log --oneline
```

Ver diferencias (qué cambió)



bash

```
# Ver todos los cambios
```

```
git diff
```

```
# Ver cambios de un archivo específico
```

```
git diff archivo.kt
```

Descartar cambios locales



bash

```
# Descartar cambios de un archivo
```

```
git checkout -- archivo.kt
```

```
# Descartar TODOS los cambios (cuidado!)
```

```
git reset --hard
```

Ver branches



bash

`git branch`

Flujo Completo de Trabajo



bash

1. Ver qué archivos cambiaron

`git status`

2. Agregar todos los cambios

`git add .`

3. Hacer commit

`git commit -m "feat: Add new health metric support"`

4. (Opcional) Crear tag para versión

`git tag v1.5.0-nueva-metrica`

5. Subir a GitHub

`git push origin main`

6. Subir tag

`git push origin v1.5.0-nueva-metrica`

Convenciones de Commits

Prefijos recomendados:

- feat: Nueva funcionalidad
- fix: Corrección de bug
- refactor: Refactoring de código
- docs: Cambios en documentación
- style: Cambios de formato (no afectan funcionalidad)
- test: Agregar o modificar tests
- chore: Tareas de mantenimiento

Ejemplos:



bash

```
git commit -m "feat: Add Blood Pressure export support"
git commit -m "fix: Resolve VO2 Max serialization error"
git commit -m "refactor: Improve HealthDataSerializer performance"
git commit -m "docs: Add metric integration guide to README"
```

Resolución de Problemas Comunes

Error: "Your branch is behind"



bash

```
git pull
```

Error: "Merge conflict"



bash

```
# 1. Abrir los archivos en conflicto
# 2. Resolver manualmente los conflictos
# 3. Agregar los archivos resueltos
git add .
git commit -m "Resolve merge conflicts"
```

Olvidé hacer commit antes de modificar



bash

Guardar cambios temporalmente

`git stash`

Hacer lo que necesites hacer

Recuperar cambios

`git stash pop`

Recursos Adicionales

- Documentación oficial de Git: <https://git-scm.com/doc>
- GitHub Guides: <https://guides.github.com/>
- Health Connect API: <https://developer.android.com/health-and-fitness/guides/health-connect>

Última actualización: 4 de Octubre, 2025

Versión actual: v1.4.0-vo2max