unleash your best

Google Fit Developer Challenge Guide for miCoach MultiSport Integration

adidas

micoach

introduction

This guide is a resource for developers competing in the Google Fit Developer Challenge. It will cover everything you need to know to integrate your app with adidas X_CELL sensors and Google Fit.
Find all the challenge rules at **g.co/FitDeveloperChallenge**

what you'll need

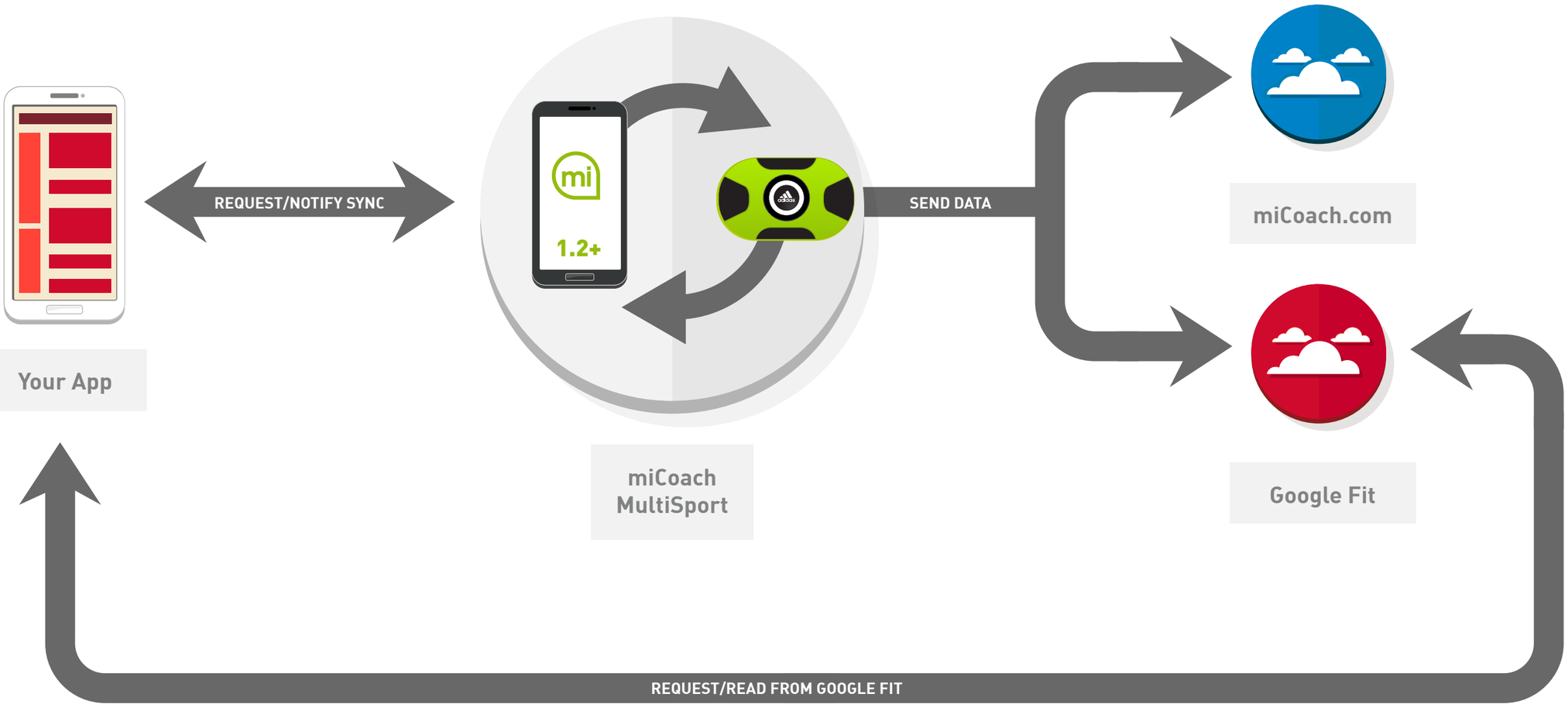
miCoach MultiSport beta app

adidas X_CELL sensor

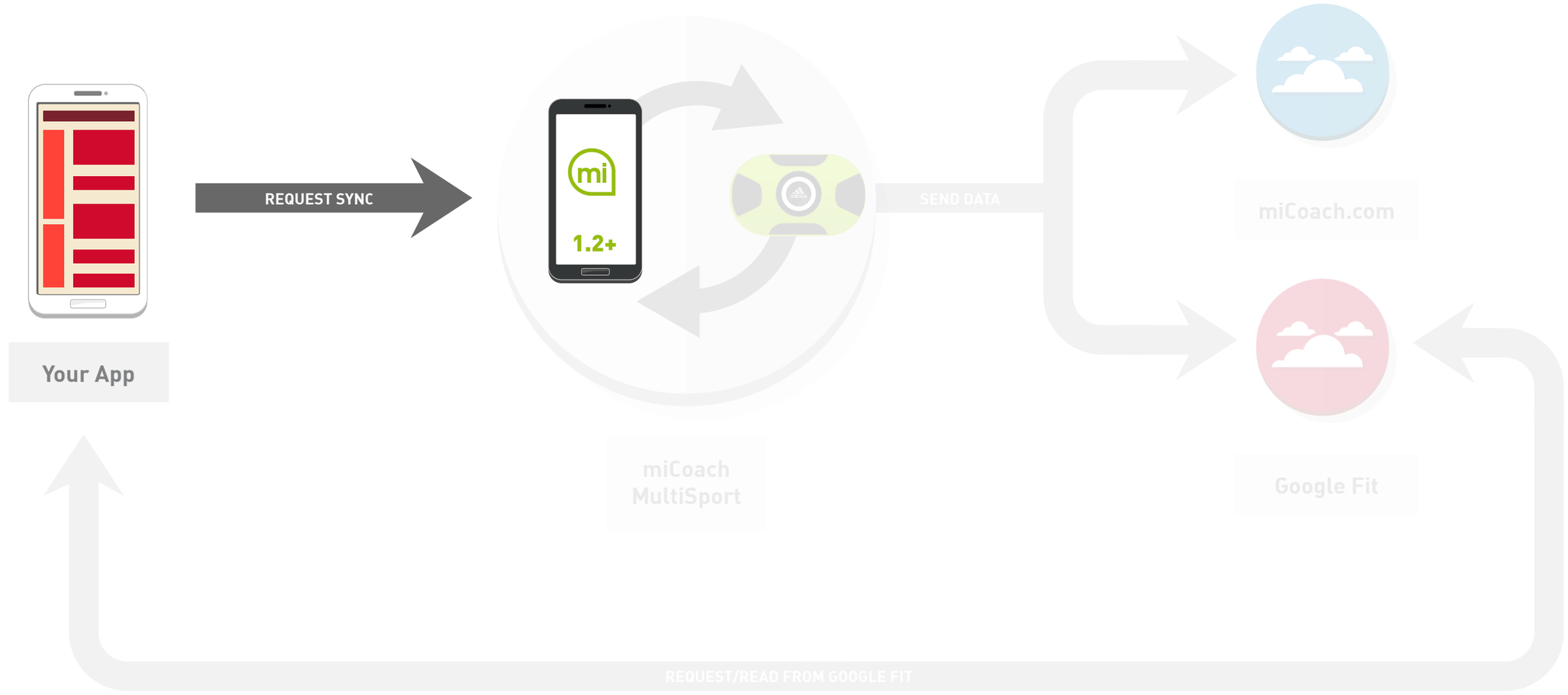
Google Fit documentation

Android development device with Bluetooth LE and Android 4.3+

how it works

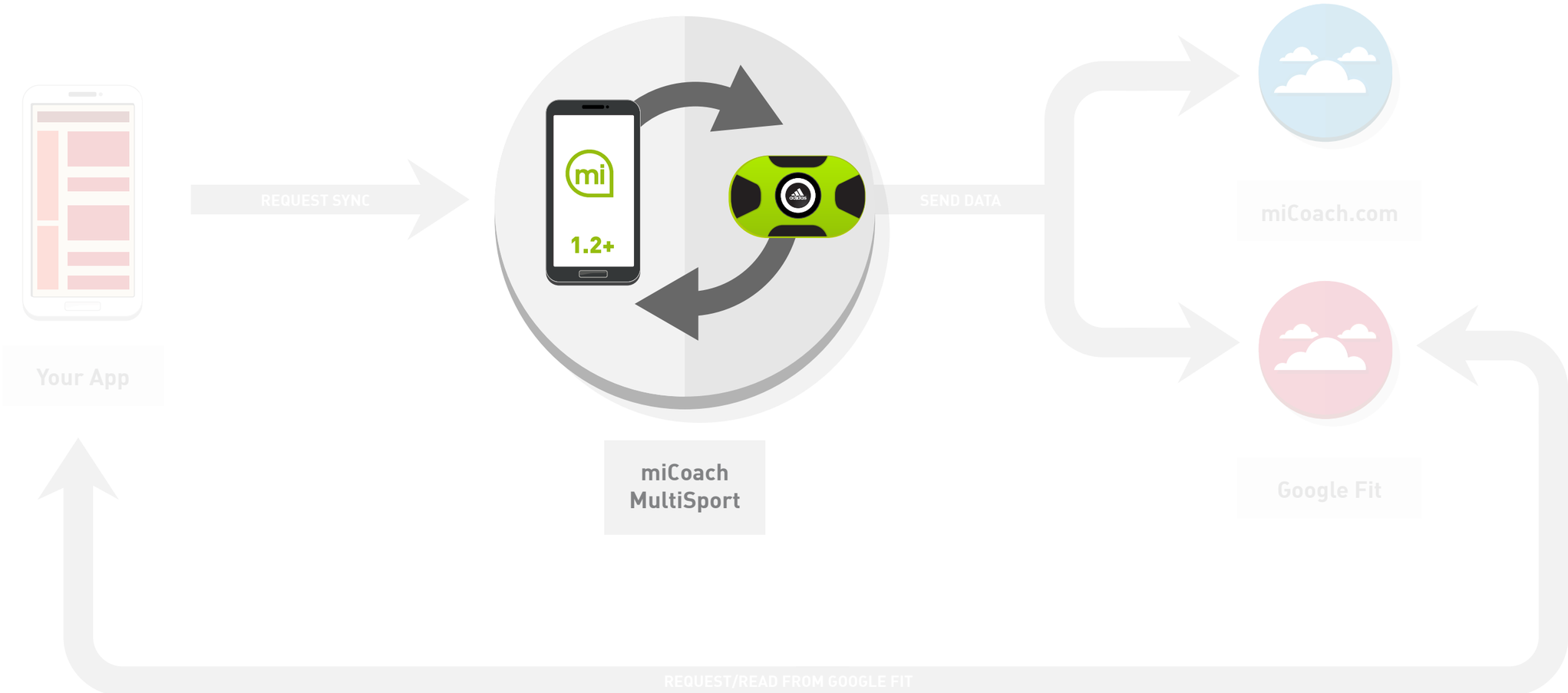

REQUEST/NOTIFY SYNC
1.2+
SEND DATA
miCoach.com
Your App
miCoach
MultiSport
Google Fit
REQUEST/READ FROM GOOGLE FIT

how it works



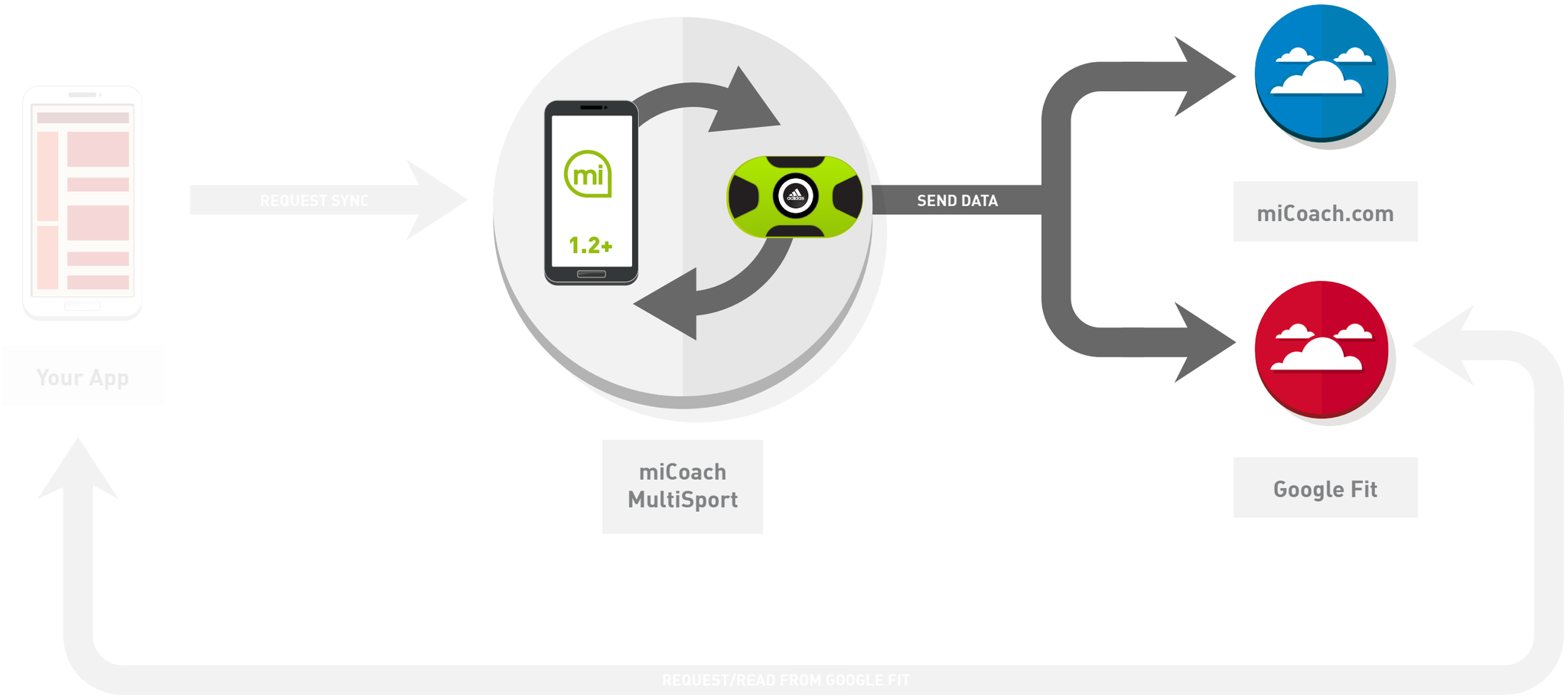
Your app requests a sync from miCoach MultiSport

how it works



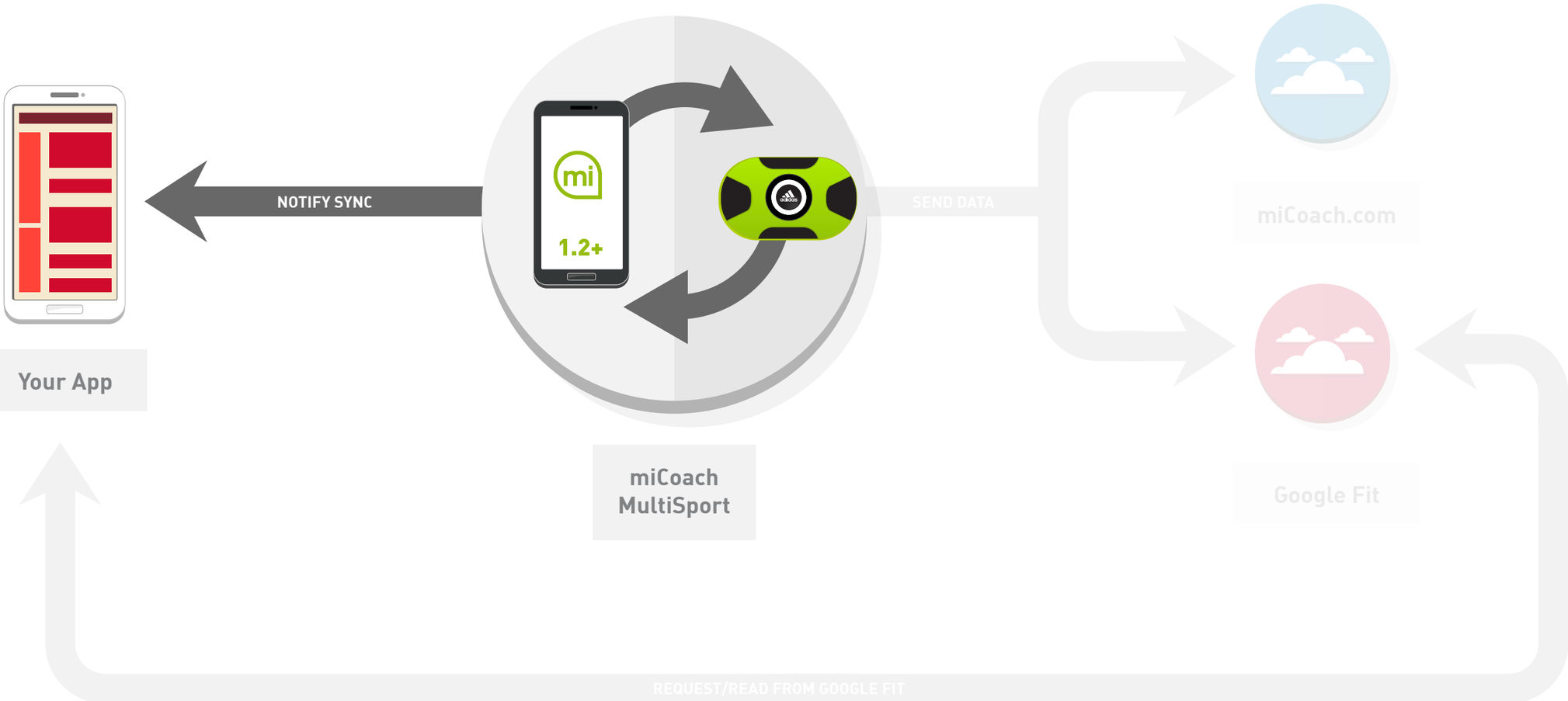
miCoach MultiSport retrieves the latest workout data from the paired adidas X_CELL sensor

how it works



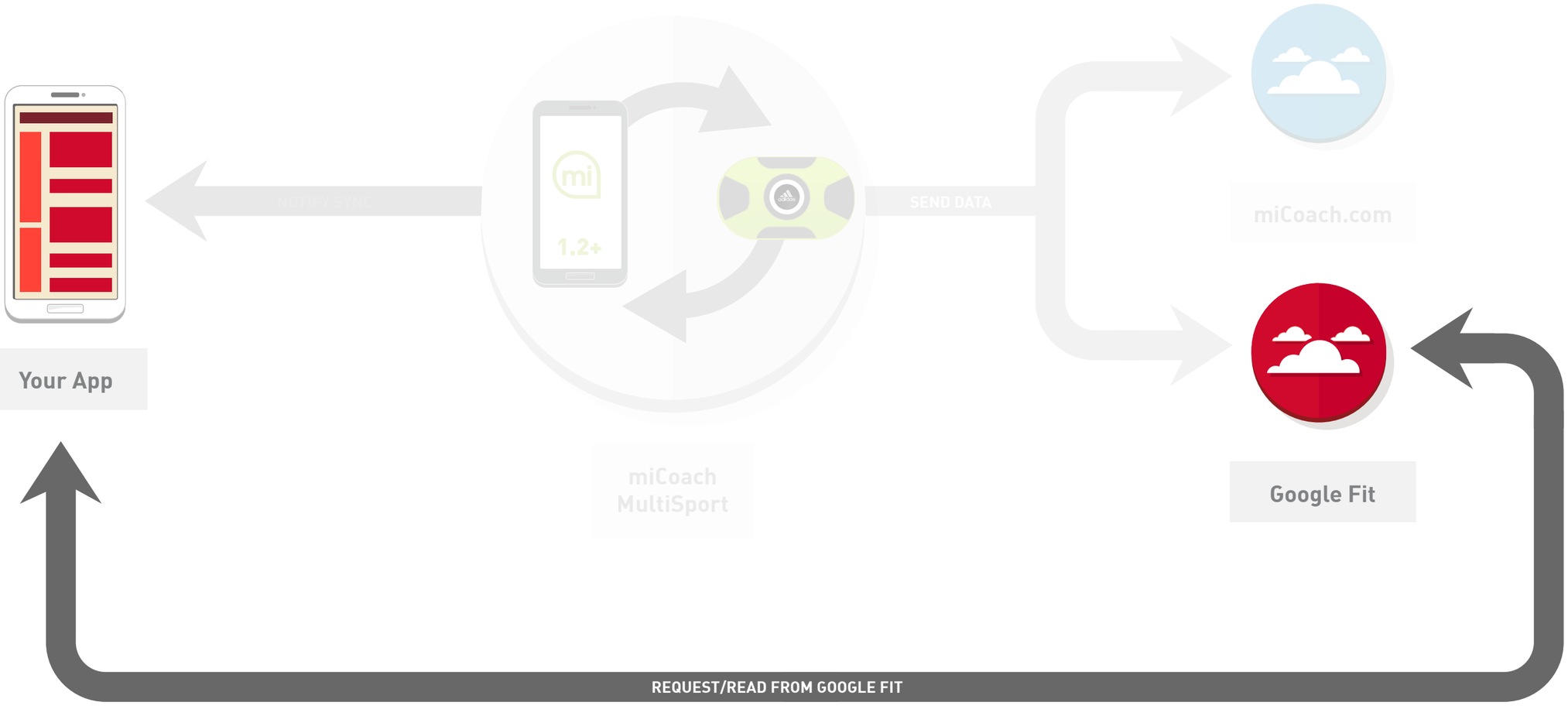
miCoach MultiSport syncs the data to miCoach.com and Google Fit

how it works



miCoach MultiSport notifies your app that the sync completed

how it works



Your app reads the workout data from Google Fit

setting up micoach multisport

adidas

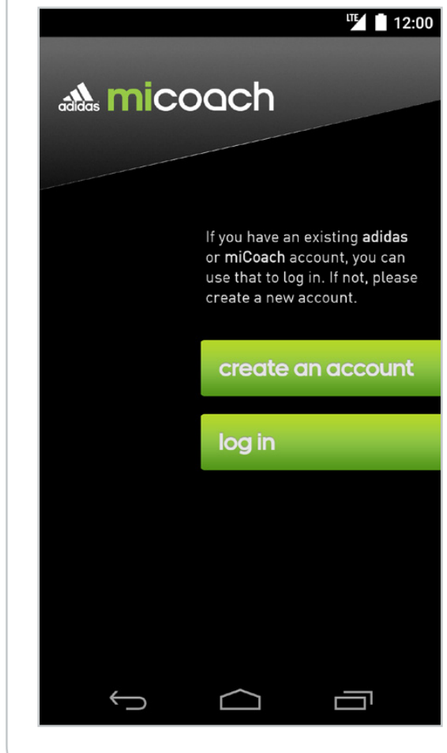
micoach

step 1: enable syncing

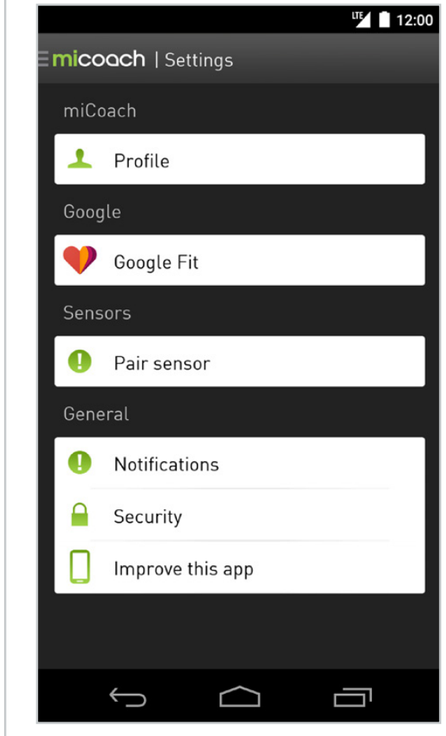
First, download the **miCoach MultiSport** beta app, which you will be able to access once you opt in to your invitation to be a tester.

step 1: enable syncing

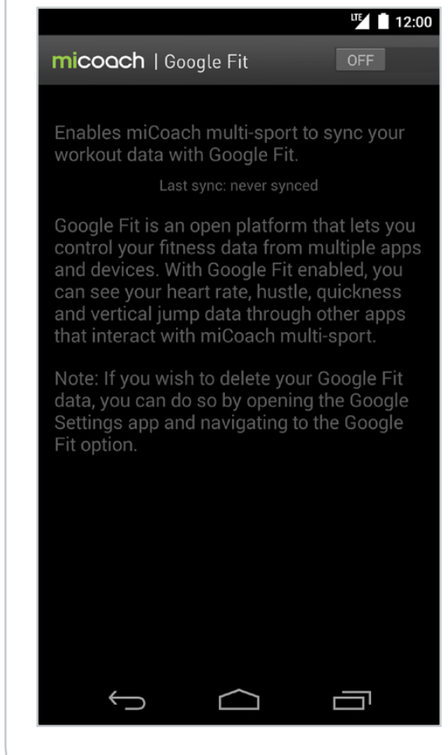
1. Log in or create an account with miCoach



2. Go to your Settings and tap "Google Fit"

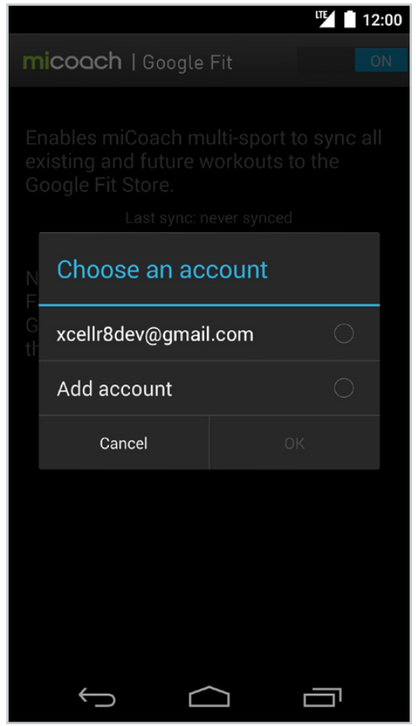


3. Tap the slider to turn syncing on

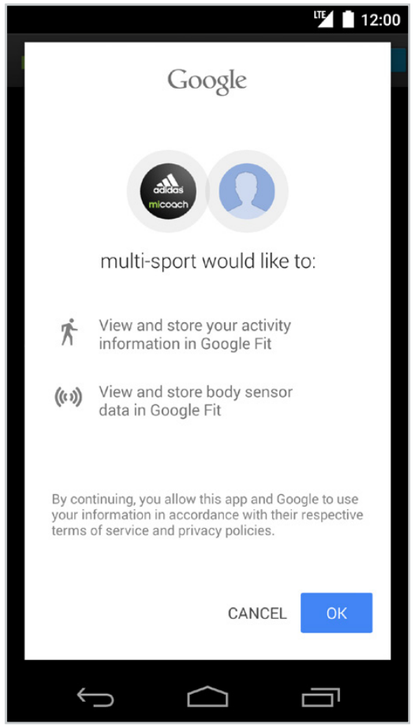


step 1: enable syncing

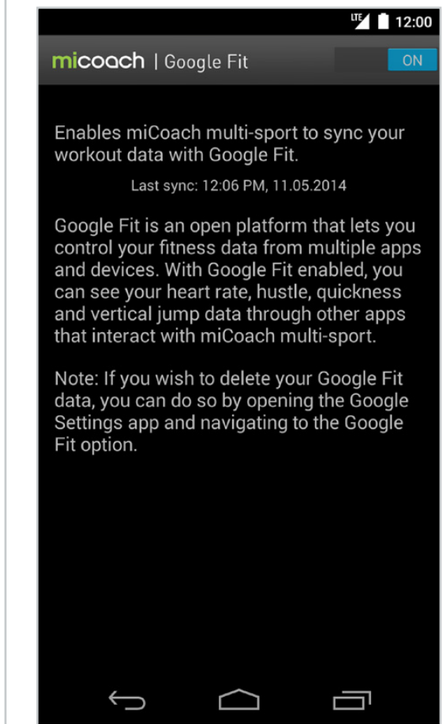
4 Choose the account you wish to sync with



5 Accept the terms



6 Your app should now indicate that syncing is turned on



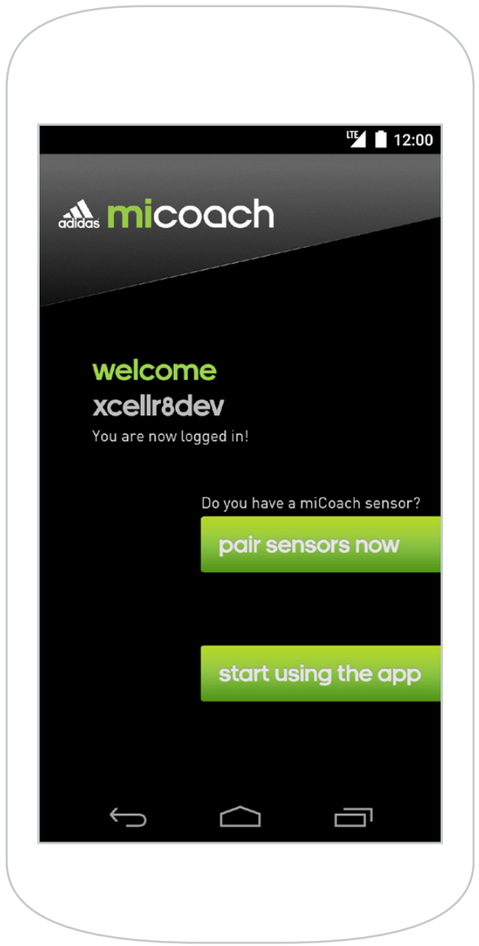
step 2: pair your sensor

miCoach MultiSport pairs with your sensor via Bluetooth so ensure you have Bluetooth enabled on your device.

Additional information and troubleshooting can be found in the **miCoach user manual**.

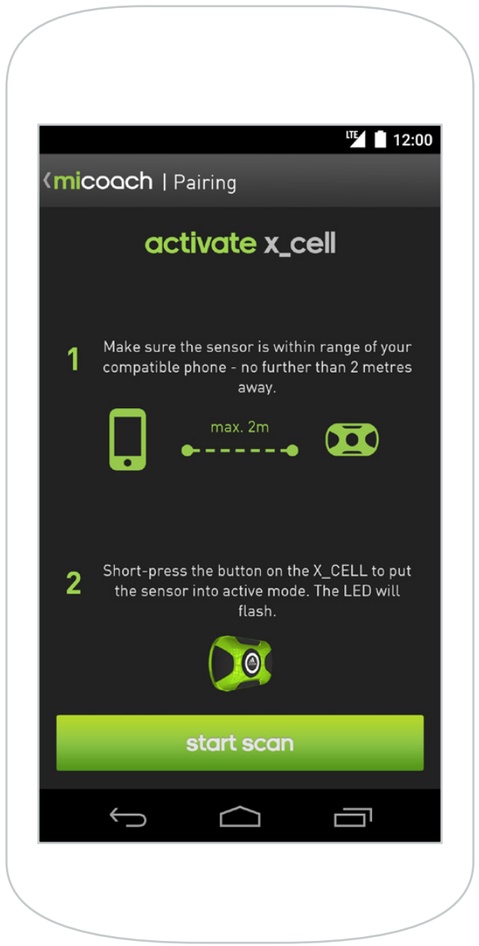
step 2: pair your sensor

1. Tap “pair sensors now” to begin pairing



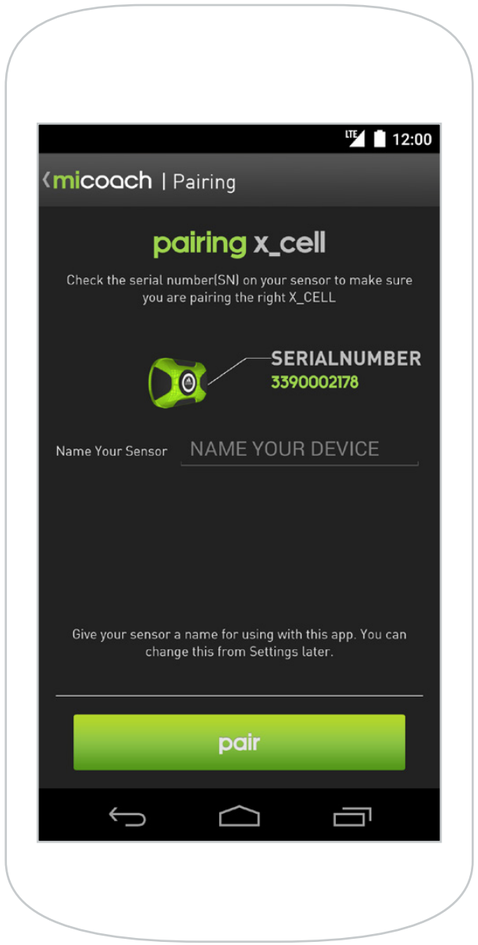
2. Activate your X_CELL by short-pressing the on/off button. You will hear an audible beep and see the lights flash once.

3. Tap “start scan”

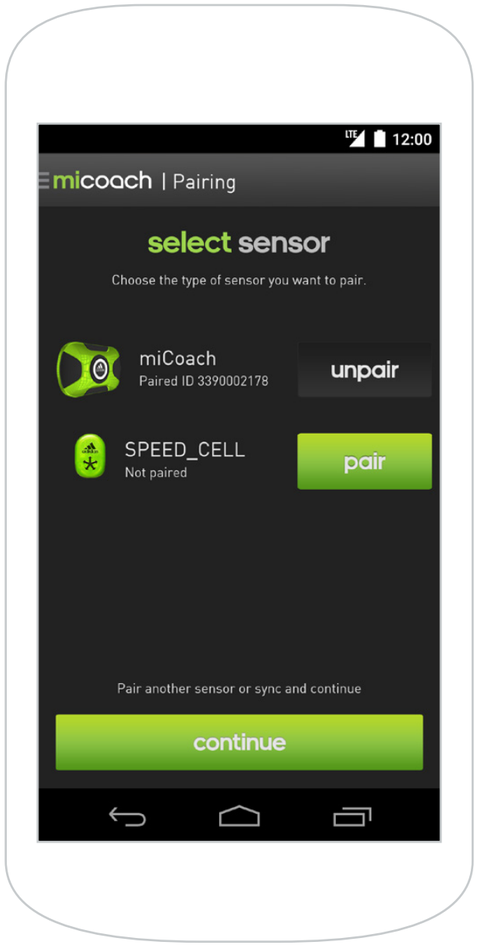


step 2: pair your sensor

4 Once your sensor is detected, select it from the list, enter a name, and tap “pair”



5 miCoach MultiSport should now indicate that your sensor is paired



developing your app

adidas

micoach

step 1: enable google fit

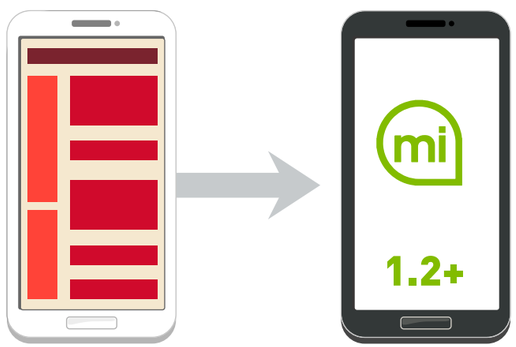
Enable the Google Fit API for your project within the **Google Developers Console** by following the instructions in the **Google Fit getting started** guide.

Note: For an example of a full implementation of a custom app, you can access the **sample app.**

step 2: add google play services

To utilize the Google Play services APIs in your project, refer to the **Google Play services setup** instructions.

step 3: request a sync



To request a sync from miCoach MultiSport, you will need to implement the following components into your project, outlined on the following pages.

- Installation Verification
- Sync Intent
- Results Capturing
- User Sync Interface

step 3: request a sync

Verify App Installation

Before you can send an intent, your app will need to check whether miCoach MultiSport is installed on the user's device.

```
PackageManager pm = context.getPackageManager();
try {
    pm.getPackageInfo("com.adidas.micoach.x_cell", PackageManager.GET_ACTIVITIES);
    return true;
} catch (PackageManager.NameNotFoundException e) {
    return false;
}
```

If miCoach MultiSport is not installed on the device, you can send the user directly to the **Google Play Store** to install it.

```
Uri uri = Uri.parse(
    "market://search?q=pname:com.adidas.micoach.x_cell");

context.startActivity(new Intent(Intent.ACTION_VIEW, uri));
```

step 3: request a sync

Send a Sync Intent

Your app will need to send an intent to miCoach MultiSport.

```
Intent intent = new Intent("com.adidas.micoach.x_cell.sync");
startActivityForResult(intent, SYNC_REQUEST_CODE);
```

step 3: request a sync

Capture Results

Implement `onActivityForResult()` in your Activity or Fragment to receive results from miCoach MultiSport.

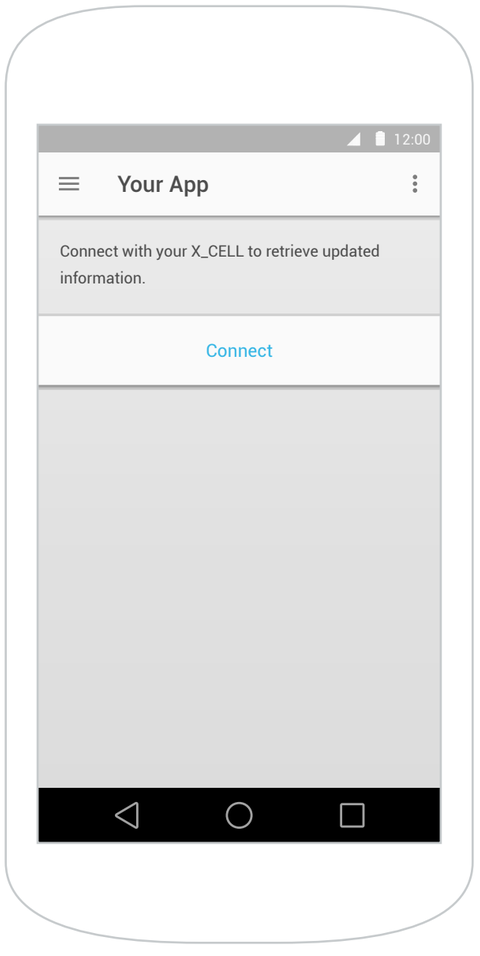
```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (ADIDAS_MSA_REQUEST_CODE != requestCode) {
        return;
    }

    String message = null != data && data.hasExtra(ADIDAS_MSA_RESULT_EXTRA_MESSAGE)
        ? data.getStringExtra(ADIDAS_MSA_RESULT_EXTRA_MESSAGE)
        : null;

    if (message == null) {
        return;
    }

    switch (resultCode) {
        case ADIDAS_MSA_RESULT_SYNC_OK: {
            Log.d(TAG, "Sync succeeded: " + message);
            // TODO: Read data from Google Fit.
            break;
        }
        case ADIDAS_MSA_RESULT_SYNC_FAILED: {
            Log.d(TAG, "Sync failed: " + message);
            // TODO: Notify user of failure, retry, etc.
            break;
        }
        case ADIDAS_MSA_RESULT_NOT_ENABLED: {
            Log.d(TAG, "Not authenticated: " + message);
            // TODO: Notify user to authenticate to Google Fit in miCoach MultiSport.
            break;
        }
    }
}
```

step 3: request a sync

Create a Sync Interface

Once you have these components implemented, you will want to provide an interface for the user to initiate a sync within your app.

example



step 3: request a sync

During the sync process, miCoach MultiSport sends the data from the adidas X_CELL sensor to Google Fit. Your app must be equipped to handle the possible results outlined in the table.

result	description
-1	Workout data is available in Google Fit
1	Workout data could not be synced with Google Fit
2	User has not logged in to miCoach MultiSport or Google Fit integration is not enabled

step 4: retrieve adidas workout data

Once your data is successfully synced, you can retrieve workout data from Google Fit. For example code on how to do this, please refer to the **sample app.**

step 4: retrieve adidas workout data

1 **Connect to the Google Fit service,** which is part of Google Play services.

2 **Send a custom data type request** to Google Fit to retrieve the custom adidas data types: *jump height*, *hustle*, *quickness.* For more information about these data types, reference the **shareable data types.**

Note: *Heart rate* is a DataType object built into the Google Fit API and doesn't require a custom data type request.

3 **Send a session read request** to Google Fit to retrieve the workout data for the authenticated account.

resources

Google Fit Competition Website
g.co/FitDeveloperChallenge

Implementation Sample App
github.com/adidasDigitalSports/MSA-GoogleFit

Google Fit Developer Guide
developers.google.com/fit/android/get-started

Google Play Services Developer Guide
developer.android.com/google/play-services/index.html

miCoach MultiSport app
play.google.com/store/apps/details?id=com.adidas.micoach.x_cell

unleash your best

adidas

micoach