**TRƯỜNG ĐẠI HỌC THỦY LỢI**

**KHOA CÔNG NGHỆ THÔNG TIN**



**GIÁO TRÌNH**

**THỰC HÀNH PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ**

**DI ĐỘNG**

Hà Nội, 2.2025

MỤC LỤC

[CHƯƠNG 1. Làm quen 4](#_Toc190855147)

[Bài 1) Tạo ứng dụng đầu tiên 4](#_Toc190855148)

[1.1) Android Studio và Hello World 4](#_Toc190855149)

[1.2) Giao diện người dùng tương tác đầu tiên 7](#_Toc190855150)

[1.3) Trình chỉnh sửa bố cục 7](#_Toc190855151)

[1.4) Văn bản và các chế độ cuộn 7](#_Toc190855152)

[1.5) Tài nguyên có sẵn 7](#_Toc190855153)

[Bài 2) Activities 7](#_Toc190855154)

[2.1) Activity và Intent 7](#_Toc190855155)

[2.2) Vòng đời của Activity và trạng thái 7](#_Toc190855156)

[2.3) Intent ngầm định 7](#_Toc190855157)

[Bài 3) Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ 7](#_Toc190855158)

[3.1) Trình gỡ lỗi 7](#_Toc190855159)

[3.2) Kiểm thử đơn vị 7](#_Toc190855160)

[3.3) Thư viện hỗ trợ 7](#_Toc190855161)

[CHƯƠNG 2. Trải nghiệm người dùng 8](#_Toc190855162)

[Bài 1) Tương tác người dùng 8](#_Toc190855163)

[1.1) Hình ảnh có thể chọn 8](#_Toc190855164)

[1.2) Các điều khiển nhập liệu 8](#_Toc190855165)

[1.3) Menu và bộ chọn 8](#_Toc190855166)

[1.4) Điều hướng người dùng 8](#_Toc190855167)

[1.5) RecycleView 8](#_Toc190855168)

[Bài 2) Trải nghiệm người dùng thú vị 8](#_Toc190855169)

[2.1) Hình vẽ, định kiểu và chủ đề 8](#_Toc190855170)

[2.2) Thẻ và màu sắc 8](#_Toc190855171)

[2.3) Bố cục thích ứng 8](#_Toc190855172)

[Bài 3) Kiểm thử giao diện người dùng 8](#_Toc190855173)

[3.1) Espresso cho việc kiểm tra UI 8](#_Toc190855174)

[CHƯƠNG 3. Làm việc trong nền 8](#_Toc190855175)

[Bài 1) Các tác vụ nền 8](#_Toc190855176)

[1.1) AsyncTask 8](#_Toc190855177)

[1.2) AsyncTask và AsyncTaskLoader 9](#_Toc190855178)

[1.3) Broadcast receivers 9](#_Toc190855179)

[Bài 2) Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền 9](#_Toc190855180)

[2.1) Thông báo 9](#_Toc190855181)

[2.2) Trình quản lý cảnh báo 9](#_Toc190855182)

[2.3) JobScheduler 9](#_Toc190855183)

[CHƯƠNG 4. Lưu dữ liệu người dùng 9](#_Toc190855184)

[Bài 1) Tùy chọn và cài đặt 9](#_Toc190855185)

[1.1) Shared preferences 9](#_Toc190855186)

[1.2) Cài đặt ứng dụng 29](#_Toc190855187)

[Bài 2) Lưu trữ dữ liệu với Room 29](#_Toc190855188)

[2.1) Room, LiveData và ViewModel 29](#_Toc190855189)

[2.2) Room, LiveData và ViewModel 29](#_Toc190855190)

3.1) Trinfh gowx loi ……………………………………………………………………...

# LÀM QUEN

## Tạo ứng dụng đầu tiên

### Android Studio và Hello World

Giới thiệu

Trong bài thực hành này, bạn sẽ tìm hiểu cách cài đặt Android Studio, môi trường phát triển Android. Bạn cũng sẽ tạo và chạy ứng dụng Android đầu tiên của mình, Hello World, trên một trình giả lập và trên một thiết bị vật lý.

Những gì Bạn nên biết

Bạn nên có khả năng:

* Hiểu quy trình phát triển phần mềm tổng quát cho các ứng dụng lập trình hướng đối tượng sử dụng một IDE (môi trường phát triển tích hợp) như Android Studio.
* Chứng minh rằng bạn có ít nhất 1-3 năm kinh nghiệm trong lập trình hướng đối tượng, với một phần trong số đó tập trung vào ngôn ngữ lập trình Java. (Các bài thực hành này sẽ không giải thích về lập trình hướng đối tượng hoặc ngôn ngữ Java.

A screenshot of a computer

AI-generated content may be incorrect.

Những gì Bạn sẽ cần:

* Một máy tính chạy Windows hoặc Linux, hoặc một Mac chạy macOS. Xem trang tải xuống Android Studio để biết yêu cầu hệ thống cập nhật.
* Truy cập Internet hoặc một phương pháp thay thế để tải các cài đặt mới nhất của Android Studio và Java lên máy tính của bạn.

Những gì bạn sẽ học

* Cách cài đặt và sử dụng IDE Android Studio.
* Cách sử dụng quy trình phát triển để xây dựng ứng dụng Android.
* Cách tạo một dự án Android từ một mẫu.
* Cách thêm thông điệp ghi lại vào ứng dụng của bạn để phục vụ mục đích gỡ lỗi.

Những gì bạn sẽ làm

* Cài đặt môi trường phát triển **Android Studio**.
* Tạo một trình giả lập (thiết bị ảo) để chạy ứng dụng của bạn trên máy tính.
* Tạo và chạy ứng dụng **Hello World** trên các thiết bị ảo và vật lý.
* Khám phá cấu trúc dự án.
* Tạo và xem các thông điệp ghi lại từ ứng dụng của bạn.
* Khám phá tệp **AndroidManifest.xml**

### Giao diện người dùng tương tác đầu tiên

### Trình chỉnh sửa bố cục

### Văn bản và các chế độ cuộn

### Tài nguyên có sẵn

## Activities

### Activity và Intent

### Vòng đời của Activity và trạng thái

### Intent ngầm định

## Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ

### Trình gỡ lỗi

### Kiểm thử đơn vị

### Thư viện hỗ trợ

# TRẢI NGHIỆM NGƯỜI DÙNG

## Tương tác người dùng

### Hình ảnh có thể chọn

### Các điều khiển nhập liệu

### Menu và bộ chọn

### Điều hướng người dùng

### RecycleView

## Trải nghiệm người dùng thú vị

### Hình vẽ, định kiểu và chủ đề

### Thẻ và màu sắc

### Bố cục thích ứng

## Kiểm thử giao diện người dùng

### Espresso cho việc kiểm tra UI

# LÀM VIỆC TRONG NỀN

## Các tác vụ nền

### AsyncTask

### AsyncTask và AsyncTaskLoader

### Broadcast receivers

## Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền

### Thông báo

### Trình quản lý cảnh báo

### JobScheduler

# LƯU DỮ LIỆU NGƯỜI DÙNG

## Tùy chọn và cài đặt

### Shared preferences

**Giới thiệu**

Shared preferences cho phép bạn lưu trữ 1 lượng nhỏ primitive data theo kiểu cặp key/value trong tệp trên thiết bị. Để lấy 1 tệp preference và để đọc, viết và quản lý preference data, sử dụng lớp SharedPreferences. Android framework quản lý tệp sharedPreferences của nó. Tệp này cho phép truy cập bởi mọi thành phần trong ứng dụng của bạn, nhưng mà nó không cho ứng dụng khác truy cập vào.

Dữ liệu của bạn lưu ở Shared preferences khác với dữ liệu trong saved activity state, chúng tôi đã đề cập đến saved activity state ở chương trước.

- Dữ liệu trong saved activity instance state được dữ lại trong cùng phiên làm việc của người dùng.

- Shared preferences tồn tại trong suốt phiên làm việc của người dùng. Shared preferences được dữ lại ngay cả khi nếu ứng dụng của bạn dừng và khởi động lại hoặc nếu thiết bị khởi động lại

Sử dụng Shared preferences chỉ khi bạn cần lưu 1 lượng nhỏ dữ liệu như là cặp key/value đơn giản. Để quản lý lượng lớn dữ liệu lưu trữ lâu dài trong ứng dụng, sử dụng phương pháp lưu trữ khác như là Room library hoặc SQL Database.

**CÓ THỂ BẠN ĐÃ BIẾT**

Bạn nên thành thạo về:

* Tạo, xây và chạy ứng dụng trên android studio
* Thiết kế giao diện với buttons và text views
* Lưu và khôi phục activity instance state.

Bạn sẽ học về?

Bạn sẽ được học cách:

Xác định Shared preferences là gì

Tạo tệp Shared preferences cho ứng dụng

Lưu dữ liệu vào Shared preferences và đọc lại dữ liệu đó

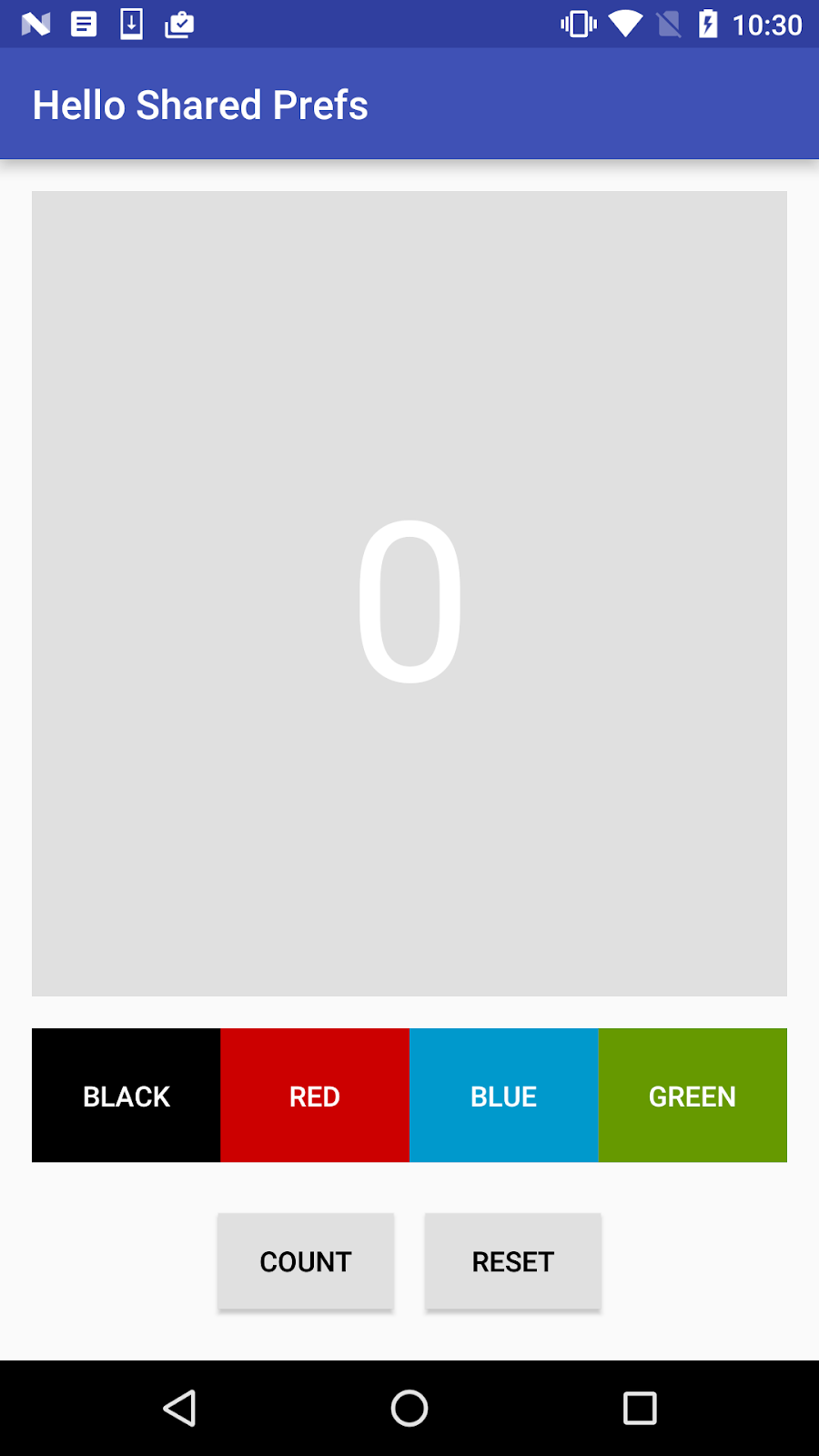
Xóa dữ liệu trong Shared preferences.

Bạn sẽ làm gì?

Nâng cấp ứng dụng để nó có thể lưu, truy xuất hoặc đặt lại Shared preferences.

Tổng quan về ứng dụng

Ứng dụng HelloSharedPrefs là 1 biến thể khác của ứng dụng HelloToast bạn đã tạo ở bài 1. nó có những buttons để tăng số count, thay đổi màu nền và khởi tạo lại count và màu về mặt định. Ứng dụng cũng sử dụng themes và style để định nghĩa buttons.



Bắt đầu với ứng dụng khở tạo và thêm shared preferences vào main activity code. Bạn cũng thêm cả button “Reset” để thiết lập count và màu nền về mặc định và xóa hết tệp preferences.

**Nhiệm vụ 1: khám phá HelloSharedPrefs**

Project ứng dụng starter cho bài thực hành này có sẵn tại HelloSharedPrefs-Srarter. Trong nhiệm vụ này, bạn sẽ tải project into Android Studio và khám phá và đặc điểm của key trong ứng dụng.

* 1. Mở và chạy project HelloSharedPrefs-Srarter

1. Tải ứng dụng HelloSharedPrefs-Srarter và giải nén tệp.
2. Mở project trên Android Studio sau đó xây và chạy thử ứng dụng. Thử những điều sau:

* Bấm vào nút “count” thì số trong text view chính sẽ tăng lên.
* Bấm bất kì nút màu để thay đổi màu nền và text view chính cũng sẽ thay đổi
* Xoay thiết bị và ghi chú rằng màu nền và count vẫn được giữ nguyên.
* Bấm button “Reset” để cài lại màu và count về mặc định.
* Buộc dừng ứng dụng sử dựng 1 trong những phương pháp sau:
* Trong android, chọn Run > Stop ‘app’ hoặc ấn vào biểu tượng Stop  ở thanh công cụ.
* Trên thiết bị, ấn vào nút quay lại (nút hình vuông ở góc dưới bên phải, vuốt thẻ ứng dụng HelloSharedPrefs để thoát hoặc là ấn vào X trên góc phải của thẻ. Nếu bạn thoát ứng dụng theo cách này, hãy đợi vài giây để hệ thống dọn dẹp trước khi khởi động lại

1. Chạy lại ứng dụng:

Khi bạn khởi động lại ứng dụng, giao diện mặc điịnh sẽ hiển thị - count là 0 và màu nền là màu xám.

* 1. Khám phá Activity code

1. Mở MainActivity
2. Xem xét mã nguồn và lưu ý những điểm sau:

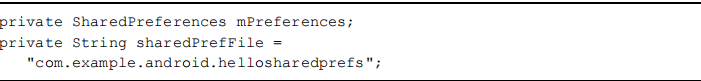
* Biến đếm (mCount) đã được định nghĩa ở kiểu integer. Phương thức onClick countUp() được gọi khi nhấn vào nút Count, giúp tăng giá trị này và cập nhật TextView chính.
* Màu nền (mColor) cũng là kiểu integer, ban đầu được đặt là màu xám từ tệp colors.xml với tên default\_background.
* Phương thức changeBackground() được gọi khi nhấn vào một trong các màu sau đó sẽ đặt text view chính theo màu đó.
* Phương thức onSaveinstanceState() và khôi phục trong phương thức onCreate(). Các khóa(key) bundle cho count và màu được định nghĩa bởi các biến private COUNT\_KEY và COLOR\_KEY.

**Nhiệm vụ 2: Lưu và khôi phục lại dữ liệu của tệp shared preferences**

Trong nhiệm vụ này, bạn sẽ lưu lại trạng thái của ứng dụng ở tệp shared preferences và đọc dữ liệu lại mỗi khi ứng dụng khởi động lại. Bởi vì trạng thái dữ liệu mà bạn lưu ở shared preferences ( nơi lưu count và color gần đây) trùng với dữ liệu mà bạn lưu trữ trong intance state nên không cần phải lưu 2 lần mà thay thế hoàn toàn intance state bằng shared preferences

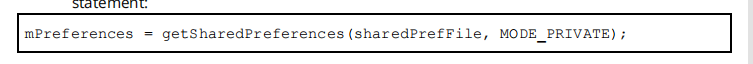
* 1. Bắt đầu shared preferences:

1. Thêm biến vào lớp MainActivity để lưu tên của tệp shared preferences và tham chiếu đến đối tượng shared preferences.



Bạn có thể đặt tên tệp shared preferences của bạn tùy thích nhưng theo quy ước thì phải trùng tên với tên package của ứng dụng.

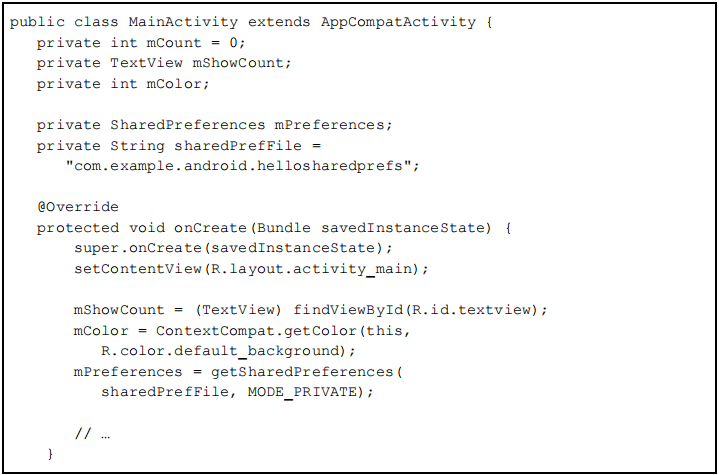
1. Trong phương thức onCreate(), khởi tạo đối tượng shared preferences bằng cách thêm dòng mã sau trước câu lệnh if:

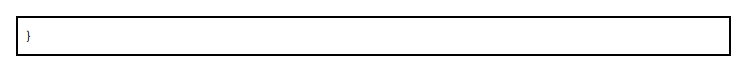


Phương thức getSharedPreferences ( từ Context của activity) mở tệp tại tên tệp được chỉ định (sharePreFile) với chế độ MODE\_PRIVATE.

**LƯU Ý:** các phiên bản android cũ hơn đã có các chế độ khác mà cho phép bạn tạo tệp preferences có thể đọc.ghi bởi các ứng dụng khác. Các chế độ này đã bị loại bỏ từ API 17 và hiện nay không được khuyến nghị vì lý do bảo mật. Nếu bạn cần chia sẻ dữ liệu với các ứng dụng khác, hãy cân nhắc sử dụng content URI do FileProvider cung cấp.

Đoạn mã giải quyết vấn đề cho MainActivit, chỉ có 1 phần được hiển thị thôi:





* 1. Lưu preferences trong onPause()

Lưu preferences khá giống với lưu instance state - cả 2 thao tác đều đặt dữ liệu dưới dạng cặp key/value vào 1 đối tượng Bundle. Tuy nhiên, bạn lưu dữ liệu đó trong callback vòng đời onPause() và cần 1 đối tượng editor (SharedPreferences.Editor) để ghi vào đối tượng Shared Preferences.

1. Thêm phương thức vòng đời onPause vào MainActivity



1. Trong onPause(), lấy 1 đối tượng editor cho đối tượng SharedPreferences:



Một editor của preferences được chấp thuận để viết vào đối tượng shared preferences. Thêm dòng này vào onPause() sau khi gọi đến super.onPause().

1. Sử dụng phương thức putInt() để đẩy kiểu integer của mCount và mColor đến shared preferences với các khóa tương ứng:



Lớp SharedPreferences.Editors bao gồm đa phương thức “đẩy” cho những kiểu dữ liệu khác nhau bao gồm puInt() và puString().

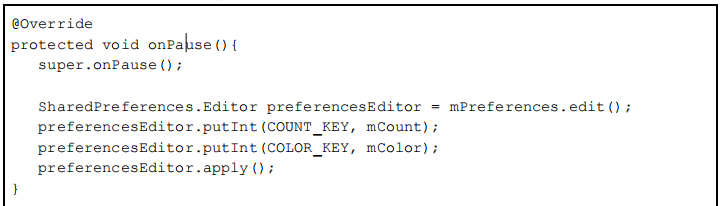
1. Gọi apply() để lưu preferences:



Phương thức apply() lưu shared preferences 1 cách bất đồng bộ, ngoài luồng giao diện người dùng(UI thread). Đối tượng Shared preferences, editor cũng có phương thức commit() để lưu preferences 1 cách đồng bộ. Phương thức commit() không được khuyến khích vì nó có thể chặn các thao tác khác.

1. Xóa toàn bộ phương thức onSaveInstanceState(). Bởi vì ví activity instance state bao gồm dữ liệu giống như shared preferences, bạn có thể thay thế hoàn toàn instance state bằng shared preferences.

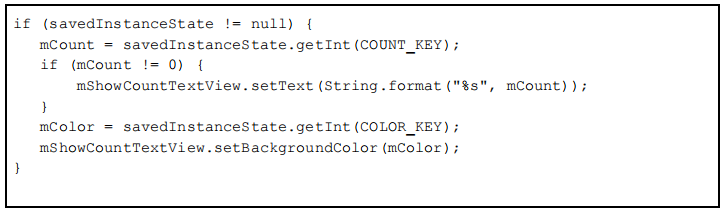
Code giải pháp cho phướng thức onPause(): trong MainActivity



* 1. Phục hồi preferences trong onCreate()

Cũng giống như instance state, phần mềm của bạn đọc tất cả shared preferences trong phương thức onCreate(). 1 lần nữa, bởi vì shared preferences bao gồm dữ liệu giống như intance state, chúng ta có thể thay thế intance state bằng shared preferences ở đây. Mỗi lần onCreate() được gọi - khi ứng dụng khổi động hoặc khi có thay đổi cấu hình - shared preferences sẽ được sử dụng để khôi phục trạng thái của giao diện.

1. Xác định phần trong phương thức onCreate() mà thử nếu đối số savedInstanceState là null và phục hồi instance state:



1. Xóa toàn bộ khối đó
2. Trong phương thức onCreate(), tại vị trí tương tự như phần code instance state, lấy biến đếm từ trong preferences với COUNT\_KEY và gán cho nó biến mCount.



Khi bạn đọc dữ liệu từ preferences bạn không cần lấy shared preferences editor. Sử dụng bất cứ phương thức get nào trong đối tượng shared preferences (như là getInt() hoặc getString() để truy xuất dữ liệu preference.

Lưu ý rằng phương thức getInt() lấy 2 đối số: 1 cho key và 1 cho giá trị mặc định nếu key không tìm thấy. Trong trường hợp này giá trị mặc định là 0, giống với khở tạo của mCount.

1. Cập nhật giá trị của TextView chính với count mới:



1. Lấy màu từ preferences bằng key COLOR\_KEY và gán cho nó biến mColor



Trước đó, đối số getInt() đặt là giá trị mặc định đẻ sử dụng trong trường hợp khóa không tồn tại trong shared preferences. Trong trường hợp này bạn có thể chỉ sử dụng lại giá trị mColor ( đã được khởi tạo với nền mặc định trong phương thức).

1. Cập nhật màu nền của main text view



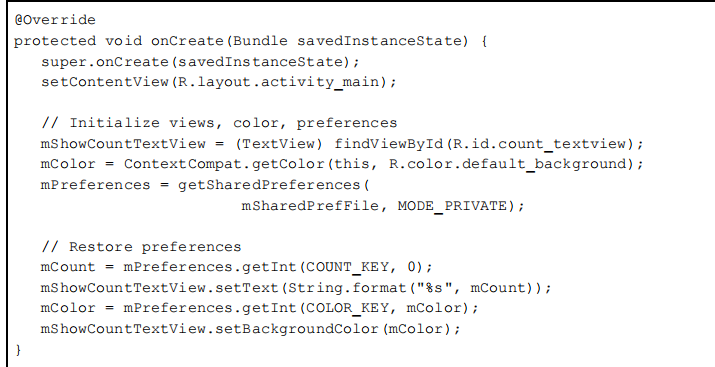
1. Chạy thử ứng dụng, nhấp vào nút Count và thay dổi màu nền để cập nhật instance state và preferences.
2. Quay thiết bị hoặc máy ảo để phản hồi số đếm và màu đã được lưu lại khi có thay đổi cấu hình.
3. Buộc dừng ứng dụng bằng một trong các phương thức:

- Trong Android Studio, chọn Run > Stop ‘app’

- Trên thiết bị, nhấn nút Recents ( nút hình vuông ở phía dưới góc phải). Vuốt thẻ ứng dụng HelloSharedPrefs để thoát ứng dụng hoặc nhấp vào X trên góc phải của thẻ.

1. Chạy lại ứng dụng. Ứng dụng khởi động lại và tải preferences, giữ nguyên các trạng thái.

Mã nguồn giải pháp cho phương thức onCreate() của MainActivity:



* 1. Đật lại preferences trong trình xử lý sự kiện reset()

Nút “đặt lại” trong ứng dụng ban đầu sẽ cài lại cả số đếm và màu cho activity về giá trị mặc định. Bởi vì preferences giữ trạng thái của activity, nó rất quan trọng khi xóa cả preferences cùng lúc.

1. Trong phước thức reset() của sự kiện onClick, sau khi màu và số đếm đã được đặt lại, lấy editor của đối tượng SharedPreferences:



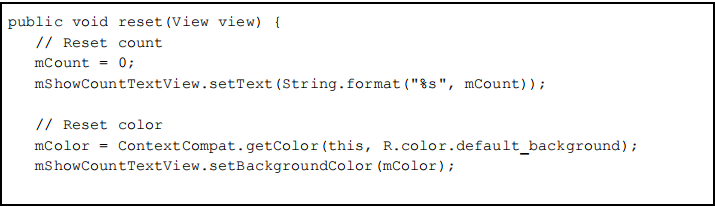
1. Xóa tất cả shared preferences

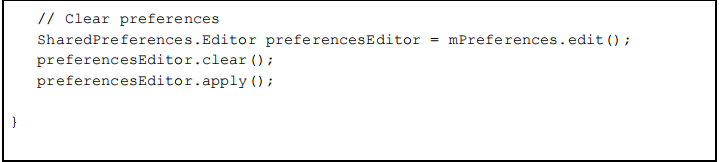


1. Áp dụng sự thay đổi:



Đoạn mã giải quyết cho phương thức reset():





Mã giải quyết

Android Studio project: HelloSharedPrefs

**Thách thức:**

Lưu ý: chỉnh sửa ứng dụng HelloSharedPrefs để thay vì tự động hóa lưu trạng thái của tệp preferences, thêm activity thứ 2 để thay đổi, cài lại những preferences. Thêm nút có tên Settings để khởi chạy activity đó, bao gồm nút toggle và spinner để thay đổi preferences

**Tổng kết**

* lớp SharedPreferences chấp nhận ứng dụng lưu trữ lượng nhỏ dữ liệu primitive như là cặp key/value.
* Shared preferences được lưu trữ qua nhiều phiên làm việc của người dùng trong cùng 1 ứng dụng.
* Để ghi dữ liệu vào shared preferences, lấy đối tượng SharedPreferences.Editor
* Sử dụng các phương thức “put” trong đối tượng SharedPreferences.Editor, như là putInt() hoặc putSring() để đẩy dữ liệu lên shared preferences với key và value.
* Sử dụng các phương thức “get” trong đối tượng SharedPreferences, như là getInt() hoặc getString() để lấy dữ liệu từ SharedPreferences bằng key.
* Sử dụng phương thức clear() trong đối tượng SharedPreferences.Editor để loại bỏ tất cả dữ liệu lưu trong preferences.
* Sử dụng phương thức apply() trong đối tượng SharedPreferences.Editor để lưu sự thay đổi trong tệp preferences.

**Khái niệm liên quan**

Tài liệu lien quan nằm trong 9.0 Data storage và 9.1 Shared preferences.

**Tìm hiểu thêm**

Tài liệu dành cho nhà phát triển android:

* Data and file storage overview
* Save key-value data
* SharedPreferences
* SharedPreferences.Editor

Stack Overflow:

* How to use SharePreferences in Android to store, fetch and edit values
* onSavedInstanceState vs. SharedPreferences

**BÀI TẬP VỀ NHÀ**

Xây dựng và chạy 1 ứng dụng

Mở ứng dụng ScoreKeeper mà bạn đã tạo trong Android fundamentals 5.1: Drawables, style and themes lesson

1. Thay intance state đã lưu bằng shared preferences với mỗi điểm.
2. Chạy thử ứng dụng, quay thiết bị để đảm bảo cấu hình thay đổi đọc và lưu preferences và cập nhật UI
3. Dừng ứng dụng và khởi tạo lại nó để đảm bảo rằng preferences đã được lưu
4. Thêm nút Reset ( nút cài lại tất giá trị điểm thành 0 và xóa shared preferences)

**Trả lời những câu hỏi sau**

Câu hỏi 1:

Bạn lưu trạng thái của ứng dụng vào shared preferences trong phương thức vòng đời nào?

Câu hỏi 2:

Bạn khôi phục trạng thái ứng dụng bằng phương thức vòng đời nào?

Câu hỏi 3:

Bạn có nghĩ ra trường hợp nếu nó tạo cả preferences và intance state là hợp lý không?

**Nộp ứng dụng của bạn cho người chấm điểm**

Hướng dẫn cho người chấm điểm

* Ứng dụng giữ nguyên điểm số khi quay thiết bị
* Ứng dụng giữ nguyên điểm số sau khi ứng dụng bị dừng hoặc khởi động lại
* Ứng dụng lưu điểm của shared preferences trong phương thức onPause().
* Ứng dụng khôi phục shared preferencse bằng phương thức onCreate()
* Ứng dụng hiển thị nút Reset để cài lại điểm = 0

Đảm bảo rằng việc triển khai phương thức xử lý sự kiện nhấn(on-click handler) thỏa mãn những điều sau:

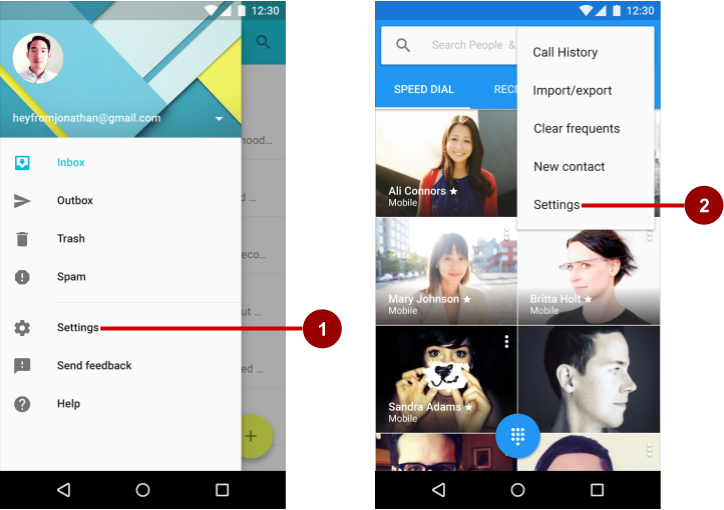
* Đặt lại cả 2 biến số về 0
* Cập nhật cả 2 text views
* Xóa shared preferences

**BÀI 9.2 App settings**

**Giới thiệu**

Ứng dụng thường bao gồm các cài đặt để cho phép người dùng sửa các tính năng và hành vi của ứng dụng. Ví dụ như vài ứng dụng cho phép người dùng cài địa điểm nhà, đơn vị mặc định, thước đo và các cài đặt khác áp dụng cho toàn bộ ứng dụng. Người dùng không truy cập cài đặt thường xuyên, bởi vì 1 khi người dùng thay đổi cài đặt, như là địa điểm nhà, họ hiếm khi cần thay đổi nó 1 lần nữa.

Người dùng mong đợi có thể truy cập cài đặt ứng dụng bằng cách nhấn vào Settings trong menu điều hướng, như là thanh điều hướng trượt (navigation drawer) như trong hình bên trái dưới đây, hoặc menu lựa chọn ở thanh ứng dụng, hiện lên ở phía phải màn hình dưới đây.



Trong bức ảnh phía trên

1. Settings trong menu điều hướng (navigation drawer)
2. Settings trong menu tùy chọn trên thanh ứng dụng

Trong bài thực hành này bạn sẽ thêm settings activity vào ứng dụng. Người dùng có thể truy cập cài đặt bằng cách bấm Settings (Cái mà được đặt ở menu tùy chọn trong thanh ứng dụng (app bar).

**Kiến thức yêu cầu:**

* Tạo project Android Studio từ template và giao diện chính
* Chạy ứng dụng trên máy ảo hoặc máy thật được kết nối
* Tạo và sửa các phần tử UI bằng layout editor hoặc XML.
* Trích xuất tài nguyên chuỗi và sửa giá trị chuỗi
* Truy cập phần tử UI từ code của bạn bằng cách sử dụng findViewById()
* Xử lý sự kiện nhấn nút
* Hiển thị thông báo Toast
* Thêm 1 Activity vào ứng dụng
* Thêm menu options vào thanh ứng dụng (app bar)
* Thêm và chỉnh sửa các mục trong menu options
* Xử dụng styles và themes trong project
* Sử dụng SharedPreferences

**Bạn sẽ được học**

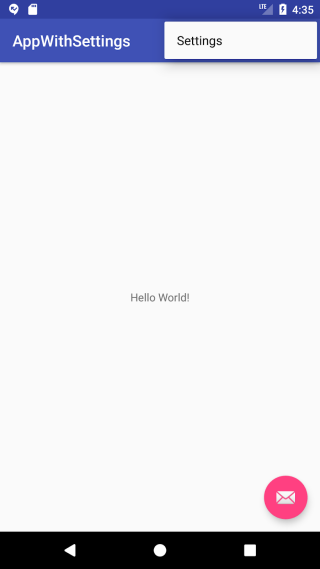
* Thêm Fragment cho quản lý cài đặt
* Tạo 1 tệp tài nguyên XML chứa các cài đặt cùng các thuộc tính của chúng
* Tạo điều hướng đến settings Activity
* Đặt giá trị mặc định cho settings
* Đọc giá trị settings thay đổi bởi người dùng
* Tùy chỉnh mẫu Settings Activity

**Bạn sẽ làm gì?**

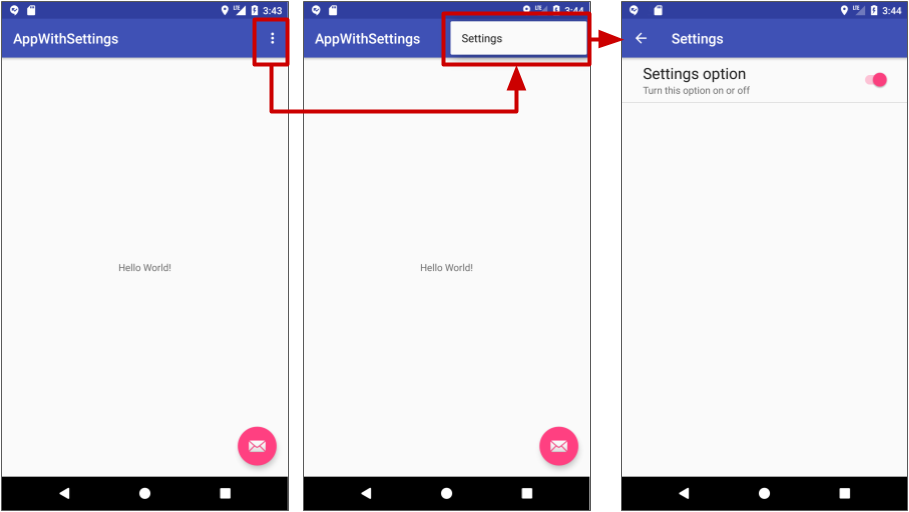
* Tạo 1 ứng dụng bao gồm Settings ở menu options
* Thêm công tắc chuyển đổi (Toggle switch) cho Settings option
* Thêm code để đặt giá trị mặc định cho cài đặt và truy cập giá trị setting sau khi nó đã được đổi.
* Sử dụng và tùy chỉnh mẫu Android Studio Settings Activity

**Tổng quan ứng dụng**

Android studio cung cấp 1 cách nhanh chóng cho thiết lập menu tùy chọn có chứa Settings. Nếu bạn bắt đầu 1 Android studio project cho điện thoại hoặc máy tính bảng sử dụng Basic Activity Template, Ứng dụng mới bao gồm Settings được trình bày ở phía dưới.



Mẫu này cũng bao gồm Floating Action Button ở góc dưới bên phải của màn hình với biểu tượng phong bì. Bạn có thể bỏ qua nút này trong phần thực hành này nếu bạn không muốn dùng tới nó. Bạn sẽ bắt đầu bằng cách tạo 1 ứng dụng có ten là AppWithSettings sử dụng Basic Activity template và bạn sẽ thêm settings Activity để cung cấp công tắc bật tắc(toggle switch) để người dùng có thể bật tắt nó.

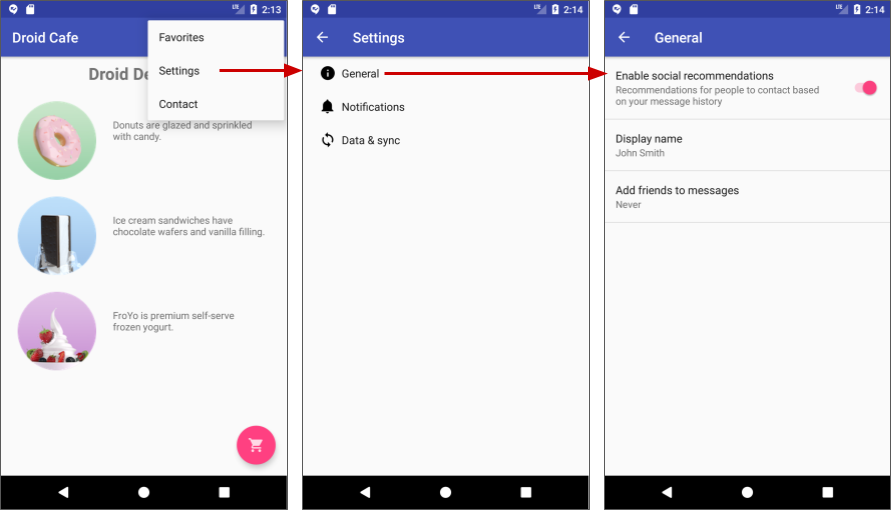


Bạn sẽ thêm code để đọc settings và thực hiện 1 hành động dựa trên giá trị của nó. Đơn giản như là hành động hiển thị thông báo Toast với giá trị của cài đặt.

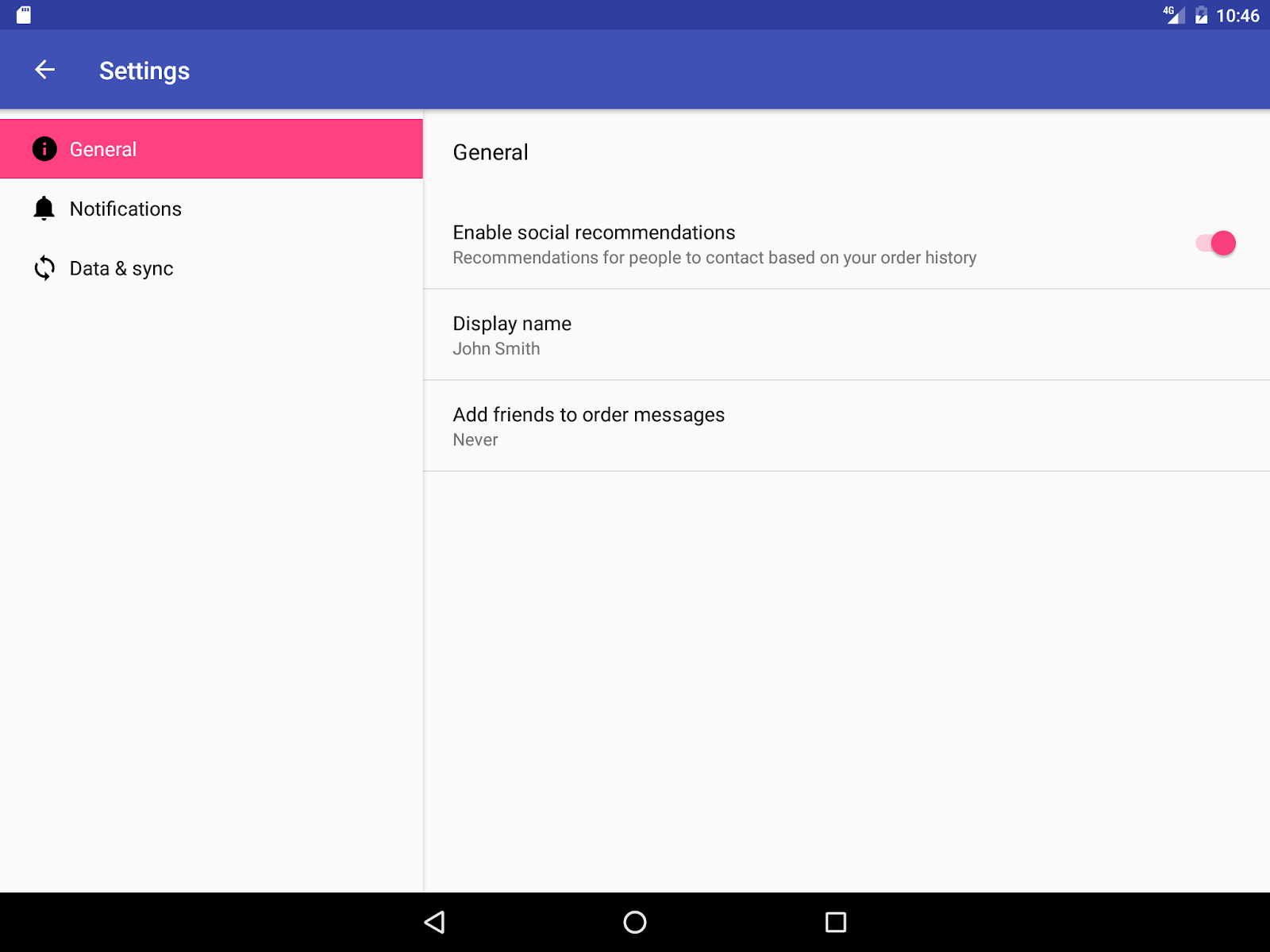
Trong nhiệm vụ thứ 2, bạn sẽ thêm mẫu Settings Activity được cung cấp bởi Android Studio cho ứng dụng DroidCafeOptionUp mà bạn đã tạo ở bài trước.

Mẫu Settings Activity được điền sẵn các cài đặt mà bạn có thể tùy chỉnh cho ứng dụng và cung cấp các bố cục khác nhau cho điện thoại và cho máy tính bảng:

* Điện thoại: Màn hình Settings chính với 1 liên kết tiêu đề cho mỗi nhóm cài đặt, chẳng hạn như nhóm General cho cài đặt general, như được trình bày phía dưới



* Máy tính bảng: Giao diện màn hình bố chí theo kiểu master/detail với liên kết tiêu đề cho mỗi nhóm ở phía trái (master) và nhóm cài đặt ở bên phía phải (detail) được trình bày phía dưới



Để tùy chỉnh template này, bạn sẽ thay đổi headers, setting titles, setting descriptions và giá trị cho settings

Ứng dụng DroidCafeOptionUp đã được tạo trong bài trước từ Basic Activity template (cái mà cung cấp các tùy chọn menu trong ứng dụng trong app bar cho tùy chỉnh Settings option, bạn sẽ tùy chỉnh lại supplied Settings Activity template bằng cách thay đổi 1 cài đặt của title, description, giá trị và giá trị mặc định. Bạn sẽ thêm code để đọc giá trị của settings sau khi người dùng thay đổi nó và hiển thị giá trị đó.

Nhiệm vụ 1: thêm bộ chuyển đổi cho ứng dụng

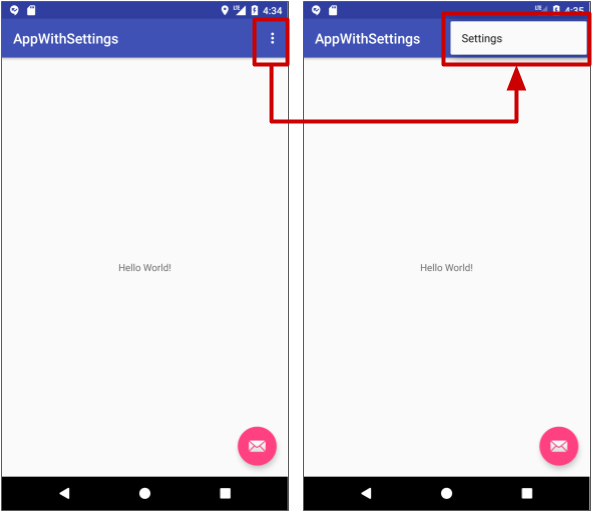
Trong nhiệm vụ này, bạn sẽ được học những điều sau:

* Tạo mới khung project bằng Basic Activity template ( thứ mà cung cấp option menu)
* Tạo nút công tắc (SwitchPreference) cùng các thuộc tính trong tệp preference XML
* Thêm activity cho settings và fragment cho cụ thể 1 cài đặt. Để đảm bảo khả năng tương thích với AppCompatActivity, bạn sử dụng PreferenceFragmentCompat nhiều hơn PreferenceFragment. Bạn cũng thêm thư viện android.support.v7.preference
* Kết nối mục Settings trong menu options với settings activity.
  1. Tạo mới project và thêm thư mục XML cùng tệp tài nguyên

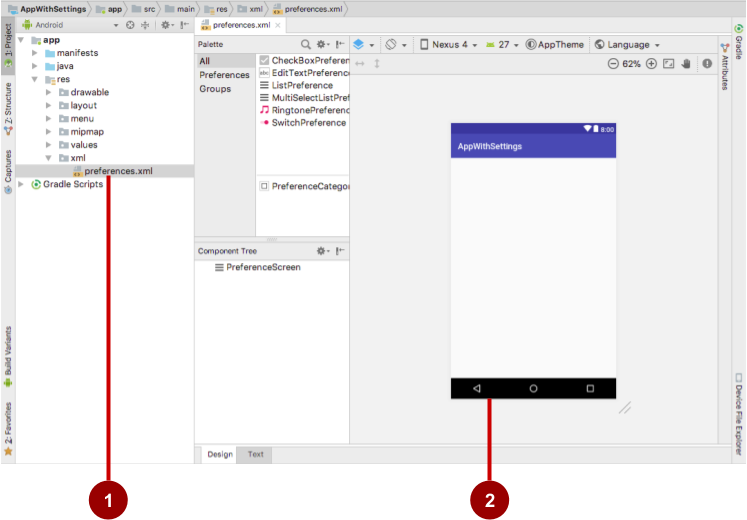
1. Trong Android Studio, tạo mới 1 project với những tham số sau:

|  |  |
| --- | --- |
| Attribute | Value |
| Application Name | AppWithSettings |
| Company Name | Android.example.com (hoặc domain của bạn) |
| Project location | Đường dẫn đến thư mục các dự án của bạn |
| Phone and Tablet Minimum SDK | API15: Android 4.0.3 IceCreamSandwich |
| Template | Basic Activity |
| Activity Name | MainActivity |
| Layout Name | Activity\_main |
| Title | MainActivity |

1. Chạy ứng dụng và nhấn vào biểu tượng overflow trên app bar để xem menu tùy chọn, như hình bên dưới. Mục duy nhấ trong menu tùy chọn là Settings.

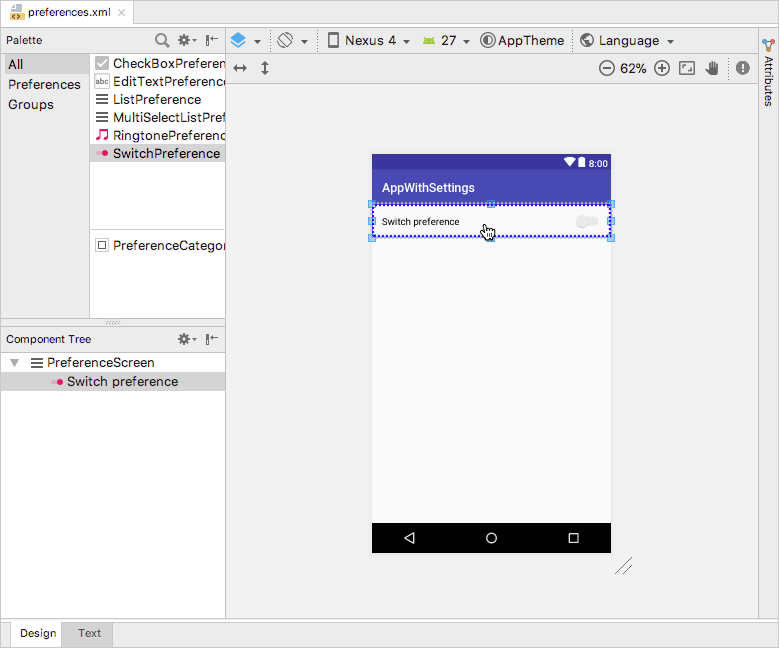


1. Bạn cần tạo mới thư mục tài nguyên để giữ tệp XML bao gồm settings. Chọn thư mục res trong khung Project > Android và chọn File > New > Android Resource Directory. Hộp thoại New Resource Directory sẽ xuất hiện.
2. Trong Resource chọn drop-down menu, chọn xml. Directory name tự động chuyển sang xml. Bấm OK.
3. Thư mục xml xuất hiện trong bảng Project > Android trong thư mục res. Chọn xml và chọn File > New > XML resource file ( hoặc chuột phải vào xml và chọn New > XML rescource file)
4. Nhập tên của tệp XML là preferences, trong trường File name, và chọn OK. Tệp preferences.xml sẽ xuất hiện trong thư mục xml và layout editor sẽ xuất hiện, như hình bên dưới.



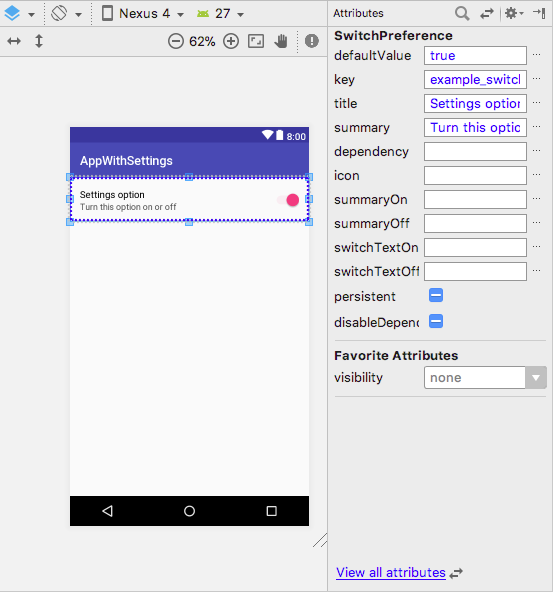
Trong ảnh trên:

1. Tệp preferences.xml trong thư mục (directory) xml.
2. Layout editor sẽ xuất hiện trong nội dung của preferences.xml
   1. Thêm XML preferences và thuộc tính cho cài đặt
3. Kéo 1 SwitchPreference từ bảng Palette ở bên vào phía trên cùng của layout, như hình minh họa dưới đây.

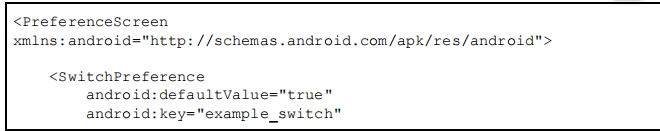


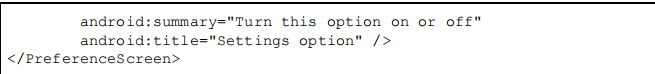
1. Thay đổi giá trị trong bảng Attributes phía phải của layout editor như trong hình phía dưới:

* DefaultValue: true
* Key: example\_switch
* Title: Settings option
* Summary: Turn this option on or off



1. Nhập vào tab Text ở dưới cùng của layout editor để xem XML code.





1. Trích xuất tài nguyên chuỗi cho các giá trị thuộc tính android:title và android:summary thành @string/switch\_title và @string/switch\_summary.

Thuộc tính XML cho các preference là:

* android:defaultValue: giá trị mặc định cho cài đặt khi ứng dụng khởi động lần đầu tiên.
* Android:title: tiêu đề của cài đặt cho SwitchPreference, tiêu đề xuất hiển ở phía trái của nút công tắc(toggle switch).
* Android:key: Khóa sử dụng để lưu trữ giá trị cài đặt. Mỗi cài đặt có một cặp key-value tương thích mà hệ thống sử dụng để lưu cài đặt trong tệp SharedPreferences mặc định cho cài đặt mà hệ thống sử dụng để lưu cài đặt ở tệp mặc định SharedPreferences cho ứng dụng cài đặt.
* Android:summary: Văn bản tóm tắt hiển thị bên dưới cài đặt
  1. Sử dụng SwitchPreferenceCompat

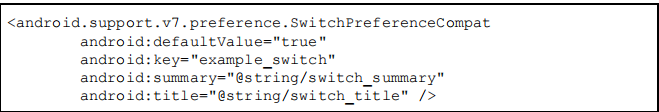
Để sử dụng phiên bản PreferenceFragmentCompat của PreferenceFragment, bạn bắt buộc phải sử dụng phiên bản android.support.v7 của SwitchPreference(SwitchPreferenceCompat)

1. Trong bảng Project > Android, mở tệp build.gradle (Module:app) trong thư mục Gradle Scripts và thêm đoạn mã sau vào phần dependencies:



Câu lệnh được hiển thị ở bên trên thêm thư viện android.support.v7.preference để sử dụng phiên bản PreferenceFragmentCompat của PreferenceFragment.

1. Trong tệp preference.xml trong thư mục xml, thay đổi <SwitchPreference trong code thành <android.support.v7.preference.SwitchPreferenceCompat:



Dòng SwitchPreferenceCompat ở trên có thể hiện thị biểu tượng bóng đèn cảnh báo màu vàng nhưng bạn có thể bỏ qua nó.

1. Mở tệp style.xml trong tệp values và thêm mục preferenceTheme vào trong khai báo của Apptheme:



Để sử dụng PreferenceFragmentCompat, bạn bắt buộc phải khai báo preferenceTheme với kiểu PreferenceThemeOverlay trong app theme.

* 1. **Thêm Activity cho settings**

Để tạo settings Activity( thứ cung cấp cài đặt UI) thêm Empty Activity vào ứng dụng, làm theo các bước sau:

1. Chọn app ở trên cùng của bảng Project>Android và chọn New>Activity>Empty Activity
2. Đặt tên Activity là SettingsActivity. Đặt tùy chọn Generate Layout File là unchecked (Bạn sẽ không cần đến nó) và để tùy chọn Launcher Activity là unchecked.
3. Đặt lựa chọn Backwards Compatibility (AppCompat) là checked. Tên của Package nên đặt là com.example.android.projectname.
4. Bấm **Finish**
   1. **Thêm Fragment cho cài đặt cụ thể**

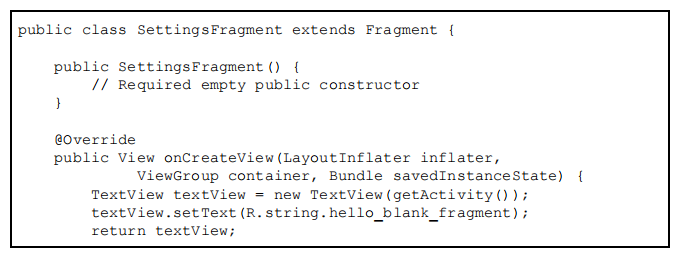
Fragment giống như 1 phần mô-đun của Activity, nó có vòng đời riêng và nhận các sự kiện đầu vào riêng. Bạn có thể thêm hoặc xóa 1 Fragment trong khi Activity đang chạy. Bạn sẽ sử dụng 1 lớp con chuyên biệt của Fragment để hiển thị danh sách cài đặt. Chỗ hay nhất của phần thực hành này là sử dụng Activity thường xuyên để lưu trữ và chạy (host) PreferenceFragment để hiển thị cài đặt ứng dụng. PreferenceFragment cung cấp rất nhiều kiến trúc linh hoạt hơn là sử dụng Activity cho preferences.

Bạn sẽ sử dụng PreferenceFragmentCompat nhiều hơn là PreferenceFragment để duy trì tính tương thích với AppCompatActivity.

Trong bước này bạn sẽ thêm 1 Fragment blank cho 1 nhóm các cài đặt tương tự( không có layout, phương thức factory hay interface callbacks) cho ứng dụng và mở rộng PreferenceFragmentCompat

Làm theo các bước sau:

1. Chọn app 1 lần nữa, và chọn New > Fragment > Fragment (Blank)
2. Đặt tên Settings của fragment. Bỏ chọn lựa chọn Create layout XML ( vì bạn k cần đến nó)
3. Bỏ chọn lựa chọn bao gồm phương thức fragment factory và interface callbacks
4. Target Source Set nên đặt là main
5. Bấm Finish. Kết quả của lớp định nghĩa trong SettingsFragment như sau:

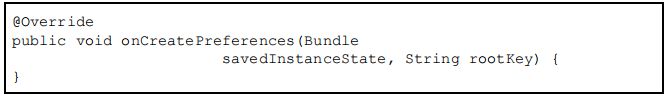




1. Chỉnh sửa lớp định nghĩa của SettingsFragment để mở rộng PreferenceFragmentCompat:



Khi bạn đã thay dổi lớp định nghĩa nên nó sẽ nối với định nghĩa được trình bày phía trên, biểu tượng bóng đèn đỏ xuất hiện bên lề trái. Nhấp vào bóng đèn đỏ và chọn **Implement methods** và sau đó chọn onCreatePreferences. Android Studio sẽ tạo ra 1 mẫu phương thức (stub) cho phương thức onCreatePreferences() như sau:



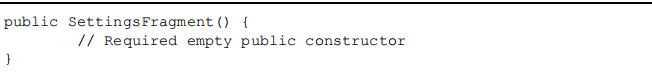
Để mở rộng Fragment, Android Studio thêm các câu lệnh import sau:



1. Xóa tất cả phương thức onCreateView() trong fragment.

Lý do mà bạn quan tâm đến thay thế onCreateView() với onCreatePreferences() bởi vì bạn sẽ thêm SettingsFragment vào SettingsActivity hiện có thể hiển thị các tùy chọn(preferences), nhiều hơn là hiện thị màn hình Fragment riêng biệt. Thêm nó vào Activity đã có sẵn và lmaf nó dễ dàng để thêm hoặc xóa Fragment trong khi Activity đang chạy. Fragment hiện thị các tùy chọn sẽ được gắn vào PreferenceScreen bằng rootkey.

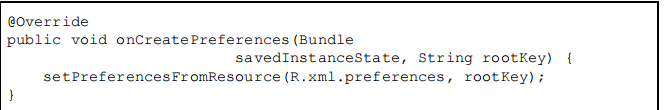
Bạn có thể xóa 1 cách an toàn constructor rỗng khỏi SettingsFragment rất an toàn bởi vì Fragment không hiển thị bản thân nó.



1. Bạn cần liên kết Fragment này với tệp tài nguyên preferences.xml mà bạn đã tạo ở bước trước. Thêm vào phương thức mẫu vừa tạo( phương thức onCreatePreference() ) 1 lệnh gọi setPreferencesFromResource(), truyền vào ID của tệp XML (R.xml.preference) và rootKey để xác định gốc của preference trong PreferenceScreen:



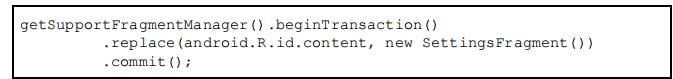
Phương thức onCreatePreference() nên giống như thế này:



* 1. **Hiển thị Fragment trong SettingsActivity**

Để hiển thị Fragment trong SettingsActivity, làm theo những bước sau:

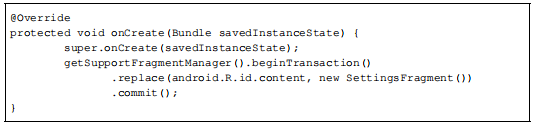
1. Mở SettingsActivity
2. Thêm những code sau vào cuối phương thức onCreate() để Fragment được hiển thị như là nội dung chính.



Đoạn mã dưới đây sử dụng mô hình phổ biến để thêm một Fragment vào một Activity, sao cho Fragment xuất hiện như nội dung chính của Activity:

* Sử dụng **getFragmentManager()** nếu lớp mở rộng từ **Activity** và **Fragment** mở rộng từ **PreferenceFragment**.
* Sử dụng **getSupportFragmentManager()** nếu lớp kế thừa từ **AppCompatActivity** và **Fragment** kế thừa từ **PreferenceFragmentCompat**.

Phương thức **onCreate()** trong **SettingsActivity** sẽ có dạng như sau:



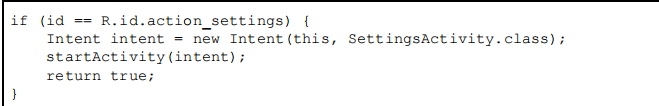
### ****1.7 Kết nối mục Settings trong menu với SettingsActivity****

Sử dụng **Intent** để khởi chạy **SettingsActivity** từ **MainActivity** khi người dùng chọn **Settings** từ menu tùy chọn.

1. Mở **MainActivity** và tìm khối **if** trong phương thức **onOptionsItemSelected()**, nơi xử lý sự kiện khi người dùng nhấn vào **Settings** trong menu tùy chọn:



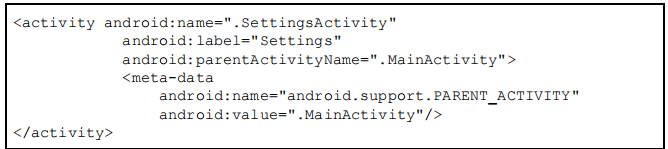
1. Thêm một **Intent** vào khối **if** để khởi chạy **SettingsActivity**.



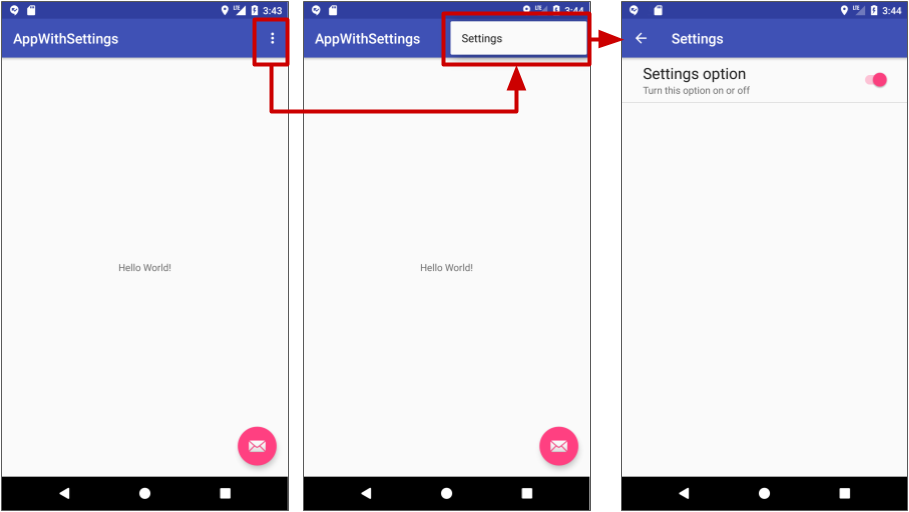
1. Để thêm nút điều hướng **Up** trên thanh **app bar** cho **SettingsActivity**, bạn cần chỉnh sửa khai báo của nó trong tệp **AndroidManifest.xml** để xác định **SettingsActivity** có **MainActivity** làm activity cha.Mở **AndroidManifest.xml** và tìm khai báo **SettingsActivity**:



Thay đổi khai báo thành:



1. Chạy ứng dụng. Nhấn vào biểu tượng menu (overflow icon) để mở menu tùy chọn, như trong hình bên trái. Chọn **Settings** để mở **SettingsActivity**, như trong hình trung tâm. Nhấn nút **Up** trên thanh **app bar** của **Settings Activity**, như trong hình bên phải, để quay lại **Main Activity**



### ****1.8 Lưu giá trị mặc định trong SharedPreferences****

Mặc dù giá trị mặc định cho cài đặt **toggle switch** đã được thiết lập trong thuộc tính **android:defaultValue** (ở [Bước 1.2 của bài này](" \l ")), ứng dụng vẫn cần lưu giá trị mặc định trong tệp **SharedPreferences** cho mỗi cài đặt khi người dùng mở ứng dụng lần đầu tiên. Hãy làm theo các bước sau để thiết lập giá trị mặc định cho **toggle switch**:

1. Mở **MainActivity**.
2. Thêm đoạn mã sau vào cuối phương thức **onCreate()**, ngay sau đoạn mã của **FloatingActionButton**:



**Mã trên đảm bảo rằng các cài đặt được khởi tạo đúng với các giá trị mặc định.**  
Phương thức PreferenceManager.setDefaultValues() nhận ba tham số:

* **Ngữ cảnh của ứng dụng (app context)**, ví dụ như this.
* **ID tài nguyên (preferences)** của tệp XML chứa một hoặc nhiều cài đặt.
* **Một giá trị boolean** xác định xem có nên đặt lại các giá trị mặc định nhiều lần hay không.
  + Khi **false**, hệ thống chỉ đặt giá trị mặc định nếu phương thức này chưa từng được gọi trước đó. Nếu bạn luôn đặt tham số thứ ba là **false**, bạn có thể gọi phương thức này mỗi khi **MainActivity** khởi động mà không làm thay đổi các giá trị cài đặt mà người dùng đã lưu. Tuy nhiên, nếu bạn đặt nó là **true**, phương thức này sẽ ghi đè bất kỳ giá trị nào trước đó bằng giá trị mặc định.

### ****1.9 Đọc giá trị cài đặt đã thay đổi từ SharedPreferences****

Khi ứng dụng khởi động, phương thức **onCreate()** trong **MainActivity** có thể đọc các giá trị cài đặt đã được thay đổi và sử dụng chúng thay vì các giá trị mặc định.

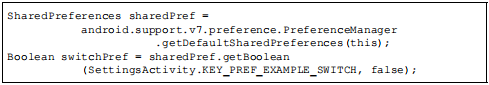
Mỗi cài đặt được xác định bằng một cặp **key-value**. Hệ thống Android sử dụng cặp này để lưu hoặc truy xuất các cài đặt từ tệp **SharedPreferences** của ứng dụng. Khi người dùng thay đổi một cài đặt, hệ thống sẽ cập nhật giá trị tương ứng trong **SharedPreferences**. Để sử dụng giá trị cài đặt, ứng dụng có thể dùng **key** để truy xuất giá trị từ **SharedPreferences** bằng cách sử dụng phương thức **PreferenceManager.getDefaultSharedPreferences()**.

Hãy làm theo các bước sau để thêm đoạn mã này:

1. Mở **SettingsActivity** và tạo một biến **static String** để lưu khóa cho giá trị:

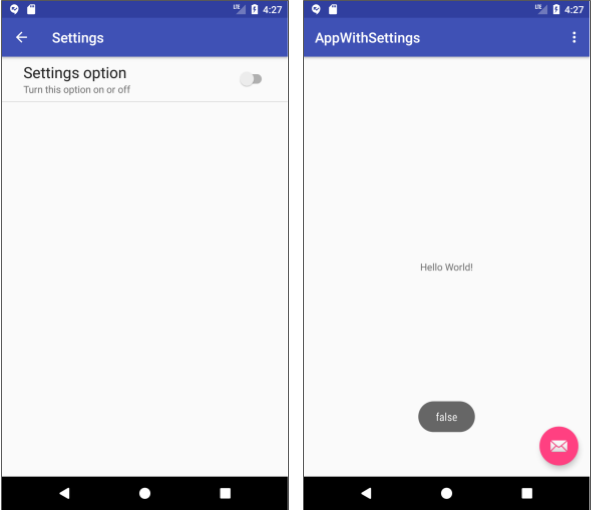


1. Mở **MainActivity** và thêm đoạn mã sau vào cuối phương thức **onCreate()**





1. Chạy ứng dụng và nhấn **Settings** để mở **SettingsActivity**.
2. Nhấn vào tùy chọn cài đặt để thay đổi trạng thái **toggle** từ **on** sang **off**, như minh họa ở phía bên trái của hình dưới đây.
3. Nhấn nút **Up** trong **SettingsActivity** để quay lại **MainActivity**. Một thông báo **Toast** sẽ xuất hiện trong **MainActivity** với giá trị của cài đặt, như minh họa ở phía bên phải của hình dưới đây.
4. Lặp lại các bước trên để kiểm tra xem thông báo **Toast** có thay đổi khi bạn thay đổi cài đặt hay không.



**Đoạn mã trên sử dụng các phương thức sau:**

* android.support.v7.preference.PreferenceManager.getDefaultSharedPreferences(this) để lấy cài đặt dưới dạng đối tượng **SharedPreferences** (được lưu trong biến sharedPref).
* getBoolean() để lấy giá trị **Boolean** của cài đặt, sử dụng khóa (**key**) được định nghĩa là KEY\_PREF\_EXAMPLE\_SWITCH trong **SettingsActivity** và gán giá trị này cho biến switchPref. Nếu không có giá trị nào cho khóa này, phương thức getBoolean() sẽ gán giá trị mặc định **false** cho switchPref. Đối với các kiểu dữ liệu khác như chuỗi, số nguyên hoặc số thực dấu chấm động, bạn có thể sử dụng lần lượt các phương thức getString(), getInt(), hoặc getFloat().
* Toast.makeText() và show() được sử dụng để hiển thị giá trị của cài đặt switchPref.

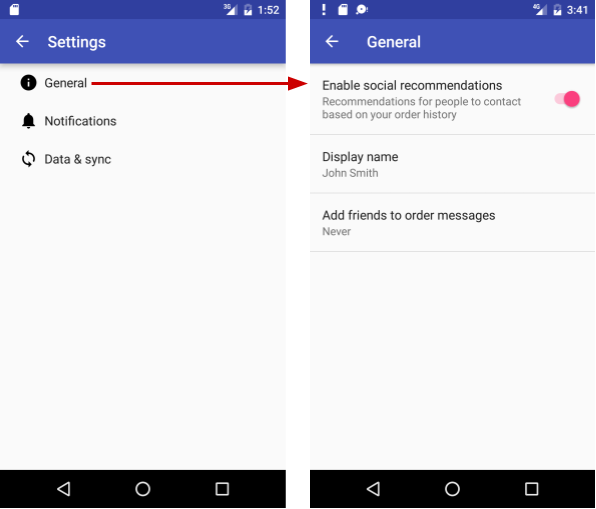
Mỗi khi **MainActivity** khởi động hoặc khởi động lại, phương thức **onCreate()** cần đọc các giá trị cài đặt để sử dụng trong ứng dụng. Trong phiên bản thực tế, phương thức Toast.makeText() sẽ được thay thế bằng một phương thức khởi tạo cài đặt.

Bạn đã có một **Settings Activity** hoạt động trong ứng dụng của mình.

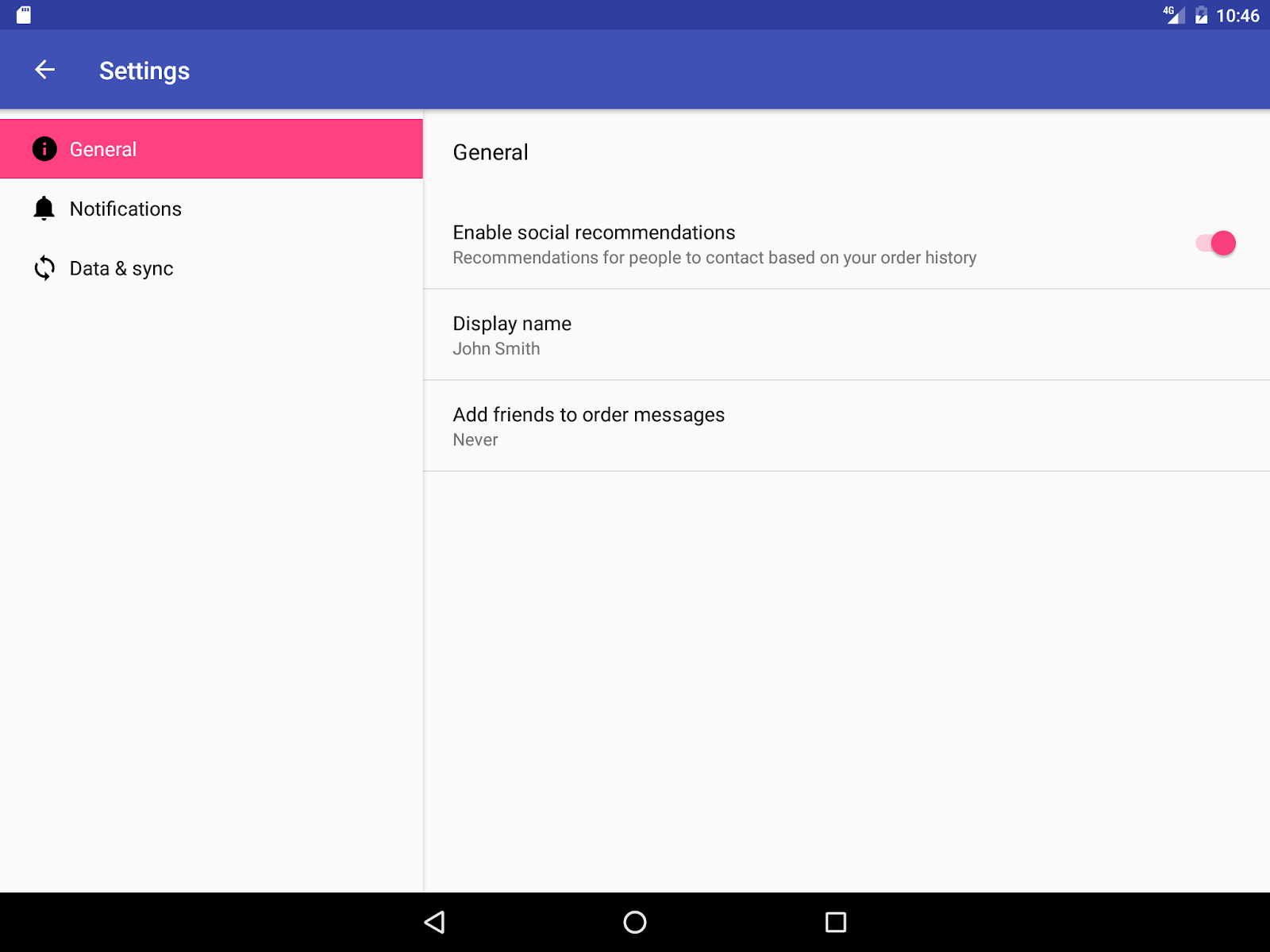
**Task 1: Mã giải pháp**  
Dự án Android Studio: **AppWithSettings**

**Task 2: Sử dụng mẫu (template) Settings Activity**  
Nếu bạn cần tạo nhiều màn hình con cho cài đặt và muốn tận dụng không gian màn hình của các thiết bị tablet cũng như đảm bảo tính tương thích với các phiên bản Android cũ hơn dành cho tablet, Android Studio cung cấp một giải pháp rút gọn: mẫu Settings Activity.  
Trong bài trước, bạn đã học cách sử dụng một Settings Activity trống và một Fragment trống để thêm cài đặt vào ứng dụng. Task 2 sẽ hướng dẫn bạn cách sử dụng mẫu Settings Activity có sẵn trong Android Studio để:

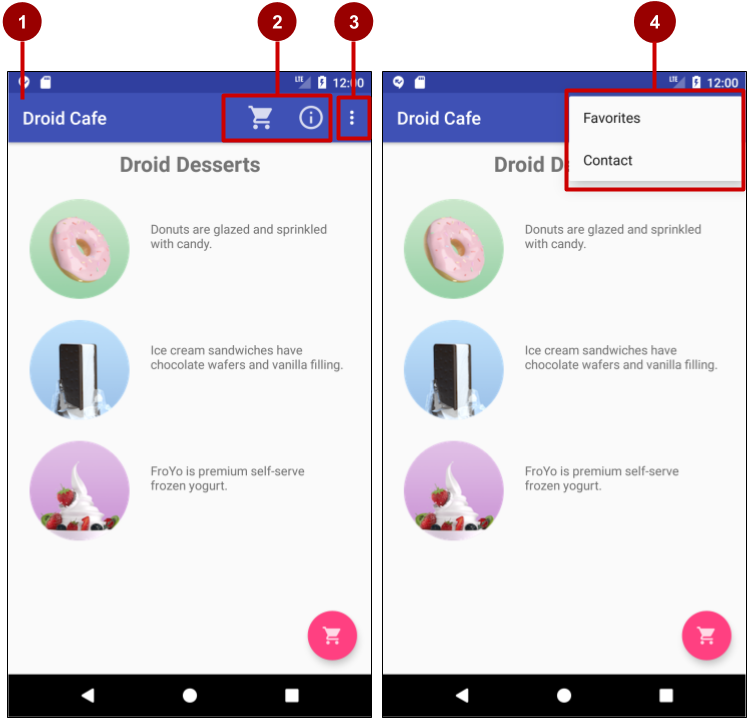
* **Phân chia** các cài đặt thành nhiều nhóm.
* **Tùy chỉnh** các cài đặt và giá trị của chúng.
* **Hiển thị** màn hình cài đặt chính với một tiêu đề liên kết cho mỗi nhóm cài đặt, chẳng hạn như nhóm cài đặt chung (General settings), như minh họa trong hình bên dưới.



Hiển thị giao diện màn hình master/detail với một liên kết tiêu đề cho mỗi nhóm ở phía bên trái (master) và nhóm cài đặt ở phía bên phải (detail), như minh họa trong hình bên dưới.



Trong bài thực hành trước, bạn đã tạo một ứng dụng có tên **DroidCafeOptionsUp** bằng cách sử dụng mẫu **Basic Activity**, mẫu này cung cấp một menu tùy chọn trên thanh ứng dụng như hình bên dưới.



**Chú giải cho hình trên:**

1. Thanh ứng dụng
2. Biểu tượng hành động trong menu tùy chọn
3. Nút Overflow
4. Menu tùy chọn overflow

**2.1 Khám phá mẫu Settings Activity**  
Để thêm mẫu Settings Activity vào dự án ứng dụng trong Android Studio, hãy làm theo các bước sau:

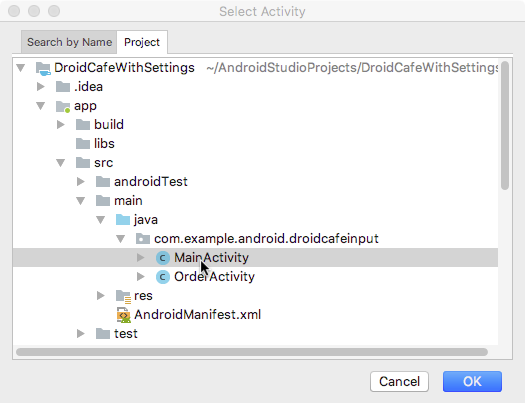
1. Sao chép thư mục dự án **DroidCafeOptionsUp** và đổi tên thành **DroidCafeWithSettings**. Chạy ứng dụng để đảm bảo rằng nó hoạt động bình thường.

2. Trong **Project** > **Android** (ở bảng điều hướng bên trái), chọn **app**, sau đó chọn **New** > **Activity** > **Settings Activity**.

3. Trong hộp thoại xuất hiện, chấp nhận tên Activity mặc định (**SettingsActivity** là tên được đề xuất) và tiêu đề (**Settings**).

4. Nhấp vào **ba dấu chấm** ở cuối menu **Hierarchical Parent**, sau đó trong hộp thoại **Select Activity**, chọn tab **Project** (xem hình bên dưới).

1. Mở rộng đường dẫn **DroidCafeWithSettings > app > src > main > java > com.example.android.droidcafeinput**, sau đó chọn **MainActivity** làm activity cha, như minh họa trong hình bên dưới. Nhấn **OK**.



Bạn chọn **MainActivity** làm activity cha để nút **Up** trên thanh ứng dụng trong **SettingsActivity** đưa người dùng quay lại **MainActivity**. Việc chọn activity cha sẽ tự động cập nhật tệp **AndroidManifest.xml** để hỗ trợ điều hướng bằng nút **Up**.

1. Nhấn **Finish**.
2. Trong bảng điều hướng **Project > Android**, mở rộng đường dẫn **app > res > xml** để xem các tệp XML được tạo bởi mẫu **Settings Activity**.

### Cài đặt ứng dụng

## Lưu trữ dữ liệu với Room

### Room, LiveData và ViewModel

### Room, LiveData và ViewModel