**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 5](#_Toc190855150)

[1.3) Trình chỉnh sửa bố cục 5](#_Toc190855151)

[1.4) Văn bản và các chế độ cuộn 5](#_Toc190855152)

[1.5) Tài nguyên có sẵn 5](#_Toc190855153)

[Bài 2) Activities 5](#_Toc190855154)

[2.1) Activity và Intent 5](#_Toc190855155)

[2.2) Vòng đời của Activity và trạng thái 5](#_Toc190855156)

[2.3) Intent ngầm định 5](#_Toc190855157)

[Bài 3) Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ 5](#_Toc190855158)

[3.1) Trình gỡ lỗi 5](#_Toc190855159)

[3.2) Kiểm thử đơn vị 5](#_Toc190855160)

[3.3) Thư viện hỗ trợ 5](#_Toc190855161)

[CHƯƠNG 2. Trải nghiệm người dùng 6](#_Toc190855162)

[Bài 1) Tương tác người dùng 6](#_Toc190855163)

[1.1) Hình ảnh có thể chọn 6](#_Toc190855164)

[1.2) Các điều khiển nhập liệu 6](#_Toc190855165)

[1.3) Menu và bộ chọn 6](#_Toc190855166)

[1.4) Điều hướng người dùng 6](#_Toc190855167)

[1.5) RecycleView 6](#_Toc190855168)

[Bài 2) Trải nghiệm người dùng thú vị 6](#_Toc190855169)

[2.1) Hình vẽ, định kiểu và chủ đề 6](#_Toc190855170)

[2.2) Thẻ và màu sắc 6](#_Toc190855171)

[2.3) Bố cục thích ứng 6](#_Toc190855172)

[Bài 3) Kiểm thử giao diện người dùng 6](#_Toc190855173)

[3.1) Espresso cho việc kiểm tra UI 6](#_Toc190855174)

[CHƯƠNG 3. Làm việc trong nền 6](#_Toc190855175)

[Bài 1) Các tác vụ nền 6](#_Toc190855176)

[1.1) AsyncTask 6](#_Toc190855177)

[1.2) AsyncTask và AsyncTaskLoader 6](#_Toc190855178)

[1.3) Broadcast receivers 6](#_Toc190855179)

[Bài 2) Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền 6](#_Toc190855180)

[2.1) Thông báo 6](#_Toc190855181)

[2.2) Trình quản lý cảnh báo 6](#_Toc190855182)

[2.3) JobScheduler 6](#_Toc190855183)

[CHƯƠNG 4. Lưu dữ liệu người dùng 7](#_Toc190855184)

[Bài 1) Tùy chọn và cài đặt 7](#_Toc190855185)

[1.1) Shared preferences 7](#_Toc190855186)

[1.2) Cài đặt ứng dụng 7](#_Toc190855187)

[Bài 2) Lưu trữ dữ liệu với Room 7](#_Toc190855188)

[2.1) Room, LiveData và ViewModel 7](#_Toc190855189)

[2.2) Room, LiveData và ViewModel 7](#_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

### Cài đặt ứng dụng

## Lưu trữ dữ liệu với Room

### Room, LiveData và ViewModel

### Room, LiveData và ViewModel

**Bài 9.1: Share Preferences**

**Giới thiệu**

Shared preferences cho phép bạn lưu trữ một lượng nhỏ dữ liệu nguyên thủy làm cặp khóa/giá trị trong một tệp trên thiết bị. Để có được xử lý tệp ưu tiên và để đọc, ghi và quản lý dữ liệu ưu tiên, sử dụng lớp SharedPreferences . Android framework quản lý chính tệp tùy chọn được chia sẻ. Tệp có thể truy cập được cho tất cả các thành phần của ứng dụng của bạn, nhưng không thể truy cập được cho các ứng dụng khác.

Dữ liệu bạn lưu vào các tùy chọn được chia sẻ khác với dữ liệu trong trạng thái hoạt động đã lưu, mà bạn đã học về một chương trước:

* Dữ liệu trong trạng thái thể hiện hoạt động đã lưu được giữ lại trong các trường hợp hoạt động trong cùng một người dùng phiên họp.
* Shared preferences vẫn tồn tại trong các phiên người dùng. Shared preferences vẫn tồn tại ngay cả khi ứng dụng của bạn dừng và khởi động lại, hoặc nếu thiết bị khởi động lại.

Chỉ sử dụng các Shared preferences khi bạn cần lưu một dữ liệu số lượng nhỏ dưới dạng các cặp khóa/giá trị đơn giản. Để quản lý số lượng lớn hơn của dữ liệu ứng dụng liên tục, hãy sử dụng phương pháp lưu trữ như thư viện phòng hoặc cơ sở dữ liệu SQL.

Những điều bạn nên biết

Bạn nên quen thuộc với:

* Tạo, xây dựng và chạy ứng dụng trong Android Studio.
* Thiết kế bố cục với các nút và chế độ xem văn bản.
* Sử dụng phong cách và chủ đề.
* Lưu và khôi phục trạng thái thể hiện hoạt động.

Những gì bạn sẽ học

Bạn sẽ học cách:

* Xác định sở thích chia sẻ là gì.
* Tạo một tệp tùy chọn được chia sẻ cho ứng dụng của bạn.
* Xóa dữ liệu trong các tùy chọn được chia sẻ.
* Cập nhật một ứng dụng để nó có thể lưu, truy xuất và đặt lại các tùy chọn được chia sẻ.

Tổng quan về ứng dụng

Ứng dụng HelloSharedPrefs là một biến thể khác của ứng dụng Hellotoast mà bạn đã tạo trong Bài 1. Nó bao gồm các nút để tăng số, để thay đổi màu nền và để đặt lại cả số lượng và màu sắc theo mặc định của họ. Ứng dụng cũng sử dụng các chủ đề và kiểu dáng để xác định các nút.

Bạn bắt đầu với ứng dụng khởi động và thêm các tùy chọn được chia sẻ vào mã hoạt động chính. Bạn cũng thêm một nút đặt lại đặt cả số đếm và màu nền cho mặc định và xóa tệp tùy chọn.

**Nhiệm vụ 1: Khám phá Hellosharedprefs**

Dự án ứng dụng khởi động hoàn chỉnh cho thực tế này có sẵn tại Hellosharedprefs Starter. Trong này

Nhiệm vụ bạn tải dự án vào Android Studio và khám phá một số tính năng chính của ứng dụng.

* 1. **Mở và chạy dự án HelloSharedPrefs**

Tải ứng dụng HelloSharedPrefs-Starter và giải nén tệp.

Mở dự án trong Android Studio, xây dựng và chạy ứng dụng.

* Nhấp vào **Count** đếm để tăng số trong chế độ xem văn bản chính.
* Nhấp vào bất kỳ nút màu nào để thay đổi màu nền của chế độ xem văn bản chính.
* Xoay thiết bị và lưu ý rằng cả màu nền và số lượng đều được bảo quản.
* Nhấp vào nút **Reset** để đặt màu và đếm lại mặc định.

Buộc đóng ứng dụng bằng một trong những cách sau.

* Trong Android Studio, chọn **Run > Stop 'app'** hoặc nhấp vào **Biểu tượng Dừng** trên thanh công cụ.
* Trên thiết bị, nhấn nút **Recents** (nút hình vuông ở góc dưới bên phải). Vuốt thẻ ứng dụng **HelloSharedPrefs** để thoát ứng dụng, hoặc nhấn vào dấu **X** ở góc phải của thẻ. Nếu bạn thoát ứng dụng theo cách này, hãy đợi vài giây trước khi khởi động lại để hệ thống có thời gian dọn dẹp.

Khởi động lại ứng dụng. Ứng dụng sẽ khởi động lại với giao diện mặc định—số đếm là 0 và màu nền là xám.

* 1. **Khám phá code của Activity.**

1. Mở MainActivity
2. Kiểm tra code và lưu ý những điều này:

* Số đếm (**mCount**) được định nghĩa là một số nguyên. Phương thức **countUp()** trong sự kiện **onClick** tăng giá trị này lên và cập nhật **TextView** chính.
* Màu sắc (**mColor**) cũng là một số nguyên, ban đầu được định nghĩa là màu xám trong tệp tài nguyên **colors.xml** với tên **default\_background**.
* Phương thức **changeBackground()** trong sự kiện **onClick** lấy màu nền của nút được nhấn, sau đó đặt màu đó cho **TextView** chính.
* Cả hai số nguyên **mCount** và **mColor** đều được lưu vào gói trạng thái phiên bản trong phương thức **onSaveInstanceState()**, và được khôi phục trong **onCreate()**. Các khóa của gói cho số đếm và màu sắc được định nghĩa bởi các biến riêng (**COUNT\_KEY**) và (**COLOR\_KEY**).

**Nhiệm vụ 2: Lưu và khôi phục dữ liệu vào tệp tùy chọn chia sẻ.**

Trong nhiệm vụ này, bạn lưu trạng thái của ứng dụng vào một tệp tùy chọn chia sẻ và đọc lại dữ liệu đó khi ứng dụng được khởi động lại. Vì dữ liệu trạng thái mà bạn lưu vào tùy chọn chia sẻ (số đếm hiện tại và màu sắc) là cùng dữ liệu mà bạn bảo toàn trong trạng thái phiên bản, bạn không cần phải thực hiện việc này hai lần. Bạn có thể thay thế hoàn toàn trạng thái phiên bản bằng trạng thái từ shared preference.

* 1. **Khởi tạo preferences**

1. Để thêm các biến thành viên (member variables) vào lớp MainActivity nhằm lưu tên của tệp Shared Preferences và một tham chiếu đến đối tượng SharedPreferences,

A screenshot of a computer

AI-generated content may be incorrect.

Bạn có thể đặt tên tệp Shared Preferences của mình bất kỳ như thế nào bạn muốn, nhưng theo thông lệ, nó thường có cùng tên với tên gói của ứng dụng của bạn.

1. Trong phương thức onCreate(), hãy khởi tạo Shared Preferences. Chèn đoạn mã này trước câu lệnh if:



Phương thức getSharedPreferences() (từ Context của hoạt động) mở tệp tại tên tệp đã cho (sharedPrefFile) với chế độ MODE\_PRIVATE.

Ghi chú: Các phiên bản cũ của Android có các chế độ khác cho phép bạn tạo tệp Shared Preferences có thể đọc được hoặc ghi được bởi toàn bộ hệ thống. Những chế độ này đã bị deprecated (không khuyến khích sử dụng) từ API 17 và hiện nay bị phản đối mạnh mẽ vì lý do bảo mật. Nếu bạn cần chia sẻ dữ liệu với các ứng dụng khác, hãy cân nhắc sử dụng content URI được cung cấp bởi FileProvider.

Code giải pháp cho lớp MainActivity, một phần:

A screen shot of a computer program

AI-generated content may be incorrect.

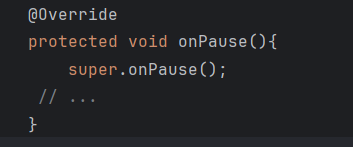
A computer screen shot of a program code

AI-generated content may be incorrect.

* 1. **Lưu** **Lưu các tùy chọn trong onPause()**

Việc lưu các tùy chọn rất giống với việc lưu trạng thái phiên bản -- cả hai thao tác đều lưu dữ liệu vào một đối tượng Bundle dưới dạng cặp khóa/giá trị. Tuy nhiên, đối với Shared Preferences, bạn lưu dữ liệu đó trong hàm gọi lại vòng đời onPause(), và bạn cần một đối tượng biên tập chia sẻ (SharedPreferences.Editor) để ghi vào đối tượng Shared Preferences.

1. Thêm phương thức vòng đời onPause() vào MainActivity



1. Trong onPause(), lấy một đối tượng biên tập (Editor) cho đối tượng SharedPreferences:



Một trình biên tập Shared Preferences là cần thiết để ghi vào đối tượng Shared Preferences. Thêm dòng này vào `onPause()` sau lời gọi đến `super.onPause()`.

1. Sử dụng phương thức `putInt()` để lưu cả hai số nguyên `mCount` và `mColor` vào Shared Preferences với các khóa phù hợp:

A screenshot of a computer screen

AI-generated content may be incorrect.

1. Gọi apply() để lưu các tùy chọn:



Phương thức `apply()` lưu các tùy chọn một cách bất đồng bộ, ngoài luồng giao diện người dùng (UI thread). Trình biên tập Shared Preferences cũng có phương thức `commit()` để lưu các tùy chọn một cách đồng bộ. Phương thức `commit()` không được khuyến khích vì nó có thể chặn các hoạt động khác.

1. Xóa toàn bộ phương thức `onSaveInstanceState()`. Vì trạng thái phiên bản của hoạt động chứa cùng dữ liệu với Shared Preferences, bạn có thể thay thế hoàn toàn trạng thái phiên bản.

Mã giải pháp cho phương thức `onPause()` của `MainActivity`:

A screen shot of a computer program

AI-generated content may be incorrect.

* 1. **Khôi phục tùy chỉnh trong onCreate()**

Cũng giống như trạng thái phiên làm việc, ứng dụng của bạn sẽ đọc bất kỳ tùy chỉnh đã lưu nào trong phương thức onCreate(). Một lần nữa, vì các tùy chỉnh dùng chung chứa cùng dữ liệu với trạng thái phiên làm việc, chúng ta có thể thay thế trạng thái bằng các tùy chỉnh ở đây. Mỗi khi onCreate() được gọi – khi ứng dụng khởi động hoặc khi có thay đổi cấu hình – các tùy chỉnh dùng chung sẽ được sử dụng để khôi phục trạng thái của giao diện.

1. Xác định phần của phương thức onCreate() kiểm tra xem đối số savedInstanceState có bằng null hay không và khôi phục trạng thái phiên làm việc.

A computer screen shot of a code

AI-generated content may be incorrect.

1. Xóa toàn bộ khối
2. Trong phương thức onCreate(), tại cùng vị trí nơi mã trạng thái phiên (instance state) đã được sử dụng, hãy lấy giá trị đếm từ SharedPreferences bằng khóa COUNT\_KEY và gán nó cho biến mCount.



Khi bạn đọc dữ liệu từ SharedPreferences, bạn không cần lấy một SharedPreferences.Editor. Hãy sử dụng bất kỳ phương thức "get" nào trên đối tượng SharedPreferences (chẳng hạn như getInt() hoặc getString()) để truy xuất dữ liệu từ SharedPreferences.

Lưu ý rằng phương thức getInt() nhận hai đối số: một là khóa để truy xuất giá trị, và một là giá trị mặc định nếu không tìm thấy khóa đó. Trong trường hợp này, giá trị mặc định là 0, giống với giá trị khởi tạo của mCount.

1. Cập nhật giá trị của TextView chính với số đếm mới.



1. Lấy màu từ SharedPreferences bằng khóa COLOR\_KEY và gán nó cho biến mColor.



Như trước đây, đối số thứ hai của getInt() là giá trị mặc định được sử dụng trong trường hợp khóa không tồn tại trong SharedPreferences. Trong trường hợp này, bạn có thể chỉ cần sử dụng lại giá trị của mColor, vì nó vừa được khởi tạo với màu nền mặc định ở phần trên của phương thức.

1. Cập nhật màu nền của TextView chính.



1. Chạy ứng dụng. Nhấn nút **Count** và thay đổi màu nền để cập nhật trạng thái phiên (instance state) và SharedPreferences.
2. Xoay thiết bị hoặc trình giả lập để kiểm tra rằng số đếm và màu sắc được lưu lại sau khi thay đổi cấu hình.
3. Buộc đóng ứng dụng bằng một trong các phương pháp sau:  
   ○ Trong Android Studio, chọn **Run > Stop 'app.'**  
   ○ Trên thiết bị, nhấn nút **Gần đây** (nút hình vuông ở góc dưới bên phải). Vuốt thẻ ứng dụng **HelloSharedPrefs** để đóng ứng dụng hoặc nhấn **X** ở góc phải của thẻ.
4. Chạy lại ứng dụng. Ứng dụng sẽ khởi động lại và tải SharedPreferences, duy trì trạng thái.

Code giải pháp cho phương thức onCreate() trong MainActivity:

**A computer screen shot of a program code

AI-generated content may be incorrect.**

* 1. **Đặt lại SharedPreferences trong trình xử lý sự kiện nhấp reset().**

Nút đặt lại trong ứng dụng ban đầu sẽ đặt lại cả số đếm và màu sắc của hoạt động về giá trị mặc định. Vì SharedPreferences lưu trạng thái của hoạt động, nên cũng cần xóa SharedPreferences cùng lúc.

1. Trong phương thức reset() của sự kiện onClick, sau khi màu sắc và số đếm được đặt lại, hãy lấy một Editor cho đối tượng SharedPreferences: 
2. Xóa tất cả dữ liệu trong SharedPreferences. 
3. Áp dụng các thay đổi.

Code giải pháp cho phương thức reset():A screen shot of a computer program

AI-generated content may be incorrect.

**CODE GIẢI PHÁP**  
Dự án Android Studio: **HelloSharedPrefs**

**Thử thách lập trình**

**Lưu ý:** Tất cả các thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

**Thử thách:** Hãy chỉnh sửa ứng dụng **HelloSharedPrefs** sao cho thay vì tự động lưu trạng thái vào tệp SharedPreferences, bạn sẽ thêm một **Activity** thứ hai để thay đổi, đặt lại và lưu các tùy chọn. Thêm một nút có tên **Settings** để mở activity này. Trong activity mới, sử dụng các **ToggleButton** và **Spinner** để chỉnh sửa tùy chọn. Thêm hai nút **Save** và **Reset** để lưu hoặc xóa SharedPreferences.

**Tóm tắt**

* Lớp **SharedPreferences** cho phép ứng dụng lưu trữ một lượng nhỏ dữ liệu kiểu nguyên thủy dưới dạng cặp **key-value**.
* **SharedPreferences** được duy trì qua các phiên làm việc khác nhau của cùng một ứng dụng.
* Để ghi dữ liệu vào **SharedPreferences**, cần lấy một đối tượng **SharedPreferences.Editor**.
* Sử dụng các phương thức **"put"** trong **SharedPreferences.Editor**, chẳng hạn như putInt() hoặc putString(), để lưu dữ liệu vào **SharedPreferences** với một **key** và một **value**.
* Sử dụng các phương thức **"get"** trong đối tượng **SharedPreferences**, chẳng hạn như getInt() hoặc getString(), để lấy dữ liệu từ **SharedPreferences** bằng một **key**.
* Sử dụng phương thức clear() trong đối tượng **SharedPreferences.Editor** để xóa toàn bộ dữ liệu được lưu trong **SharedPreferences**.
* Sử dụng phương thức apply() trong đối tượng **SharedPreferences.Editor** để lưu các thay đổi vào tệp **SharedPreferences**.

**Khái niệm liên quan**

Tài liệu về khái niệm liên quan có trong:

* **9.0: Lưu trữ dữ liệu**
* **9.1: SharedPreferences**
* **Tìm hiểu thêm**

**Tài liệu dành cho nhà phát triển Android:**

* **Tổng quan về lưu trữ dữ liệu và tệp**
* **Lưu dữ liệu dưới dạng cặp khóa-giá trị**
* **SharedPreferences**
* **SharedPreferences.Editor**

**Stack Overflow:**

* **Cách sử dụng SharedPreferences trong Android để lưu trữ, truy xuất và chỉnh sửa giá trị**
* **onSavedInstanceState vs. SharedPreferences**

**Bài tập về nhà**

**Xây dựng và chạy ứng dụng**  
Mở ứng dụng **ScoreKeeper** mà bạn đã tạo trong bài học **Android Fundamentals 5.1: Drawables, styles, and themes**.

1. Thay thế trạng thái phiên đã lưu bằng **SharedPreferences** cho từng điểm số.
2. Để kiểm tra ứng dụng, xoay thiết bị để đảm bảo rằng khi cấu hình thay đổi, ứng dụng đọc dữ liệu từ **SharedPreferences** và cập nhật giao diện người dùng.
3. Dừng ứng dụng và khởi động lại để kiểm tra xem dữ liệu có được lưu trong **SharedPreferences** hay không.
4. Thêm nút **Reset** để đặt lại điểm số về **0** và xóa dữ liệu trong **SharedPreferences**.

* **Trả lời các câu hỏi**

**Câu hỏi 1:**  
Trong phương thức vòng đời nào bạn lưu trạng thái ứng dụng vào **SharedPreferences**?

**Câu hỏi 2:**  
Trong phương thức vòng đời nào bạn khôi phục trạng thái ứng dụng?

**Câu hỏi 3:**  
Bạn có thể nghĩ ra trường hợp nào mà việc sử dụng cả **SharedPreferences** và **Instance State** là hợp lý không?

* **Nộp ứng dụng để chấm điểm**

**Hướng dẫn cho người chấm điểm**  
Kiểm tra xem ứng dụng có các tính năng sau không:

* Ứng dụng giữ nguyên điểm số khi thiết bị xoay.
* Ứng dụng giữ nguyên điểm số hiện tại sau khi bị dừng và khởi động lại.
* Ứng dụng lưu điểm số vào **SharedPreferences** trong phương thức onPause().
* Ứng dụng khôi phục **SharedPreferences** trong phương thức onCreate().
* Ứng dụng có một nút **Reset** để đặt lại điểm số về **0**.

**Đảm bảo rằng phương thức xử lý sự kiện onClick của nút Reset thực hiện các tác vụ sau:**

* Đặt cả hai biến điểm số về **0**.
* Cập nhật cả hai **TextView**.
* Xóa dữ liệu trong **SharedPreferences**.

**Bài học 9.2: Cài đặt ứng dụng**

**Giới thiệu**

Các ứng dụng thường bao gồm **cài đặt** để cho phép người dùng tùy chỉnh tính năng và hành vi của ứng dụng. Ví dụ:

Một số ứng dụng cho phép người dùng đặt **vị trí nhà**, Chọn **đơn vị đo lường mặc định**, Và các tùy chọn khác ảnh hưởng đến toàn bộ ứng dụng. Người dùng không thường xuyên truy cập cài đặt, vì sau khi thiết lập (chẳng hạn như vị trí nhà), họ hiếm khi cần thay đổi lại.

Người dùng mong đợi có thể truy cập **cài đặt ứng dụng** bằng cách: Nhấn vào mục **Settings** trong **thanh điều hướng bên** (navigation drawer) – như hình bên trái. Hoặc trong **menu tùy chọn** trên **thanh ứng dụng** (app bar) – như hình bên phải.

Trong hình trên:

1. **Settings** trong **thanh điều hướng bên** (navigation drawer).
2. **Settings** trong **menu tùy chọn** của **thanh ứng dụng** (app bar).

**Trong bài thực hành này, bạn sẽ thêm một Settings Activity vào ứng dụng.** Người dùng có thể điều hướng đến **cài đặt ứng dụng** bằng cách nhấn vào **Settings**, được đặt trong **menu tùy chọn** trên **thanh ứng dụng** (app bar).

**Những kiến thức bạn cần biết trước**

Bạn nên có khả năng:

* Tạo một **dự án Android Studio** từ mẫu và tạo bố cục chính.
* Chạy ứng dụng trên **trình giả lập** hoặc **thiết bị kết nối**.
* Tạo và chỉnh sửa các phần tử giao diện người dùng (UI) bằng **layout editor** và mã XML.
* Trích xuất tài nguyên chuỗi và chỉnh sửa giá trị chuỗi.
* Truy cập các phần tử giao diện từ mã bằng **findViewById()**.
* Xử lý sự kiện **nhấn nút (Button click)**.
* Hiển thị **Toast message**.
* Thêm một **Activity** vào ứng dụng.
* Tạo **menu tùy chọn** trên **thanh ứng dụng** (app bar).
* Thêm và chỉnh sửa **các mục menu** trong **menu tùy chọn**.
* Sử dụng **styles và themes** trong dự án.
* Sử dụng **SharedPreferences**.

Bạn sẽ học được gì?

Bạn sẽ học cách:

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

**Những gì bạn sẽ làm**

* Tạo một ứng dụng có mục **Settings** trong **menu tùy chọn**.
* Thêm một **công tắc bật/tắt (toggle switch)** cho tùy chọn **Settings**.
* Thêm mã để đặt giá trị mặc định cho cài đặt và truy cập giá trị cài đặt sau khi nó đã thay đổi.
* Sử dụng và tùy chỉnh mẫu **Settings Activity** của Android Studio.

**Tổng quan về ứng dụng**

Android Studio cung cấp một phím tắt để thiết lập menu tùy chọn với **Cài đặt**. Nếu bạn bắt đầu một dự án Android Studio cho điện thoại hoặc máy tính bảng bằng mẫu **Basic Activity**, ứng dụng mới sẽ bao gồm **Cài đặt**, như được hiển thị bên dưới:

ảnh

Mẫu này cũng bao gồm một nút hành động nổi (**Floating Action Button**) ở góc dưới bên phải của màn hình với biểu tượng phong bì. Bạn có thể bỏ qua nút này trong bài thực hành này, vì bạn sẽ không sử dụng nó.

Bạn sẽ bắt đầu bằng cách tạo một ứng dụng có tên **AppWithSettings** bằng mẫu **Basic Activity**, sau đó thêm một **Settings Activity**, cung cấp một cài đặt công tắc bật/tắt mà người dùng có thể chuyển đổi giữa bật hoặc tắt.

Bạn sẽ thêm mã để đọc giá trị cài đặt và thực hiện một hành động dựa trên giá trị đó. Để đơn giản, hành động này sẽ là hiển thị một thông báo **Toast** với giá trị của cài đặt.

Trong nhiệm vụ thứ hai, bạn sẽ thêm mẫu **Settings Activity** tiêu chuẩn do Android Studio cung cấp vào ứng dụng **DroidCafeOptionsUp** mà bạn đã tạo trong bài học trước.

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

* Điện thoại: Một màn hình **Cài đặt chính** với liên kết tiêu đề cho từng nhóm cài đặt, chẳng hạn như **General** cho cài đặt chung, như hiển thị bên dưới.

ảnh

* Máy tính bảng: Một bố cục màn hình **chính/chi tiết**, với liên kết tiêu đề cho từng nhóm cài đặt ở bên trái (màn hình chính) và nhóm cài đặt tương ứng ở bên phải (màn hình chi tiết), như hiển thị trong hình bên dưới.

ảnh

Để tùy chỉnh mẫu, bạn sẽ thay đổi tiêu đề, tiêu đề cài đặt, mô tả cài đặt và giá trị của các cài đặt.

Ứng dụng **DroidCafeOptionsUp** đã được tạo trong bài học trước từ mẫu **Basic Activity**, mẫu này cung cấp một menu tùy chọn trên thanh ứng dụng để đặt tùy chọn **Cài đặt**.

Bạn sẽ tùy chỉnh mẫu **Settings Activity** được cung cấp bằng cách thay đổi tiêu đề, mô tả, giá trị và giá trị mặc định của một cài đặt. Bạn cũng sẽ thêm mã để đọc giá trị cài đặt sau khi người dùng thay đổi nó và hiển thị giá trị đó.

**Nhiệm vụ 1: Thêm một cài đặt công tắc (Switch Setting) vào ứng dụng**

Trong nhiệm vụ này, bạn sẽ thực hiện các bước sau:

● Tạo một dự án mới dựa trên mẫu Basic Activity, mẫu này cung cấp một menu tùy chọn.  
● Thêm một công tắc bật/tắt (SwitchPreference) với các thuộc tính trong tệp XML preference.  
● Thêm một Activity cho cài đặt và một Fragment cho một cài đặt cụ thể. Để duy trì khả năng tương thích với AppCompatActivity, bạn sẽ sử dụng PreferenceFragmentCompat thay vì PreferenceFragment. Bạn cũng cần thêm thư viện android.support.v7.preference.  
● Kết nối mục Cài đặt (Settings) trong menu tùy chọn với Settings Activity.

**1.1 Tạo dự án và thêm thư mục XML cùng tệp tài nguyên**

1. Trong **Android Studio**, tạo một dự án mới với các tham số sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Thuộc tính | |  | | --- | |  |  |  | | --- | | **Giá trị** | |
| Tên ứng dụng | AppWithSettings |
| Tên công ty | android.example.com (hoặc tên miền của bạn) |
| |  | | --- | | Vị trí dự án (**Project location**) |  |  | | --- | |  | | Đường dẫn đến thư mục chứa dự án của bạn |
| SDK tối thiểu cho điện thoại và máy tính bảng | API 15: Android 4.0.3 IceCreamSandwich |
| Mẫu | Basic Activity |
| Tên Activity | MainActivity |
| **Tên Layout** | activity\_main |
| Tiêu đề | MainActivity |
|  |  |
|  |  |

1. Chạy ứng dụng và nhấn vào biểu tượng **overflow** trên thanh ứng dụng để xem menu tùy chọn, như hiển thị trong hình bên dưới. Mục duy nhất trong menu tùy chọn là **Settings**.

ảnh

1. Bạn cần tạo một thư mục tài nguyên mới để chứa tệp XML cài đặt.Chọn thư mục **res** trong bảng **Project > Android**, sau đó chọn **File > New > Android Resource Directory**. Hộp thoại **New Resource Directory** sẽ xuất hiện.
2. Trong menu thả xuống **Resource type**, chọn **xml**. **Tên thư mục** sẽ tự động thay đổi thành **xml**. Nhấn **OK**.
3. Thư mục **xml** sẽ xuất hiện trong bảng **Project > Android** bên trong thư mục **res**. Chọn **xml** và chọn **File > New > XML resource file** (hoặc nhấp chuột phải vào **xml** và chọn **New > XML resource file**).
4. Nhập tên tệp XML là **preferences** vào trường **File name**, sau đó nhấn **OK**. Tệp **preferences.xml** sẽ xuất hiện trong thư mục **xml**, và trình chỉnh sửa giao diện (**layout editor**) sẽ hiển thị, như trong hình bên dưới.

ảnh trang 23

Trong hình trên:

1. Tệp **preferences.xml** bên trong thư mục **xml**.
2. Trình chỉnh sửa giao diện (**layout editor**) hiển thị nội dung của **preferences.xml**.

**1.2. Thêm Preference XML và các thuộc tính cho cài đặt.**

1. Kéo một SwitchPreference từ bảng Palette ở bên trái vào đầu bố cục, như hiển thị trong hình bên dưới.

ảnh

1. Thay đổi các giá trị trong bảng **Attributes** ở bên phải của trình chỉnh sửa giao diện như sau, và như hiển thị trong hình bên dưới:

○ **defaultValue**: true  
○ **key**: example\_switch

○ **title**: Settings option  
○ **summary**: Turn this option on or off

ảnh

1. Nhấp vào tab **Text** ở dưới cùng của trình chỉnh sửa giao diện để xem mã XML.

ảnh

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

Các thuộc tính XML cho một **Preference** bao gồm:

● **android:defaultValue**: Giá trị mặc định của cài đặt khi ứng dụng khởi chạy lần đầu tiên.  
● **android:title**: Tiêu đề của cài đặt. Đối với **SwitchPreference**, tiêu đề xuất hiện bên trái của công tắc bật/tắt.  
● **android:key**: Khóa dùng để lưu trữ giá trị cài đặt. Mỗi cài đặt có một cặp khóa-giá trị tương ứng mà hệ thống sử dụng để lưu trữ cài đặt trong tệp **SharedPreferences** mặc định của ứng dụng.  
● **android:summary**: Văn bản mô tả xuất hiện bên dưới cài đặt.

**1.3 Sử dụng SwitchPreferenceCompat**

Để sử dụng **PreferenceFragmentCompat** thay cho **PreferenceFragment**, bạn cũng 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 dòng sau vào phần **dependencies**:

ảnh

Câu lệnh trên thêm thư viện **android.support.v7.preference** để sử dụng **PreferenceFragmentCompat** thay cho **PreferenceFragment**.

* 1. Trong tệp **preferences.xml** trong thư mục **xml**, thay đổi <SwitchPreference trong mã thành <android.support.v7.preference.SwitchPreferenceCompat.

ảnh

Dòng **SwitchPreferenceCompat** ở trên có thể hiển thị biểu tượng bóng đèn màu vàng với cảnh báo, nhưng bạn có thể bỏ qua nó ngay bây giờ.

* 1. Mở tệp **styles.xml** trong thư mục **values**, và thêm mục **preferenceTheme** sau vào phần khai báo **AppTheme**:

ảnh

Để sử dụng **PreferenceFragmentCompat**, bạn cũng phải khai báo **preferenceTheme** với kiểu **PreferenceThemeOverlay** trong chủ đề của ứng dụng.

**1.4 Thêm một Activity cho cài đặt**

Để tạo một **Settings Activity** cung cấp giao diện người dùng cho cài đặt, hãy thêm một **Empty Activity** vào ứng dụng. Thực hiện các bước sau:

1. Chọn **app** ở đầu bảng **Project > Android**, sau đó chọn **New > Activity > Empty Activity**.
2. Đặt tên **Activity** là **SettingsActivity**. Bỏ chọn tùy chọn **Generate Layout File** (vì bạn không cần tệp giao diện), và giữ nguyên tùy chọn **Launcher Activity** ở trạng thái bỏ chọn.
3. Giữ nguyên tùy chọn **Backwards Compatibility (AppCompat)** được chọn. **Package name** sẽ tự động được đặt thành **com.example.android.projectname**.
4. Chọn kết thúc.

**1.5 Thêm một Fragment cho một cài đặt cụ thể**

Một **Fragment** giống như một phần độc lập của **Activity** — nó có vòng đời riêng, nhận các sự kiện đầu vào riêng và có thể được thêm hoặc xóa trong khi **Activity** đang chạy.

Bạn sẽ sử dụng một lớp con chuyên biệt của **Fragment** để hiển thị danh sách cài đặt. Cách tốt nhất là sử dụng một **Activity** thông thường để chứa một **PreferenceFragment**, hiển thị cài đặt của ứng dụng. **PreferenceFragment** cung cấp một kiến trúc linh hoạt hơn so với việc sử dụng **Activity** để quản lý cài đặt.

Bạn sẽ sử dụng **PreferenceFragmentCompat** thay vì **PreferenceFragment** để đảm bảo khả năng tương thích với **AppCompatActivity**.

Trong bước này, bạn sẽ thêm một **Blank Fragment** để nhóm các cài đặt tương tự (không có layout, phương thức factory hoặc interface callbacks) vào ứng dụng và mở rộng từ **PreferenceFragmentCompat**.

Thực hiện các bước sau:

1. Chọn **app**, sau đó chọn **New > Fragment > Fragment (Blank)**.
2. Đặt tên **Fragment** là **SettingsFragment**. Bỏ chọn tùy chọn **Create layout XML?** (vì bạn không cần tệp giao diện).
3. Bỏ chọn các tùy chọn để thêm **fragment factory methods** và **interface callbacks**.
4. **Target Source Set** nên được đặt thành **main**.
5. Nhấn **Finish**. Kết quả là lớp **SettingsFragment** sau sẽ được tạo:

ảnh

1. Chỉnh sửa phần định nghĩa lớp **SettingsFragment** để mở rộng từ **PreferenceFragmentCompat**.

ảnh

Khi bạn chỉnh sửa định nghĩa lớp để khớp với định nghĩa ở trên, một biểu tượng bóng đèn đỏ sẽ xuất hiện ở lề bên trái. Nhấp vào bóng đèn đỏ và chọn **Implement methods**, sau đó chọn **onCreatePreferences**. Android Studio sẽ tạo ra một phương thức **onCreatePreferences()** mẫu như sau:

ảnh

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

ảnh

1. Xóa toàn bộ phương thức **onCreateView()** trong **Fragment**.

Lý do bạn thay thế **onCreateView()** bằng **onCreatePreferences()** là vì bạn sẽ thêm **SettingsFragment** vào **SettingsActivity** hiện có để hiển thị cài đặt, thay vì hiển thị một màn hình **Fragment** riêng biệt. Việc thêm **Fragment** vào **Activity** giúp bạn dễ dàng thêm hoặc xóa **Fragment** trong khi **Activity** đang chạy. **PreferenceFragment** được gắn vào **PreferenceScreen** bằng **rootKey**.

Bạn cũng có thể xóa an toàn constructor rỗng của **SettingsFragment**, vì **Fragment** này không được hiển thị độc lập.

ảnh