

Change app language programmatically in Android

Asked 14 years, 7 months ago Modified 9 months ago

Viewed 539k times  Part of [Mobile Development](#) Collective



Is it possible to change the language of an app programmatically while still using Android resources?

583



If not, is it possible to request a resource in an specific language?



I would like to let the user change the language of the app from the app.



MD

android

localization

resources

Share

Improve this question

Follow

edited Feb 4, 2018 at 21:52



[Mathieu K.](#)

283 ● 3 ● 14

asked May 24, 2010 at 20:17



[hpique](#)

120k ● 131 ● 342 ● 481

-
- 4 You can use the following library, which provides the language list, the preference for your settings screen, and overrides the language in your application:

github.com/delight-im/Android-Languages – caw Mar 24, 2014 at 2:01

@MarcoW. Do you know if Android-Languages works with Android 5.0 Lollipop? – neu242 Jan 8, 2015 at 12:41

1 @neu242 Yes, it runs on Android 5.0 without any problems. – caw Jan 8, 2015 at 17:03

2 You can use the following library: github.com/zeugma-solutions/locale-helper-android – josue.0 Jun 12, 2019 at 1:06

1 @josue.0 that library really is the cleanest solution around for this – amitavk Aug 30, 2019 at 11:18

36 Answers

Sorted by:

Highest score (default)



1

2

Next



466

It's possible. You can set the locale. However, I would not recommend that. We've tried it at early stages, it's basically fighting the system.



We have the same requirement for changing the language but decided to settle to the fact that UI should be same as phone UI. It was working via setting locale but was too buggy. And you have to set it every time you enter activity (each activity) from my experience. here is a code if you still need this (again, I don't recommend that)



```
Resources res = context.getResources();  
// Change locale settings in the app.  
DisplayMetrics dm = res.getDisplayMetrics();  
android.content.res.Configuration conf = res.getConfig
```

```
conf.setLocale(new Locale(language_code.toLowerCase()))
// Use conf.locale = new Locale(...) if targeting lowe
res.updateConfiguration(conf, dm);
```

If you have language specific content - you can change that base on the setting.

update on 26th of march 2020

```
public static void setLocale(Activity activity, St
    Locale locale = new Locale(languageCode);
    Locale.setDefault(locale);
    Resources resources = activity.getResources();
    Configuration config = resources.getConfigurat
    config.setLocale(locale);
    resources.updateConfiguration(config, resource
}
```

- NOTES: Language code cannot got '-' & must be 2 small case letter only

Share Improve this answer

Follow

edited Sep 15, 2021 at 10:53



Ashraf Amin

411 ● 5 ● 9

answered May 24, 2010 at 20:36



Alex Volovoy

68.4k ● 13 ● 76 ● 54

487 Can't believe that Android makes this so hard. I do not really see why there should be a STRICT association between the phone's locale and the application's. I always have my phone using English language although I'm not a native English speaker. The reason is that the translated

semi-technical words just gets too weird in my own language so English is just so much easier. It also makes it easier for me to follow advice from the Net. But that does not mean that I want EVERY app on my phone to use English (although perfectly ok that is default). I want to be able to choose !!! – [peterh](#) Apr 29, 2013 at 7:28

15 Oh, looks like API level 17 introduced `Context.createConfigurationContext()`, which can be used to wrap the default context with locale-specific configuration and then call `getResources` on that without having to update the configuration on the resources objects themselves. – [JAB](#) Apr 9, 2014 at 17:39 ✎

15 You need to put this in `onCreate()` of every activity. Otherwise it may get overridden by the system - for instance when you turn your device to landscape and your activity gets recreated with new (system provided) configuration. – [Zsolt Safrany](#) Jun 29, 2014 at 13:08

19 In case you set a RTL locale like "ar" and want your -ldrtl resource folders to work as well then also call `conf.setLayoutDirection(locale);` – [Zsolt Safrany](#) Jun 29, 2014 at 13:27

6 @ZsoltSafrany - Rather than adding a call to `conf.setLayoutDirection(locale)`, you can replace `conf.locale = new Locale(...)` with `conf.setLocale(new Locale(...))`. It will internally call `setLayoutDirection`. – [Ted Hopp](#) ★ Sep 1, 2015 at 4:29



This code really works:

232

fa = Persian, en = English



- NOTES: Language code cannot got '-' & must be 2 small case letter only



Enter your language code in `languageToLoad` variable:

```
import android.app.Activity;
import android.content.res.Configuration;
import android.os.Bundle;

public class Main extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        String languageToLoad = "fa"; // your language
        Locale locale = new Locale(languageToLoad);
        Locale.setDefault(locale);
        Configuration config = new Configuration();
        config.locale = locale;
        getBaseContext().getResources().updateConfiguration(
            getBaseContext().getResources().getDisplayMetrics(),
            config);
        this setContentView(R.layout.main);
    }
}
```

UPDATE on Jun 2021(Kotlin):

```
class Main : Activity() {
    // Called when the activity is first created.
    public override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        val config = resources.configuration
        val lang = "fa" // your language code
        val locale = Locale(lang)
        Locale.setDefault(locale)
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
            config.setLocale(locale)
        }
    }
}
```

```

else
    config.locale = locale

    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP)
        createConfigurationContext(config)
        resources.updateConfiguration(config, resources.configuration)

    this setContentView(R.layout.main)
}
}

```

Share Improve this answer

edited Sep 15, 2021 at 10:13

Follow



Ashraf Amin

411 ● 5 ● 9

answered Feb 7, 2012 at 9:10



AliSh

10.6k ● 5 ● 50 ● 82

4 I want to change the locale at runtime, in your code, you put your code before setContentView() method. So your code is not useful for me, So How to change the language at runtime, In my application, there are two radio button, one for English and other one for Arabic , – [Ashish Dwivedi](#) Sep 14, 2012 at 5:48 ✎

2 @Buffalo, it's just the second argument for the `Resources.updateConfiguration` method. I've indented the code to make it more clear. – [Czechnology](#) Feb 16, 2013 at 20:37

7 This is working well for all activities upon setting in the launching activity. But, the action bar title seems unaffected and still continues to display the default language. Any idea what I might have missed? – [Viral Patel](#) Oct 30, 2015 at 5:11

11 Config.locale is deprecated – [Zoe - Save the data dump](#) ♦ Jul 19, 2017 at 15:43

5 instead of "config.locale = locale;" use "if (Build.VERSION.SDK_INT >= 17) { config.setLocale(locale); } else { config.locale = locale; } – [roghayeh hosseini](#) Aug 25, 2018 at 12:47



38



I was looking for a way to change the system language programmatically. While I fully understand that a normal application should never do that and instead either:

- the user should be pointed(through an intent) to the system settings to change it manually
- the application should handle its localization on its own just like described in the answer of Alex

there was a need to really change the language of the system programmatically.

This is undocumented API and thus should not be used for market/end-user applications!

Anyway heres the solution i found:

```
Locale locale = new Locale(targetLocaleAsString);

Class amnClass = Class.forName("android.app.ActivityManagerNative");
Object amn = null;
Configuration config = null;

// amn = ActivityManagerNative.getDefault();
Method methodGetDefault = amnClass.getMethod("getDefault");
methodGetDefault.setAccessible(true);
amn = methodGetDefault.invoke(amnClass);

// config = amn.getConfiguration();
```

```

Method methodGetConfiguration = amnClass.getMethod("
methodGetConfiguration.setAccessible(true);
config = (Configuration) methodGetConfiguration.invo

// config.setUserLocale = true;
Class configClass = config.getClass();
Field f = configClass.getField("userSetLocale");
f.setBoolean(config, true);

// set the locale to the new value
config.locale = locale;

// amn.updateConfiguration(config);
Method methodUpdateConfiguration = amnClass.getMetho
Configuration.class);
methodUpdateConfiguration.setAccessible(true);
methodUpdateConfiguration.invoke(amn, config);

```

Share Improve this answer

answered Jan 13, 2011 at 18:09


Follow



icyerasor

5,222 ● 1 ● 47 ● 57

-
- 2 give exception invocationtarget exception – [Ravi](#) Jun 13, 2013 at 6:54
-
- 1 Well depends where the invocationTargetException gets thrown. Then you should know the class that was changed. – [icyerasor](#) Jun 13, 2013 at 12:25
-
- 1 @Rat-a-tat-a-tat Ratatouille ,starting from Android 4.2 the `android.permission.CHANGE_CONFIGURATION` can only be granted by app signed with perform key. – [Yeung](#) May 5, 2014 at 9:34
-
- 3 I put my app in /system/priv-app to work around the Android 6.0 issue. [Details here](#). – [weiyin](#) Aug 26, 2015 at 2:10
-

- 3 API level 24 onwards, there's possibility to set multiple languages with [setLocales](#) – Juuso Ohtonen Mar 28, 2017 at 19:58 
-



If you want to maintain the language changed over all your app you have to do two things.

36



First, create a base Activity and make all your activities extend from this:



```
public class BaseActivity extends AppCompatActivity {

    private Locale mCurrentLocale;

    @Override
    protected void onStart() {
        super.onStart();

        mCurrentLocale = getResources().getConfiguration()
            .getLocales().get(0);
    }

    @Override
    protected void onRestart() {
        super.onRestart();
        Locale locale = getLocale(this);

        if (!locale.equals(mCurrentLocale)) {
            mCurrentLocale = locale;
            recreate();
        }
    }

    public static Locale getLocale(Context context){
        SharedPreferences sharedPreferences =
            PreferenceManager.getDefaultSharedPreferences(context)

        String lang = sharedPreferences.getString("lan
```

```

        switch (lang) {
            case "English":
                lang = "en";
                break;
            case "Spanish":
                lang = "es";
                break;
        }
        return new Locale(lang);
    }
}

```

Note that I save the new language in a sharedPreferences.

Second, create an extension of Application like this:

```

public class App extends Application {

    @Override
    public void onCreate() {
        super.onCreate();
        setLocale();
    }

    @Override
    public void onConfigurationChanged(Configuration newConfig) {
        super.onConfigurationChanged(newConfig);
        setLocale();
    }

    private void setLocale() {

        final Resources resources = getResources();
        final Configuration configuration = resources.getConfiguration();
        final Locale locale = getLocale(this);
        if (!configuration.locale.equals(locale)) {
            configuration.setLocale(locale);
            resources.updateConfiguration(configuration);
        }
    }
}

```

Note that `getLocale()` it's the same as above.

That's all! I hope this can help somebody.

Share Improve this answer

edited Feb 16, 2017 at 16:21

Follow

answered Feb 25, 2016 at 3:02



Daniel S.

2,677 ● 2 ● 22 ● 19

App activity it is a primary activity, like a MainActivity? for example i can resolve this in `setLocale()` in my method `onCreate()`? – [Morozov](#) Feb 15, 2017 at 17:14

App is an extension of Application, it's not an Activity. I do not understand what you need, sorry. Maybe you can try to explain me again :) – [Daniel S.](#) Feb 16, 2017 at 16:21

3 for those Android noobs like me, come here to learn what the `Application` is and how to use.
mobomo.com/2011/05/how-to-use-application-object-of-android – [Siwei](#) Mar 5, 2017 at 10:03

2 `configuration.locale` is deprecated, `setLocale` requires API 17+ and `updateConfiguration` is deprecated
– [Zoe - Save the data dump](#) ♦ Jul 19, 2017 at 15:44



28



According to [this article](#). You will need to download `LocaleHelper.java` referenced in that article.

1. Create `MyApplication` class that will extends `Application`
2. Override `attachBaseContext()` to update language.
3. Register this class in manifest.

```
public class MyApplication extends Application {
    @Override
    protected void attachBaseContext(Context base) {
        super.attachBaseContext(LocaleHelper.onAttach(
    })
}

<application
    android:name="com.package.MyApplication"
.../>
```

4. Create `BaseActivity` and override `onAttach()` to update language. **Needed for Android 6+**

```
public class BaseActivity extends Activity {
    @Override
    protected void attachBaseContext(Context base) {
        super.attachBaseContext(LocaleHelper.onAttach(
    })
}
```

5. Make all activities on your app extends from `BaseActivity`.

```
public class LocaleHelper {

    private static final String SELECTED_LANGUAGE =
        "Locale.Helper.Selected.Language";
```

```

public static Context onAttach(Context context) {
    String lang = getPersistedData(context, Locale
    return setLocale(context, lang);
}

public static Context onAttach(Context context, St
    String lang = getPersistedData(context, default
    return setLocale(context, lang);
}

public static String getLanguage(Context context)
    return getPersistedData(context, Locale.getDef
}

public static Context setLocale(Context context, S
    persist(context, language);

    if (Build.VERSION.SDK_INT >= Build.VERSION_COD
        return updateResources(context, language);
    }

    return updateResourcesLegacy(context, language
}

private static String getPersistedData(Context con
{
    SharedPreferences preferences =
    PreferenceManager.getDefaultSharedPreferences(cont
    return preferences.getString(SELECTED_LANGUAGE
}

private static void persist(Context context, Strin
    SharedPreferences preferences =
    PreferenceManager.getDefaultSharedPreferences(cont
    SharedPreferences.Editor editor = preferences.

    editor.putString(SELECTED_LANGUAGE, language);
    editor.apply();
}

@TargetApi(Build.VERSION_CODES.N)
private static Context updateResources(Context con
    Locale locale = new Locale(language);

```

```

        Locale.setDefault(locale);

        Configuration configuration = context.getResources().getConfiguration();
        configuration.setLocale(locale);
        configuration.setLayoutDirection(locale);

        return context.createConfigurationContext(configuration);
    }

    @SuppressWarnings("deprecation")
    private static Context updateResourcesLegacy(Context context) {
        Locale locale = new Locale(language);
        Locale.setDefault(locale);

        Resources resources = context.getResources();

        Configuration configuration = resources.getConfiguration();
        configuration.locale = locale;
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.JELLY_BEAN_MR1) {
            configuration.setLayoutDirection(locale);
        }

        resources.updateConfiguration(configuration,
            resources.getDisplayMetrics());

        return context;
    }
}

```

Share Improve this answer

Follow

edited Oct 25, 2018 at 11:27



Flimm

150k ● 48 ● 272 ● 285

answered Jan 30, 2018 at 22:53



Khaled Lela

8,109 ● 7 ● 47 ● 74

can't use

super.attachBaseContext(LocaleHelper.onAttach(newBase))

cause I already using
super.attachBaseContext(CalligraphyContextWrapper.wrap(newBase)) – [Rasel](#) Dec 2, 2019 at 15:58

- 2 you can wrap one with another.
super.attachBaseContext(CalligraphyContextWrapper.wrap(LocaleHelper.onAttach(newBase))) – [Yeahia Md Arif](#) Apr 8, 2020 at 7:39
-



16



Just adding an extra piece that tripped me up.

While the other answers work fine with "de" for example

```
String lang = "de";
Locale locale = new Locale(lang);
Locale.setDefault(locale);
Configuration config = new Configuration();
config.locale = locale;
getBaseContext().getResources().updateConfiguration(config,
getBaseContext().getResources().getDisplayMetrics()
```

The above wont work with for example "fr_BE" locale so it would use the values-fr-rBE folder or similar.

Needs the following slight change to work with "fr_BE"

```
String lang = "fr";

//create a string for country
String country = "BE";
//use constructor with country
Locale locale = new Locale(lang, country);

Locale.setDefault(locale);
Configuration config = new Configuration();
config.locale = locale;
```

```
getBaseContext().getResources().updateConfiguration(co  
getBaseContext().getResources().getDisplayMetrics(
```

Share Improve this answer

answered Sep 25, 2013 at 10:01

Follow



triggs

5,900 ● 3 ● 33 ● 31

-
- 2 if you want to apply locale change to current opened activity call `activity.recreate()` – [To Kra](#) Mar 30, 2015 at 9:49

I know I'm late to the party, but the new `Locale(lang, country)` was all I needed! – [Jacob Holloway](#) Apr 22, 2015 at 20:22 ✎

`activity.recreate()` how it works or if we cal this then `String lang = "fr";String country = "BE";` will never override how it will run time – [Amitsharma](#) Jun 2, 2015 at 12:18

-
- 1 What about using `android.content.res.Configuration conf = res.getConfiguration();` instead of creating a new `Configuration` instance? Is there any benefit for using a fresh one? – [Bianca Daniciuc](#) Aug 14, 2015 at 14:42

how about `layoutDirection='locale'` ? – [nAkhmedov](#) Sep 21, 2021 at 14:43



Create a class Extends `Application` and create a static method. Then you can call this method in all activities

16

before `setContentView()` .



```
public class MyApp extends Application {  
  
    @Override  
    public void onCreate() {  
        super.onCreate();  
    }  
}
```




```

public static void setLocaleFa (Context context){
    Locale locale = new Locale("fa");
    Locale.setDefault(locale);
    Configuration config = new Configuration();
    config.locale = locale;
    context.getApplicationContext().getResources().updateConfiguration(config, null);
}

public static void setLocaleEn (Context context){
    Locale locale = new Locale("en_US");
    Locale.setDefault(locale);
    Configuration config = new Configuration();
    config.locale = locale;
    context.getApplicationContext().getResources().updateConfiguration(config, null);
}
}

```

Usage in activities:

```

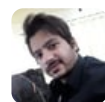
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    MyApp.setLocaleFa(MainActivity.this);
    requestWindowFeature(Window.FEATURE_NO_TITLE);
    setContentView(R.layout.activity_main);
}

```

Share Improve this answer

Follow

edited Apr 5, 2018 at 7:26



Vishal Yadav

3,682 ● 3 ● 27 ● 42

answered Jun 17, 2015 at 7:58



Behzad Taghipour

161 ● 1 ● 2



I am changed for German language for my app start itself.

15



Here is my correct code. Anyone want use this same for me.. (How to change language in android programmatically)



my code:



```
Configuration config ; // variable declaration in glob

// this part is given inside onCreate Method starting
setContentView()

public void onCreate(Bundle icic)
{
    super.onCreate(icic);
    config = new Configuration(getResources().getConfi
    config.locale = Locale.GERMAN ;

    getResources().updateConfiguration(config, getResources

    setContentView(R.layout.newdesign);
}
```

Share Improve this answer

edited Feb 19, 2013 at 14:15

Follow



Kuitsi

1,675 ● 2 ● 28 ● 48

answered Feb 19, 2013 at 6:59



harikrishnan

1,945 ● 4 ● 32 ● 67

- 1 @harikrishnan Its not working for me and keyboard is not changing to the specified language.. How you have declared



14

I know it's late to answer but i found [this article here](#) . Which explains the whole process very well and provides you a well structured code.



Locale Helper class:



```
import android.annotation.TargetApi;
import android.content.Context;
import android.content.SharedPreferences;
import android.content.res.Configuration;
import android.content.res.Resources;
import android.os.Build;
import android.preference.PreferenceManager;

import java.util.Locale;

/**
 * This class is used to change your application local
 * for the next time
 * that your app is going to be used.
 * <p/>
 * You can also change the locale of your application
 * setLocale method.
 * <p/>
 * Created by gunhansancar on 07/10/15.
 */
public class LocaleHelper {

    private static final String SELECTED_LANGUAGE =
        "Locale.Helper.Selected.Language";

    public static Context onAttach(Context context) {
        String lang = getPersistedData(context,
            Locale.getDefault().getLanguage());
        return setLocale(context, lang);
    }
}
```

```

    public static Context onAttach(Context context, String lang = getPersistedData(context, defaultLanguage);
    return setLocale(context, lang);
}

    public static String getLanguage(Context context)
    return getPersistedData(context, Locale.getDefault());
}

    public static Context setLocale(Context context, String language) {
        persist(context, language);

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
            return updateResources(context, language);
        }

        return updateResourcesLegacy(context, language);
    }

    private static String getPersistedData(Context context, String defaultLanguage) {
        SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(context);
        return preferences.getString(SELECTED_LANGUAGE, defaultLanguage);
    }

    private static void persist(Context context, String language) {
        SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(context);
        SharedPreferences.Editor editor = preferences.edit();

        editor.putString(SELECTED_LANGUAGE, language);
        editor.apply();
    }

    @TargetApi(Build.VERSION_CODES.N)
    private static Context updateResources(Context context, String language) {
        Locale locale = new Locale(language);
        Locale.setDefault(locale);

        Configuration configuration = context.getResources().getConfiguration();
        configuration.setLocale(locale);
        configuration.setLayoutDirection(locale);
    }

```

```

        return context.createConfigurationContext(conf
    }

    @SuppressWarnings("deprecation")
    private static Context updateResourcesLegacy(Conte
language) {
        Locale locale = new Locale(language);
        Locale.setDefault(locale);

        Resources resources = context.getResources();

        Configuration configuration = resources.getCon
configuration.locale = locale;
        if (Build.VERSION.SDK_INT >= Build.VERSION_COD
            configuration.setLayoutDirection(locale);
        }

        resources.updateConfiguration(configuration,
resources.getDisplayMetrics());

        return context;
    }
}

```

You need to override `attachBaseContext` and call `LocaleHelper.onAttach()` to initialize the locale settings in your application.

```

import android.app.Application;
import android.content.Context;

import com.gunhansancar.changelanguageexample.helper.L

public class MainApplication extends Application {
    @Override
    protected void attachBaseContext(Context base) {
        super.attachBaseContext(LocaleHelper.onAttach(
    }
}

```

All you have to do is to add

```
LocaleHelper.onCreate(this, "en");
```

wherever you want to change the locale.

Share Improve this answer

edited Aug 8, 2017 at 3:55

Follow

answered Feb 2, 2016 at 9:12

EVERYTHING IS
UNDER CTRL

[Anirudh Sharma](#)

7,964 ● 13 ● 42 ● 44

LocaleHelper is a class from the article. Any links have a risk of being taken down. Please add the code into your answer.

– [Zoe - Save the data dump](#) ♦ Jul 19, 2017 at 15:47

I dont want to restart my app becasue app is doing some task like recording screen . so without restarting app is there any solution for Android 7.0 – [PriyankaChauhan](#) Oct 6, 2017 at 9:51

-
- 1 @PriyankaChauhan I think that the article covers this case: You have two options to update currently visible layout: **First**, you can just update the text or any other language dependent resources one by one. – [Maksim Turaev](#) Oct 10, 2017 at 12:05 ✎

thanks for adding the new `createConfigurationContext` , that was helpful – [jacoballenwood](#) Jan 16, 2019 at 20:00

-
- 2 onCreate or onAttach to call ? – [vanste25](#) Mar 2, 2019 at 22:37
-



13

Mayuri's answer is correct but that will only work on Api 33 or above. Here is the step by step backward compatible solution :



Step 1: create locales_config.xml under res/xml folder.



```
//res/xml/locales_config.xml
<?xml version="1.0" encoding="utf-8"?>
<locale-config xmlns:android="http://schemas.android.c
    <!-- Add your required languages -->
    <locale android:name="hi" />
    <locale android:name="en" />
</locale-config>
```

Step 2 : Add localeConfig in Manifest under Application

```
<manifest>
<application
    android:localeConfig="@xml/locales_config">
</application>
```

Step 3 : Add this service in Manifest

```
<service
    android:name="androidx.appcompat.app.AppLocalesMetadataHolderService"
    android:enabled="false"
    android:exported="false">
    <meta-data
        android:name="autoStoreLocales"
        android:value="true" />
</service>
```

Step 4 : specify the same languages using the resConfigs property in your app's module-level build.gradle file:

```
android {  
    defaultConfig {  
        ...  
        resConfigs "hi", "en"  
    }  
}
```

(it requires appCompat version 1.6.0 or higher)

```
implementation 'androidx.appcompat:appcompat:1.6.0'
```

Step 5 : Now you can use below code to change app language (tested on android 9,10,12 & 13)

```
LocaleListCompat appLocale = LocaleListCompat.forLanguageTags(  
    user selected language code  
)  
AppCompatActivity.setApplicationLocales(appLocale);
```

Share Improve this answer

edited Feb 24, 2023 at 7:30

Follow

answered Jan 25, 2023 at 12:44



PRANAV SINGH

1,190 ● 13 ● 22

The google docs:

developer.android.com/guide/topics/resources/app-languages , and the sample app : github.com/android/user-interface-samples/tree/main/... – JeckOnly Jan 25, 2023 at

14:39 ✎

- 3 The problem with this is that `setApplicationLocales` recreates the activity in a way that we get a black screen for

a tiny moment. – [Starwave](#) Mar 22, 2023 at 6:15

when adding localeconfig to manifest, it says: Attribute localeConfig is only used in API level 33 and higher (current min is 28). So what about backward compatibility, if it is ignored for lower versions? And if you set Autostorelocales to true, then you don't need manual config xml file

– [Darksymphony](#) Mar 18 at 18:55 

After spending 2 days, I finally found the right answer. Thank you for providing it. – [jojo](#) Apr 9 at 7:14



Time for a due update.

12

First off, the deprecated list with the API in which it was deprecated:



- `configuration.locale` (API 17)
- `updateConfiguration(configuration, displaymetrics)` (API 17)



The thing no question answered recently has gotten right is the usage of the new method.

`createConfigurationContext` is the new method for `updateConfiguration`.

Some have used it standalone like this:

```
Configuration overrideConfiguration = ctx.getResources
Locale locale = new Locale("en_US");
overrideConfiguration.setLocale(locale);
createConfigurationContext(overrideConfiguration);
```

... but that doesn't work. Why? The method returns a context, which then is used to handle Strings.xml translations and other localized resources (images, layouts, whatever).

The proper usage is like this:

```
Configuration overrideConfiguration = ctx.getResources()
Locale locale = new Locale("en_US");
overrideConfiguration.setLocale(locale);
//the configuration can be used for other stuff as well
Context context = createConfigurationContext(overrideConfiguration)
Resources resources = context.getResources();
```

If you just copy-pasted that into your IDE, you may see a warning that the API requires you targeting API 17 or above. This can be worked around by putting it in a method and adding the annotation `@TargetApi(17)`

But wait. What about the older API's?

You need to create another method using `updateConfiguration` without the `TargetApi` annotation.

```
Resources res = YourApplication.getInstance().getResources()
// Change locale settings in the app.
DisplayMetrics dm = res.getDisplayMetrics();
android.content.res.Configuration conf = res.getConfiguration()
conf.locale = new Locale("th");
res.updateConfiguration(conf, dm);
```

You don't need to return a context here.

Now, managing these can be difficult. In API 17+ you need the context created (or the resources from the context created) to get the appropriate resources based on localization. How do you handle this?

Well, this is the way I do it:

```
/**
 * Full locale list: https://stackoverflow.com/question/24915167/list-of-supported-languages-locales-on-android
 * @param lang language code (e.g. en_US)
 * @return the context
 * PLEASE READ: This method can be changed for usage of Context
 Simply add a Context to the arguments
 */
public Context setLanguage(String lang/*, Context c*/)
    Context c = AndroidLauncher.this; //remove if the c
passed. This is a utility line, can be removed totally
with the activity (if argument Context isn't passed)
    int API = Build.VERSION.SDK_INT;
    if(API >= 17){
        return setLanguage17(lang, c);
    }else{
        return setLanguageLegacy(lang, c);
    }
}

/**
 * Set language for API 17
 * @param lang
 * @param c
 * @return
 */
@TargetApi(17)
public Context setLanguage17(String lang, Context c){
    Configuration overrideConfiguration = c.getResources().getConfiguration();
    Locale locale = new Locale(lang);
    Locale.setDefault(locale);
    overrideConfiguration.setLocale(locale);
    //the configuration can be used for other stuff as
    Context context =
```

```

createConfigurationContext(overrideConfiguration);// "l
redundant" if the below line is uncommented, it is nee
    //Resources resources = context.getResources();//I
resources instead of a Context, uncomment this line an
    return context;
}

public Context setLanguageLegacy(String lang, Context
    Resources res = c.getResources();
    // Change locale settings in the app.
    DisplayMetrics dm = res.getDisplayMetrics();//Util
    android.content.res.Configuration conf = res.getCo

    conf.locale = new Locale(lang);//setLocale require
createConfigurationContext
    Locale.setDefault(conf.locale);
    res.updateConfiguration(conf, dm);

    //Using this method you don't need to modify the C
at the start of the app is enough. As you
    //target both API's though, you want to return the
clue what is called. Now you can use the Context
    //supplied for both things
    return c;
}

```

This code works by having one method that makes calls to the appropriate method based on what API. This is something I have done with a lot of different deprecated calls (including `Html.fromHtml`). You have one method that takes in the arguments needed, which then splits it into one of two (or three or more) methods and returns the appropriate result based on API level. It is flexible as you don't have to check multiple times, the "entry" method does it for you. The entry-method here is `setLanguage`

PLEASE READ THIS BEFORE USING IT

You need to use the Context returned when you get resources. Why? I have seen other answers here who use `createConfigurationContext` and doesn't use the context it returns. To get it to work like that, `updateConfiguration` has to be called. Which is deprecated. Use the context returned by the method to get resources.

Example usage:

Constructor or somewhere similar:

```
ctx = getLanguage(lang); // lang is loaded or generated.  
lang is not something this answer handles (nor will ha
```

And then, wherever you want to get resources you do:

```
String fromResources = ctx.getString(R.string.hellowor
```

Using any other context will (in theory) break this.

AFAIK you still have to use an activity context to show dialogs or Toasts. for that you can use an instance of an activity (if you are outside)

And finally, use `recreate()` on the activity to refresh the content. Shortcut to not have to create an intent to

refresh.

Share Improve this answer

edited Jul 19, 2017 at 17:51

Follow

answered Jul 19, 2017 at 17:08



Zoe - Save the data
dump ♦

28.1k ● 22 ● 127 ● 158

-
- 1 Some may wonder if the created context will cost your memory. However according to Android Official Documentation: "Each call to this method returns a new instance of a Context object; Context objects are not shared, however common state (ClassLoader, other Resources for the same configuration) may be so the Context itself can be fairly lightweight." So I think Android does expect you to use a separate context object for locale things. – [Sira Lam](#) Sep 18, 2017 at 9:22
-



10



**For Android 7.0 Nougat (and lower)
follow this article:**

[Change Language Programatically in Android](#)

Old answer

This include RTL/LTR support:

```
public static void changeLocale(Context context, Locale
    Configuration conf = context.getResources().getCon
    conf.locale = locale;
    Locale.setDefault(locale);
```

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.J
    conf.setLayoutDirection(conf.locale);
}

context.getResources().updateConfiguration(conf,
context.getResources().getDisplayMetrics());
}
```

Share Improve this answer

edited Feb 14, 2017 at 14:13

Follow

answered Aug 31, 2016 at 11:22



Duda

1,673 ● 17 ● 23

-
- 1 updateConfiguration is deprecated. The link is useful, please add it into your answer. (Link only answers are not good, as the link may be taken down. If that happens, this answer is useless) – [Zoe - Save the data dump](#) ♦ Jul 19, 2017 at 15:46
-



If you write

9

```
android:configChanges="locale"
```



In every activity (in the manifest file) then no need to set it every time you enter `Activity`.



Share Improve this answer

edited Jun 8, 2019 at 4:22

Follow



Ricardo A.

688 ● 2 ● 8 ● 35

answered Jul 20, 2010 at 12:50



Brijesh Masrani

1,449 ● 3 ● 16 ● 27

13 If it's in the manifest then how does this constitute a change at runtime, which appeared to be what the O.P. wanted?

– [user316117](#) Nov 1, 2012 at 16:14

1 @user316117 It indicates to Android that the app will handle all matters regarding locale configuration internally, not that the locale is static. I'm not sure if that would prevent Android from setting the locale when changing between Activities, though, as I've only seen `configChanges` used for a hack to preserve Activity state on rotations/etc. – [JAB](#) Apr 9, 2014 at 12:35 ✎

how to set the language only to english specific?

– [Kaveesh Kanwal](#) Jun 2, 2016 at 9:16 ✎

1 ... until Android kills your activity because it needs more RAM – [Louis CAD](#) Aug 29, 2016 at 8:13

@Brijesh If we have change the app language then if we have some searching option within app, and if we search in that then, how app will show data, should we develop some different database for each language or some android code setting is there so that app could show data according to search ? – [Vishwa Pratap](#) Sep 16, 2019 at 8:39



8

The only solution that fully works for me is a combination of Alex Volovoy's code with application restart mechanism:



```
void restartApplication() {  
    Intent i = new Intent(MainTabActivity.context, Mag  
        i.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);  
        i.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);  
}
```




```
MainTabActivity.context.startActivity(i);  
}  
  
/** This activity shows nothing; instead, it restarts  
public class MagicAppRestart extends Activity {  
    @Override  
    protected void onActivityResult(int requestCode, i  
data) {  
        super.onActivityResult(requestCode, resultCode  
finish());  
    }  
  
    protected void onResume() {  
        super.onResume();  
        startActivityForResult(new Intent(this, MainTa  
    }  
}
```

Share Improve this answer

Follow

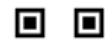
edited Mar 27, 2016 at 22:59



Matias Elorriaga

9,140 ● 5 ● 43 ● 60

answered Apr 29, 2013 at 8:04



Misha

5,380 ● 6 ● 39 ● 63

2 after locale change you can also call
`activity.recreate()` – To Kra Mar 30, 2015 at 9:47

1 I dont want to restart my app becasue app is doing some task
like recording screen . so without restarting app is there any
solution for Android 7.0 – PriyankaChauhan Oct 6, 2017 at
9:52



I was facing the same issue. On GitHub I found the
[Android-LocalizationActivity library](#).

8



This library makes it very simple to change the language of your app at runtime, as you can see in the code sample below. A sample project including the sample code below and more information can be found at the [github page](#).

The LocalizationActivity extends AppCompatActivity, so you can also use it when you are using Fragments.

```
public class MainActivity extends LocalizationActivity
    View.OnClickListener {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_simple);

        findViewById(R.id.btn_th).setOnClickListener(t
        findViewById(R.id.btn_en).setOnClickListener(t
    }

    @Override
    public void onClick(View v) {
        int id = v.getId();
        if (id == R.id.btn_en) {
            setLanguage("en");
        } else if (id == R.id.btn_th) {
            setLanguage("th");
        }
    }
}
```

Share Improve this answer

answered Apr 15, 2016 at 22:27

Follow



Rockney

10.6k ● 3 ● 22 ● 27



You can ask the user to select the language in first screen and save it in `SharedPreferences`

8



```
SharedPreferences.Editor editor = getSharedPreferences(MODE_PRIVATE).edit();
editor.putString("lang", "si");
editor.apply();

recreate();
```



Then you can take it in every `Activity` in your application. Here I have set English and Sinhala languages.

```
@Override
protected void attachBaseContext(Context base) {
    SharedPreferences prefs = base.getSharedPreferences();
    String restoredText = prefs.getString("lang", "No");

    if (restoredText.equals("si")){
        super.attachBaseContext(LocaleHelper.localeUpd
"si"));
    }else{
        super.attachBaseContext(LocaleHelper.localeUpd
"en"));
    }
}
```

And this is your `localUpdateResources` method. Place it in `LocaleHelper` class

```
public class LocaleHelper {
    public static Context localeUpdateResources(Context
languageCode) {

        Context newContext = context;
```

```

Locale locale = new Locale(languageCode);
Locale.setDefault(locale);

Resources resources = context.getResources();
Configuration config = new Configuration(resou

if (Build.VERSION.SDK_INT >= Build.VERSION_COD

    config.setLocale(locale);
    newContext = context.createConfigurationCo

} else {

    config.locale = locale;
    resources.updateConfiguration(config,
resources.getDisplayMetrics());
}

return newContext;

}
}

```

Share Improve this answer

edited Jul 26, 2021 at 4:35

Follow

answered Jun 21, 2021 at 5:18



Lakpriya Senevirathna

5,051 ● 3 ● 26 ● 44



`Resources.updateConfiguration()` has been **deprecated** and I have resolved this **without** creating any custom

8

`ContextWrapper` .



First I created an extension function



```
fun Context.setAppLocale(language: String): Context {  
    val locale = Locale(language)  
    Locale.setDefault(locale)  
    val config = resources.configuration  
    config.setLocale(locale)  
    config.setLayoutDirection(locale)  
    return createConfigurationContext(config)  
}
```

Then in the activity's `attachBaseContext` method, simply replacing the context with the new one.

```
override fun attachBaseContext(newBase: Context) {  
    super.attachBaseContext(ContextWrapper(newBase.setAp  
})
```

Share Improve this answer

edited Aug 21, 2021 at 13:28

Follow

answered Jun 26, 2021 at 4:59



S Haque

7,261 ● 5 ● 32 ● 38

1 In that case you have to call the `recreate()` method of the activity when a language is being selected from dropdown.
– [S Haque](#) Jul 26, 2021 at 11:37 ✎

1 Thanks for pointing it out. I'd already done that and it's working fine. – [Feroz Khan](#) Jul 26, 2021 at 11:41

1 How do you update locale for application context? It's working for me for activity contexts, but some strings (that are using resources from application context) are still not translated. – [Micer](#) Sep 23, 2021 at 21:07

2 Perfect work this code :) – [NAP-Developer](#) Feb 6, 2022 at 13:28

1 @Micer you can do it if you override application on create and change locale also there as shown in some other examples... – [Renetik](#) May 7, 2022 at 20:59



8

The support to Per-app language preferences was just added to **API 33 (Android 13, Tiramisu)** currently on Developer Preview).



To change the app's locale just call [setApplicationLocales](#) from **LocaleManager**:



```
// Set app locale to pt-BR (Portuguese, Brazil)
getSystemService(LocaleManager::class.java)
    .applicationLocales = LocaleList(Locale.forLanguageTag("pt-BR"))
```

See more at

<https://developer.android.com/about/versions/13/features/app-languages#api-impl>

I've wrote an article about this feature

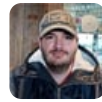
<https://proandroiddev.com/exploring-the-new-android-13-per-app-language-preferences-8d99b971b578>

Share Improve this answer

edited Apr 20, 2022 at 12:29

Follow

answered Feb 17, 2022 at 2:15



Heitor Paceli

530 ● 7 ● 19

-
- 1 @Renetik [setApplicationLocales backward compatibility](#)
– Reza Mohammadi May 8, 2022 at 23:48
-



8

This feature is officially launched by Google for Android 13 (and has backward support too). Android now lets you choose language per app.



Official documentation here -

<https://developer.android.com/guide/topics/resources/app-languages>



To set a user's preferred language, you would ask the user to select a locale in the language picker, then set that value in the system:

```
// 1. Inside an activity, in-app language picker gets
// 2. App calls the API to set its locale
mContext.getSystemService(LocaleManager.class
    ).setApplicationLocales(newLocaleList(Locale.forLa
// 3. The system updates the locale and restarts the a
configuration updates
// 4. The app is now displayed in "xx-YY" language
```

To get a user's current preferred language to display in the language picker, your app can get the value back from the system:

```
// 1. App calls the API to get the preferred locale
LocaleList currentAppLocales =
```

```
mContext.getSystemService(LocaleManager.class).get
// 2. App uses the returned LocaleList to display lang
```

Share Improve this answer

answered Aug 23, 2022 at 2:59

Follow



[Mayuri Khinvasara](#)

1,517 ● 1 ● 16 ● 13

Context.getSystemService(LocaleManager.class).setApplicationLocales(newLocaleList(Locale.forLanguageTag("xx-YY"))); This code wont work on below android 14.

– [Ajay Prajapati](#) Aug 13 at 16:55



6



For Arabic/RTL support

1. You must update your language settings through - attachBaseContext()
2. For android version N and above you must use createConfigurationContext() & updateConfiguration() - else RTL layout not working properly

```
@Override
protected void attachBaseContext(Context newBase) {
    super.attachBaseContext(updateBaseContextLocale(newBase));
}

public Context updateBaseContextLocale(Context c) {
    String language = SharedPreference.getInstance(c)
        .getString("lan");//it return "en", "ar" like this
    if (language == null || language.isEmpty())
        //when first time enter into app (get the default language)
        language = Locale.getDefault().getLanguage();
    return createConfigurationContext().updateConfiguration(
        new Configuration.Builder().setLocale(Locale.forLanguageTag(language)).build(),
        c);
}
```



```

        if (language.equals("ar")) {
            SharedPreferences.getInstance().save()
        }
    }
    Locale locale = new Locale(language);
    Locale.setDefault(locale);

    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
        updateResourcesLocale(context, locale);
        return updateResourcesLocaleLegacy(context, locale);
    }

    return updateResourcesLocaleLegacy(context, locale);
}

@TargetApi(Build.VERSION_CODES.N)
private Context updateResourcesLocale(Context context) {
    Configuration configuration =
context.getResources().getConfiguration();
    configuration.setLocale(locale);
    return context.createConfigurationContext(configuration);
}

@SuppressWarnings("deprecation")
private Context updateResourcesLocaleLegacy(Context context) {
    Resources resources = context.getResources();
    Configuration configuration = resources.getConfiguration();
    configuration.locale = locale;
    resources.updateConfiguration(configuration,
resources.getDisplayMetrics());
    return context;
}

```



Run code snippet

[Expand snippet](#)

Share Improve this answer

answered May 18, 2020 at 8:23

Follow



Ranjithkumar

18.3k ● 16 ● 131 ● 170

3 This should be the correct answer. Just test it and works (22/02/2021). Thank you my friend. – [Ramiro G.M.](#) Feb 22, 2021 at 21:24



At first create multi string.xml for different languages; then use this block of code in `onCreate()` method:

5



```
super.onCreate(savedInstanceState);
String languageToLoad = "fr"; // change your language
Locale locale = new Locale(languageToLoad);
Locale.setDefault(locale);
Configuration config = new Configuration();
config.locale = locale;
getBaseContext().getResources().updateConfiguration(config,
getBaseContext().getResources().getDisplayMetrics())
this setContentView(R.layout.main);
```

Share Improve this answer

edited Feb 9, 2017 at 15:10

Follow



[Tonechas](#)

13.7k ● 16 ● 50 ● 83

answered Feb 6, 2017 at 10:20



[Mohsen mokhtari](#)

3,032 ● 1 ● 32 ● 41

Thank you, this code works great, I tested on Android 5.x and 6.x without any problems – [innovaciones](#) May 3, 2017 at 19:25



```
Locale locale = new Locale("en");
Locale.setDefault(locale);
```

5



```
Configuration config = context.getResources().getConfi  
config.setLocale(locale);  
context.createConfigurationContext(config);
```

Important update:

```
context.getResources().updateConfiguration(config,  
context.getResources().getDisplayMetrics());
```

Note, that on SDK ≥ 21 , you need to call *'Resources.updateConfiguration()'*, otherwise resources will not be updated.

Share Improve this answer

answered May 12, 2017 at 13:07

Follow



[Maxim Petlyuk](#)

1,154 ● 15 ● 21

updateConfiguration is deprecated. AFAIK you use createConfigurationContext and apply the context you have to it (`Context ctx = createConfigurationContext(args);` and get resources from that – [Zoe - Save the data dump](#) ♦ Jul 19, 2017 at 15:50



I know that it is deprecated. But anyway I don't know any solution which can work on android 5 and higher.

– [Maxim Petlyuk](#) Jul 19, 2017 at 17:44

Then you clearly didn't check the javadoc. you call the context created from createConfigurationContext

– [Zoe - Save the data dump](#) ♦ Jul 19, 2017 at 17:45

Ok, but anyway we should call updateConfiguration(), right?

– [Maxim Petlyuk](#) Jul 19, 2017 at 18:03

- 1 **Don't use the deprecated call.** Meaning no calling
updateConfiguration – [Zoe - Save the data dump](#) ♦ Jul 19,
2017 at 18:11
-



None of the solutions listed here helped me.

5



The language did not switch on android >= 7.0 if
AppCompatActivity.setDefaultNightMode(AppCompatActivity.MODE_NIGHT_YES)



This LocaleUtils works just fine:

<https://gist.github.com/GigigoGreenLabs/7d555c762ba2d3a810fe>



LocaleUtils

```
public class LocaleUtils {

    public static final String LAN_SPANISH      = "es";
    public static final String LAN_PORTUGUESE  = "pt";
    public static final String LAN_ENGLISH     = "en";

    private static Locale sLocale;

    public static void setLocale(Locale locale) {
        sLocale = locale;
        if(sLocale != null) {
            Locale.setDefault(sLocale);
        }
    }

    public static void updateConfig(ContextThemeWrapper wr
        if(sLocale != null && Build.VERSION.SDK_INT >=
        Build.VERSION_CODES.JELLY_BEAN_MR1) {
        Configuration configuration = new Configuratio
        configuration.setLocale(sLocale);
        wrapper.applyOverrideConfiguration(configurati
```

```

    }
}

public static void updateConfig(Application app, Configuration config) {
    if(sLocale != null && Build.VERSION.SDK_INT <
        Build.VERSION_CODES.JELLY_BEAN_MR1) {
        //Wrapping the configuration to avoid Activity
        Configuration newConfig = new Configuration(config);
        newConfig.locale = sLocale;
        Resources res = app.getBaseContext().getResources();
        res.updateConfiguration(newConfig, res.getDisplayMetrics());
    }
}
}

```

Added this code to Application

```

public class App extends Application {
    public void onCreate(){
        super.onCreate();

        LocaleUtils.setLocale(new Locale("iw"));
        LocaleUtils.updateConfig(this,
            getBaseContext().getResources().getConfiguration());
    }

    @Override
    public void onConfigurationChanged(Configuration newConfig) {
        super.onConfigurationChanged(newConfig);
        LocaleUtils.updateConfig(this, newConfig);
    }
}

```

Code in Activity

```

public class BaseActivity extends AppCompatActivity {
    public BaseActivity() {
        LocaleUtils.updateConfig(this);
    }
}

```

```
}  
}
```

Share Improve this answer

edited Jul 20, 2019 at 16:37

Follow

answered Jul 1, 2019 at 21:50



Pavel Shirokov

462 ● 4 ● 6

This works, thank you. Also, it doesn't have anything to do with Calligraphy library while changing the font which is so great. – [moḥsen](#) 🌟 Jul 8, 2020 at 6:11



4



```
/*change language at Run-time*/  
//use method like that:  
//setLocale("en");  
public void setLocale(String lang) {  
    myLocale = new Locale(lang);  
    Resources res = getResources();  
    DisplayMetrics dm = res.getDisplayMetrics();  
    Configuration conf = res.getConfiguration();  
    conf.locale = myLocale;  
    res.updateConfiguration(conf, dm);  
    Intent refresh = new Intent(this, AndroidLocalize.cl  
    startActivity(refresh);  
}
```

Share Improve this answer

answered Feb 25, 2015 at 15:27

Follow



altan yuksel

183 ● 1 ● 6

6 no need to start new activity, just refresh actual
`activity.recreate()` – [To Kra](#) Mar 30, 2015 at 9:50



`Locale` `configuration` should be set in each `activity`
before setting the content -

4

```
this setContentView(R.layout.main);
```



Share Improve this answer

edited Aug 3, 2015 at 5:30

Follow



[Anil Meenugu](#)

1,431 ● 2 ● 11 ● 16



answered Nov 13, 2012 at 14:48



[cheskapac](#)

51 ● 1 ● 4

2 But what if you want to toggle it on the fly, after
`setContentView()` has been called? – [IgorGanapolsky](#) ★ Jun
17, 2014 at 16:14

2 after locale change you can also call
`activity.recreate()` – [To Kra](#) Mar 30, 2015 at 9:48 ✎



Here is some code that works for me:

4

```
public class MainActivity extends AppCompatActivity {  
    public static String storeLang;
```



```
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState)
```



```
        SharedPreferences shp =  
        PreferenceManager.getDefaultSharedPreferences(this);
```



```
storeLang = shp.getString(getString(R.string.k

// Create a new Locale object
Locale locale = new Locale(storeLang);

// Create a new configuration object
Configuration config = new Configuration();
// Set the locale of the new configuration
config.locale = locale;
// Update the configuration of the Application
getResources().updateConfiguration(
    config,
    getResources().getDisplayMetrics()
);

super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
}
```

Source: [here](#)

Share Improve this answer

Follow

edited Jan 12, 2019 at 10:47



PaulCK

1,268 ● 3 ● 20 ● 29

answered Jul 29, 2018 at 12:55



Til Schweiger

103 ● 7



Things changed and today, relevant solution comes from:
[Android site](#).

4



There are many ways, but the simplest solution is to
implement it using the AndroidX support library:

1. add to the AndroidManifest.xml:



```
<service
    android:name="androidx.appcompat.app.AppLocal
    android:enabled="false"
    android:exported="false">
    <meta-data
        android:name="autoStoreLocales"
        android:value="true" />
</service>
```

2. in the build.gradle

```
dependencies { implementation
```

```
'androidx.appcompat:appcompat:1.6.1'
```

3. Add the code where you want to change the locale

```
SharedPreferences preferences = getSharedPreferen
Context.MODE_PRIVATE);
String langCode = preferences.getString("Lang_cod

LocaleListCompat appLocale = LocaleListCompat.for
or use "xx-YY"
AppCompatActivity.setApplicationLocales(appLocale
```

Beware: calling `setApplicationLocales()` recreates your Activity

Share Improve this answer

answered Sep 14, 2023 at 7:57

Follow



Lotfi

721 ● 1 ● 6 ● 23

Thx. This works for me. Im using android Upside down cake.

– [Michał Schmude](#) Nov 12, 2023 at 12:53



I finally figured out how to setup it to work on both $\geq N$ android versions.

2



Extend AppCompatActivity with your own abstract class, like:



```
abstract class MAppCompatActivity : AppCompatActivity {
    override fun attachBaseContext(newBase: Context?) {
        super.attachBaseContext(LocaleHelper.wrap(newBase))
    }

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        if (Build.VERSION.SDK_INT < Build.VERSION_CODES.N)
            LocaleHelper.wrap(this)
    }
}
```

attachBaseContext is called on Android $\geq N$ versions and on this way activity will use the correct context. On Android $< N$, we have to call this function on one other way, before setting content view. Therefore we override onCreate function to set correct context. Means, whenever you create a new Activity you have to extend your abstract class. Like this one:

```
class TermsActivity : MAppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_terms)
    }
}
```

And finally the LocaleHelper is like this:

```

import android.annotation.TargetApi;
import android.content.Context;
import android.content.ContextWrapper;
import android.content.SharedPreferences;
import android.content.res.Configuration;
import android.content.res.Resources;
import android.os.Build;
import android.util.DisplayMetrics;

import com.at_zone.constants.SharedPreferencesKeys;

import java.util.Locale;

public class LocaleHelper extends ContextWrapper {

    public LocaleHelper(Context base) {
        super(base);
    }

    public static Context wrap(Context context) {
        SharedPreferences sharedPreferences = context.
            SharedPreferencesKeys.SHARED_PREFERENC
        );
        String language =
sharedPreferences.getString(SharedPreferencesKeys.CURR
        if (!language.equals("default")) {
            Configuration config = context.getResource
            if (!language.equals("")) {
                Locale locale = new Locale(language);
                Locale.setDefault(locale);
                if (Build.VERSION.SDK_INT >= Build.VER
                    setSystemLocale(config, locale);
                } else {
                    setSystemLocaleLegacy(context, con
                }
                config.setLayoutDirection(locale);
                context = context.createConfigurationC
            }
            return new LocaleHelper(context);
        }
        return context;
    }
}

```

```

    public static String getSystemLanguage(Context context) {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
            return getSystemLocale(context).getLanguage();
        } else {
            return getSystemLocaleLegacy(context).getLanguage();
        }
    }

    public static Locale getSystemLocaleLegacy(Context context) {
        Configuration config = context.getResources().getConfiguration();
        return config.locale;
    }

    @TargetApi(Build.VERSION_CODES.N)
    public static Locale getSystemLocale(Context context) {
        return context.getResources().getConfiguration().getLocales().get(0);
    }

    public static void setSystemLocaleLegacy(Context context, Locale locale) {
        Configuration config = context.getResources().getConfiguration();
        config.locale = locale;
        Resources res = context.getResources();
        DisplayMetrics dm = res.getDisplayMetrics();
        res.updateConfiguration(config, dm);
    }

    @TargetApi(Build.VERSION_CODES.N)
    public static void setSystemLocale(Context context, Locale locale) {
        Configuration config = context.getResources().getConfiguration();
        config.setLocale(locale);
    }
}

```

Share Improve this answer

answered Aug 18, 2020 at 8:24

Follow



RuSsCiTy

466 ● 1 ● 3 ● 11



There are some steps that you should implement

1

First, you need to change the locale of your configuration



```
Resources resources = context.getResources();

Configuration configuration = resources.getConfiguration();
configuration.locale = new Locale(language);

resources.updateConfiguration(configuration, resources
```

Second, if you want your changes to apply directly to the layout that is visible, you either can update the views directly or you can just call `activity.recreate()` to restart the current activity.

And also you have to persist your changes because after user closes your application then you would lose the language change.

I explained more detailed solution on my blog post [Change Language Programmatically in Android](#)

Basically, you just call `LocaleHelper.onCreate()` on your application class and if you want to change locale on the fly you can call `LocaleHelper.setLocale()`

Share Improve this answer

answered Oct 8, 2015 at 12:56

Follow



Gunhan

7,035 ● 3 ● 45 ● 38

@LunarWatcher Yes if you actually check the code on github or gist, it is already handled. – [Gunhan](#) Jul 19, 2017 at 16:20



This is working when i press button to change text language of my TextView.(strings.xml in values-de folder)

1



```
String languageToLoad = "de"; // your language
Configuration config = getBaseContext().getResources()
Locale locale = new Locale(languageToLoad);
Locale.setDefault(locale);
config.locale = locale;
getBaseContext().getResources().updateConfiguration(co
getBaseContext().getResources().getDisplayMetrics());
recreate();
```



Share Improve this answer

answered Apr 3, 2017 at 12:15

Follow



ashishdhiman2007

817 ● 2 ● 13 ● 29

1

2

Next



Highly active question. Earn 10 reputation (not counting the [association bonus](#)) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.