# 1、kotlin的基本语法

1.1.1定义函数：

a.带两个int参数返回int值的函数

fun sum(a: Int, b: Int): Int {

return a + b

}

fun main(args: Array<String>) {

print("sum of 3 and 5 is ")

println(sum (3, 5))

}

b.将表达式作为函数体、返回值类型自动推断的函数：

fun sum(a: Int, b: Int) = a + b

fun main(args:Array<String>) {

println("sum of 19 and 23 is ${sum(19, 23)}")

}

# 2、kotlin类的创建和使用

# 3、kotlin调用java代码

# 4、在Android中的具体使用

## 4.1 activity

## 4.2 常见的view

### 4.2.1 ViewPager

viewpager的使用

activity代码：

package com.hs.myfirstkotlin

import android.content.Intent

import android.os.Bundle

import android.os.Handler

import android.os.Message

import android.support.v7.app.AppCompatActivity

import android.util.Log

import android.view.View

import com.hs.myfirstkotlin.adapter.MyViewPagwerAdapter

// 使用来自主代码集的 R.layout.activity\_main

import kotlinx.android.synthetic.main.activity\_main.\*

class MainActivity : AppCompatActivity() {

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

testSetText("大家好", "我叫Kotlin")

//轮播图

myViewPager.adapter = MyViewPagwerAdapter(this, 4)

myViewPager.currentItem = 0

startPlay()

}

var handler = object : Handler() {

override fun handleMessage(msg: Message?) {

super.handleMessage(msg)

myViewPager.currentItem = msg!!.what

}

}

fun startPlay() {

Thread(Runnable {

//保持代码同步

synchronized(this) {

for (i in 1..4000) {// for循环从1到4000

try {

Thread.sleep(3000)//休眠3秒

handler.sendEmptyMessage(i)

} catch (e: InterruptedException) {

e.printStackTrace()

}

}

}

}).start()

}

/\*\*

\* activity的跳转

\*/

fun skip2Activity(view: View) {

//获取intent对象

val intent = Intent()

// 获取class是使用 :: 反射

intent.setClass(this, TestActivity::class.java)

//正常的java代码

startActivity(intent)

Log.d("123456","12346546")

}

fun testSetText(str: String?) {

mytext.text = str

}

fun testSetText(str: String?, name: String?) {

mytext.setText(name)

}

}

adapter的代码：

package com.hs.myfirstkotlin.adapter

import android.content.Context

import android.support.v4.view.PagerAdapter

import android.view.View

import android.view.ViewGroup

import android.widget.ImageView

import com.hs.myfirstkotlin.R

/\*\*

\* 作者：zhanghaitao on 2017/8/1 17:00

\* 邮箱：820159571@qq.com

\* viewPager的适配器（伪无限轮播的代码）

\*/

class MyViewPagwerAdapter(context: Context, pageCount: Int) : PagerAdapter() {

private var context: Context? = context

private var pageCount: Int = pageCount

override fun isViewFromObject(view: View?, `object`: Any?): Boolean {

return view == `object`

}

override fun getCount(): Int {

return pageCount \* 1000

}

override fun destroyItem(container: ViewGroup?, position: Int, `object`: Any?) {

container?.removeView(`object` as View)

}

override fun instantiateItem(container: ViewGroup?, position: Int): Any {

var imageView: ImageView? = ImageView(context)

var resId: Int

when (position % pageCount) {

0 -> {

resId = R.drawable.ic\_lunbo1

}

1 -> {

resId = R.drawable.ic\_lunbo2

}

2 -> {

resId = R.drawable.ic\_lunbo3

}

else -> {

resId = R.drawable.ic\_lunbo4

}

}

imageView?.setImageDrawable(container?.context?.resources?.getDrawable(resId))

var layoutParams: ViewGroup.LayoutParams = ViewGroup.LayoutParams(ViewGroup.LayoutParams.MATCH\_PARENT, ViewGroup.LayoutParams.MATCH\_PARENT)

imageView?.layoutParams = layoutParams

imageView?.scaleType = ImageView.ScaleType.CENTER\_CROP

container?.addView(imageView)

return imageView!!

}

}

XML的代码：

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/activity\_main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context="com.hs.myfirstkotlin.MainActivity">

<android.support.v4.view.ViewPager

android:id="@+id/myViewPager"

android:layout\_width="match\_parent"

android:layout\_height="200dp"/>

<TextView

android:id="@+id/mytext"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:onClick="skip2Activity"

android:text="Hello World!"/>

</LinearLayout>

## 4.3 service

## 4.4 provided

## 4.5 broadcast

# 五、进程间通讯

# 七、自定义View的开发

# 八、联网

# 九、图片处理

# 十、数据存储

# 十一、多媒体交互