

Please note that GitHub no longer supports your web browser.
We recommend upgrading to the latest [Google Chrome](#) or [Firefox](#).

Ignore


Learn more

Branch: master ▼

Find file

Copy path

AndroidUtilCode / utilcode / README-CN.md

 Blankj see 06/30 log
96a3d02 Jun 30, 2019

1 contributor

Raw

Blame

History

1147 lines (1021 sloc) 57 KB

Download

Gradle:

```
implementation 'com.blankj:utilcode:1.24.7'  
  
// if u use AndroidX, use the following  
implementation 'com.blankj:utilcodex:1.24.7'
```

APIs

- Activity 相关 -> [ActivityUtils.java](#) -> [Demo](#)

getActivityByView	: 根据视图获取 Activity
getActivityByContext	: 根据上下文获取 Activity
isActivityExists	: 判断 Activity 是否存在
startActivity	: 启动 Activity
startActivityForResult	: 启动 Activity 为返回结果
startActivities	: 启动多个 Activity
startHomeActivity	: 回到桌面
getActivityList	: 获取 Activity 栈链表
getLauncherActivity	: 获取启动项 Activity
getTopActivity	: 获取栈顶 Activity
isActivityAlive	: 判断 Activity 是否存活
isActivityExistsInStack	: 判断 Activity 是否存在栈中
finishActivity	: 结束 Activity
finishToActivity	: 结束到指定 Activity

`finishOtherActivities` : 结束所有其他类型的 Activity
`finishAllActivities` : 结束所有 Activity
`finishAllActivitiesExceptNewest`: 结束除最新之外的所有 Activity

• AdaptScreen 相关 -> [AdaptScreenUtils.java](#) -> [Demo](#)

`adaptWidth` : 适配宽度
`adaptHeight`: 适配高度
`closeAdapt` : 关闭适配 (pt 等同于 dp)
`pt2Px` : pt 转 px
`px2Pt` : px 转 pt

• App 相关 -> [AppUtils.java](#) -> [Demo](#)

`registerAppStatusChangeListener` : 注册 App 前后台切换监听器
`unregisterAppStatusChangeListener`: 注销 App 前后台切换监听器
`installApp` : 安装 App (支持 8.0)
`uninstallApp` : 卸载 App
`isAppInstalled` : 判断 App 是否安装
`isAppRoot` : 判断 App 是否有 root 权限
`isAppDebug` : 判断 App 是否是 Debug 版本
`isAppSystem` : 判断 App 是否是系统应用
`isAppForeground` : 判断 App 是否处于前台
`isAppRunning` : 判断 App 是否运行
`launchApp` : 打开 App
`relaunchApp` : 重启 App
`launchAppDetailsSettings` : 打开 App 具体设置
`exitApp` : 关闭应用
`getAppIcon` : 获取 App 图标
`getAppPackageName` : 获取 App 包名
`getAppName` : 获取 App 名称
`getAppPath` : 获取 App 路径
`getAppVersionName` : 获取 App 版本号
`getAppVersionCode` : 获取 App 版本码
`getAppSignature` : 获取 App 签名
`getAppSignatureSHA1` : 获取应用签名的的 SHA1 值
`getAppSignatureSHA256` : 获取应用签名的的 SHA256 值
`getAppSignatureMD5` : 获取应用签名的的 MD5 值
`getAppInfo` : 获取 App 信息
`getAppsInfo` : 获取所有已安装 App 信息
`getApkInfo` : 获取 Apk 信息

• 栏相关 -> [BarUtils.java](#) -> [Demo](#)

`getStatusBarHeight` : 获取状态栏高度 (px)
`setStatusBarVisibility` : 设置状态栏是否可见
`isStatusBarVisible` : 判断状态栏是否可见
`setStatusBarLightMode` : 设置状态栏是否为浅色模式

<code>addMarginTopEqualStatusBarHeight</code>	: 为 view 增加 <code>MarginTop</code> 为状态栏高度
<code>subtractMarginTopEqualStatusBarHeight</code>	: 为 view 减少 <code>MarginTop</code> 为状态栏高度
<code>setStatusBarColor</code>	: 设置状态栏颜色
<code>setStatusBarColor4Drawer</code>	: 为 <code>DrawerLayout</code> 设置状态栏颜色
<code>getActionBarHeight</code>	: 获取 <code>ActionBar</code> 高度
<code>setNotificationBarVisibility</code>	: 设置通知栏是否可见
<code>getNavBarHeight</code>	: 获取导航栏高度
<code>setNavBarVisibility</code>	: 设置导航栏是否可见
<code>isNavBarVisible</code>	: 判断导航栏是否可见
<code>setNavBarColor</code>	: 设置导航栏颜色
<code>getNavBarColor</code>	: 获取导航栏颜色
<code>isSupportNavBar</code>	: 判断是否支持导航栏

• 亮度相关 -> [BrightnessUtils.java](#) -> [Demo](#)

<code>isAutoBrightnessEnabled</code>	: 判断是否开启自动调节亮度
<code>setAutoBrightnessEnabled</code>	: 设置是否开启自动调节亮度
<code>getBrightness</code>	: 获取屏幕亮度
<code>setBrightness</code>	: 设置屏幕亮度
<code>setWindowBrightness</code>	: 设置窗口亮度
<code>getWindowBrightness</code>	: 获取窗口亮度

• Bus 相关 -> [BusUtils.java](#) -> [README](#)

`post`: 发送

• 磁盘缓存相关 -> [CacheDiskStaticUtils.java](#) -> [Test](#)

<code>setDefaultCacheDiskUtils</code>	: 设置默认磁盘缓存实例
<code>put</code>	: 缓存中写入数据
<code>getBytes</code>	: 缓存中读取字节数组
<code>getString</code>	: 缓存中读取 <code>String</code>
<code>getJSONObject</code>	: 缓存中读取 <code>JSONObject</code>
<code>getJSONArray</code>	: 缓存中读取 <code>JSONArray</code>
<code>getBitmap</code>	: 缓存中读取 <code>Bitmap</code>
<code>getDrawable</code>	: 缓存中读取 <code>Drawable</code>
<code>getParcelable</code>	: 缓存中读取 <code>Parcelable</code>
<code>getSerializable</code>	: 缓存中读取 <code>Serializable</code>
<code>getCacheSize</code>	: 获取缓存大小
<code>getCacheCount</code>	: 获取缓存个数
<code>remove</code>	: 根据键值移除缓存
<code>clear</code>	: 清除所有缓存

• 磁盘缓存相关 -> [CacheDiskUtils.java](#) -> [Test](#)

getInstance	: 获取缓存实例
Instance.put	: 缓存中写入数据
Instance.getBytes	: 缓存中读取字节数组
Instance.getString	: 缓存中读取 String
Instance.getJSONObject	: 缓存中读取 JSONObject
Instance.getJSONArray	: 缓存中读取 JSONArray
Instance.getBitmap	: 缓存中读取 Bitmap
Instance.getDrawable	: 缓存中读取 Drawable
Instance.getParcelable	: 缓存中读取 Parcelable
Instance.getSerializable	: 缓存中读取 Serializable
Instance.getCacheSize	: 获取缓存大小
Instance.getCacheCount	: 获取缓存个数
Instance.remove	: 根据键值移除缓存
Instance.clear	: 清除所有缓存

• 二级缓存相关 -> [CacheDoubleStaticUtils.java](#) -> [Test](#)

setDefaultCacheDoubleUtils	: 设置默认二级缓存实例
put	: 缓存中写入数据
getBytes	: 缓存中读取字节数组
getString	: 缓存中读取 String
getJSONObject	: 缓存中读取 JSONObject
getJSONArray	: 缓存中读取 JSONArray
getBitmap	: 缓存中读取 Bitmap
getDrawable	: 缓存中读取 Drawable
getParcelable	: 缓存中读取 Parcelable
getSerializable	: 缓存中读取 Serializable
getCacheDiskSize	: 获取磁盘缓存大小
getCacheDiskCount	: 获取磁盘缓存个数
getCacheMemoryCount	: 获取内存缓存个数
remove	: 根据键值移除缓存
clear	: 清除所有缓存

• 二级缓存相关 -> [CacheDoubleUtils.java](#) -> [Test](#)

getInstance	: 获取缓存实例
Instance.put	: 缓存中写入数据
Instance.getBytes	: 缓存中读取字节数组
Instance.getString	: 缓存中读取 String
Instance.getJSONObject	: 缓存中读取 JSONObject
Instance.getJSONArray	: 缓存中读取 JSONArray
Instance.getBitmap	: 缓存中读取 Bitmap
Instance.getDrawable	: 缓存中读取 Drawable
Instance.getParcelable	: 缓存中读取 Parcelable
Instance.getSerializable	: 缓存中读取 Serializable
Instance.getCacheDiskSize	: 获取磁盘缓存大小
Instance.getCacheDiskCount	: 获取磁盘缓存个数
Instance.getCacheMemoryCount	: 获取内存缓存个数

Instance.remove	: 根据键值移除缓存
Instance.clear	: 清除所有缓存

• 内存缓存相关 -> [CacheMemoryStaticUtils.java](#) -> [Test](#)

setDefaultCacheMemoryUtils	: 设置默认内存缓存实例
put	: 缓存中写入数据
get	: 缓存中读取字节数组
getCacheCount	: 获取缓存个数
remove	: 根据键值移除缓存
clear	: 清除所有缓存

• 内存缓存相关 -> [CacheMemoryUtils.java](#) -> [Test](#)

getInstance	: 获取缓存实例
Instance.put	: 缓存中写入数据
Instance.get	: 缓存中读取字节数组
Instance.getCacheCount	: 获取缓存个数
Instance.remove	: 根据键值移除缓存
Instance.clear	: 清除所有缓存

• 清除相关 -> [CleanUtils.java](#) -> [Demo](#)

cleanInternalCache	: 清除内部缓存
cleanInternalFiles	: 清除内部文件
cleanInternalDbs	: 清除内部数据库
cleanInternalDbByName	: 根据名称清除数据库
cleanInternalSp	: 清除内部 SP
cleanExternalCache	: 清除外部缓存
cleanCustomDir	: 清除自定义目录下的文件

• 点击相关 -> [ClickUtils.java](#) -> [Demo](#)

applyScale	: 应用点击缩放
applySingleDebounceing	: 对单视图应用防抖点击
applyGlobalDebounceing	: 对所有设置 GlobalDebounceing 的视图应用防抖点击
ClickUtils#OnDebounceingClickListener	: 防抖点击监听器
ClickUtils#OnMultiClickListener	: 连续点击监听器

• 克隆相关 -> [CloneUtils.java](#) -> [Test](#)

deepClone: 深度克隆

• 关闭相关 -> [CloseUtils.java](#)

closeIO : 关闭 IO
closeIOQuietly: 安静关闭 IO

• 颜色相关 -> [ColorUtils.java](#)

getColor : 获取颜色
setAlphaComponent: 设置颜色透明度值
setRedComponent : 设置颜色红色值
setGreenComponent: 设置颜色绿色值
setBlueComponent : 设置颜色蓝色值
string2Int : 颜色串转颜色值
int2RgbString : 颜色值转 RGB 串
int2ArgbString : 颜色值转 ARGB 串
getRandomColor : 获取随机色

• 转换相关 -> [ConvertUtils.java](#) -> [Test](#)

bytes2Bits, bits2Bytes	: bytes 与 bits 互转
bytes2Chars, chars2Bytes	: bytes 与 chars 互转
bytes2HexString, hexString2Bytes	: bytes 与 hexString 互转
memorySize2Byte, byte2MemorySize	: 以 unit 为单位的内存大小与字节数互转
byte2FitMemorySize	: 字节数转合适内存大小
timeSpan2Millis, millis2TimeSpan	: 以 unit 为单位的时间长度与毫秒时间戳互转
转	
millis2FitTimeSpan	: 毫秒时间戳转合适时间长度
input2OutputStream, output2InputStream	: inputStream 与 outputStream 互转
inputStream2Bytes, bytes2InputStream	: inputStream 与 bytes 互转
outputStream2Bytes, bytes2OutputStream	: outputStream 与 bytes 互转
inputStream2String, string2InputStream	: inputStream 与 string 按编码互转
outputStream2String, string2OutputStream	: outputStream 与 string 按编码互转
bitmap2Bytes, bytes2Bitmap	: bitmap 与 bytes 互转
drawable2Bitmap, bitmap2Drawable	: drawable 与 bitmap 互转
drawable2Bytes, bytes2Drawable	: drawable 与 bytes 互转
view2Bitmap	: view 转 Bitmap
dp2px, px2dp	: dp 与 px 互转
sp2px, px2sp	: sp 与 px 互转

• 崩溃相关 -> [CrashUtils.java](#)

init: 初始化

• 设备相关 -> [DeviceUtils.java](#) -> [Demo](#)

isDeviceRooted : 判断设备是否 rooted
 isAdbEnabled : 判断设备 ADB 是否可用
 getSDKVersionName: 获取设备系统版本号
 getSDKVersionCode: 获取设备系统版本码
 getAndroidID : 获取设备 AndroidID
 getMacAddress : 获取设备 MAC 地址
 getManufacturer : 获取设备厂商
 getModel : 获取设备型号
 getABIs : 获取设备 ABIs
 isTablet : 判断是否是平板
 isEmulator : 判断是否是模拟器

• 闪光灯相关 -> [FlashlightUtils.java](#) -> [Demo](#)

isFlashlightEnable : 判断设备是否支持闪光灯
 isFlashlightOn : 判断闪光灯是否打开
 setFlashlightStatus: 设置闪光灯状态
 destroy : 销毁

• 编码解码相关 -> [EncodeUtils.java](#) -> [Test](#)

urlEncode : URL 编码
 urlDecode : URL 解码
 base64Encode : Base64 编码
 base64Encode2String: Base64 编码
 base64Decode : Base64 解码
 htmlEncode : Html 编码
 htmlDecode : Html 解码

• 加密解密相关 -> [EncryptUtils.java](#) -> [Test](#)

encryptMD2, encryptMD2ToString	: MD2 加密
encryptMD5, encryptMD5ToString	: MD5 加密
encryptMD5File, encryptMD5File2String	: MD5 加密文件
encryptSHA1, encryptSHA1ToString	: SHA1 加密
encryptSHA224, encryptSHA224ToString	: SHA224 加密
encryptSHA256, encryptSHA256ToString	: SHA256 加密
encryptSHA384, encryptSHA384ToString	: SHA384 加密
encryptSHA512, encryptSHA512ToString	: SHA512 加密
encryptHmacMD5, encryptHmacMD5ToString	: HmacMD5 加密
encryptHmacSHA1, encryptHmacSHA1ToString	: HmacSHA1 加密
encryptHmacSHA224, encryptHmacSHA224ToString	: HmacSHA224 加密
encryptHmacSHA256, encryptHmacSHA256ToString	: HmacSHA256 加密
encryptHmacSHA384, encryptHmacSHA384ToString	: HmacSHA384 加密
encryptHmacSHA512, encryptHmacSHA512ToString	: HmacSHA512 加密
encryptDES, encryptDES2HexString, encryptDES2Base64	: DES 加密
decryptDES, decryptHexStringDES, decryptBase64DES	: DES 解密

encrypt3DES, encrypt3DES2HexString, encrypt3DES2Base64: 3DES 加密
 decrypt3DES, decryptHexString3DES, decryptBase64_3DES : 3DES 解密
 encryptAES, encryptAES2HexString, encryptAES2Base64 : AES 加密
 decryptAES, decryptHexStringAES, decryptBase64AES : AES 解密
 encryptRSA, encryptRSA2HexString, encryptRSA2Base64 : RSA 加密
 decryptRSA, decryptHexStringRSA, decryptBase64RSA : RSA 解密

• 文件相关 -> [FileIOUtils.java](#) -> [Test](#)

writeFileFromIS : 将输入流写入文件
 writeFileFromBytesByStream : 将字节数组写入文件
 writeFileFromBytesByChannel: 将字节数组写入文件
 writeFileFromBytesByMap : 将字节数组写入文件
 writeFileFromString : 将字符串写入文件
 readFile2List : 读取文件到字符串链表中
 readFile2String : 读取文件到字符串中
 readFile2BytesByStream : 读取文件到字节数组中
 readFile2BytesByChannel : 读取文件到字节数组中
 readFile2BytesByMap : 读取文件到字节数组中
 setBufferSize : 设置缓冲区尺寸

• 文件相关 -> [FileUtils.java](#) -> [Test](#)

getFileByPath : 根据文件路径获取文件
 isFileExists : 判断文件是否存在
 rename : 重命名文件
 isDir : 判断是否是目录
 isFile : 判断是否是文件
 createOrExistsDir : 判断目录是否存在, 不存在则判断是否创建成功
 createOrExistsFile : 判断文件是否存在, 不存在则判断是否创建成功
 createFileByDeleteOldFile : 判断文件是否存在, 存在则在创建之前删除
 copyDir : 复制目录
 copyFile : 复制文件
 moveDir : 移动目录
 moveFile : 移动文件
 delete : 删除文件或目录
 deleteDir : 删除目录
 deleteFile : 删除文件
 deleteAllInDir : 删除目录下所有东西
 deleteFilesInDir : 删除目录下所有文件
 deleteFilesInDirWithFilter: 删除目录下所有过滤的文件
 listFilesInDir : 获取目录下所有文件
 listFilesInDirWithFilter : 获取目录下所有过滤的文件
 getFileLastModified : 获取文件最后修改的毫秒时间戳
 getFileCharsetSimple : 简单获取文件编码格式
 getFileLines : 获取文件行数
 getDirSize : 获取目录大小
 getFileSize : 获取文件大小
 getDirLength : 获取目录长度
 getFileLength : 获取文件长度

getFileMD5	: 获取文件的 MD5 校验码
getFileMD5ToString	: 获取文件的 MD5 校验码
getDirName	: 根据全路径获取最长目录
getFileName	: 根据全路径获取文件名
getFileNameNoExtension	: 根据全路径获取文件名不带拓展名
getFileExtension	: 根据全路径获取文件拓展名

• Fragment 相关 -> [FragmentUtils.java](#) -> [Demo](#)

add	: 新增 fragment
show	: 显示 fragment
hide	: 隐藏 fragment
showHide	: 先显示后隐藏 fragment
replace	: 替换 fragment
pop	: 出栈 fragment
popTo	: 出栈到指定 fragment
popAll	: 出栈所有 fragment
remove	: 移除 fragment
removeTo	: 移除到指定 fragment
removeAll	: 移除所有 fragment
getTop	: 获取顶部 fragment
getTopInStack	: 获取栈中顶部 fragment
getTopShow	: 获取顶部可见 fragment
getTopShowInStack	: 获取栈中顶部可见 fragment
getFragments	: 获取同级别的 fragment
getFragmentsInStack	: 获取同级别栈中的 fragment
getAllFragments	: 获取所有 fragment
getAllFragmentsInStack	: 获取栈中所有 fragment
findFragment	: 查找 fragment
dispatchBackPressed	: 处理 fragment 回退键
setBackgroundColor	: 设置背景色
setBackgroundResource	: 设置背景资源
setBackground	: 设置背景

• Gson 相关 -> [GsonUtils.java](#) -> [Test](#)

getGson	: 获取 Gson 对象
toJson	: 对象转 Json 串
fromJson	: Json 串转对象
getListType	: 获取链表类型
getSetType	: 获取集合类型
getMapType	: 获取字典类型
getArrayType	: 获取数组类型
getType	: 获取类型

• 图片相关 -> [ImageUtils.java](#) -> [Demo](#)

bitmap2Bytes, bytes2Bitmap	: bitmap 与 bytes 互转
drawable2Bitmap, bitmap2Drawable	: drawable 与 bitmap 互转
drawable2Bytes, bytes2Drawable	: drawable 与 bytes 互转
view2Bitmap	: view 转 bitmap
getBitmap	: 获取 bitmap
scale	: 缩放图片
clip	: 裁剪图片
skew	: 倾斜图片
rotate	: 旋转图片
getRotateDegree	: 获取图片旋转角度
toRound	: 转为圆形图片
toRoundCorner	: 转为圆角图片
addCornerBorder	: 添加圆角边框
addCircleBorder	: 添加圆形边框
addReflection	: 添加倒影
addTextWatermark	: 添加文字水印
addImageWatermark	: 添加图片水印
toAlpha	: 转为 alpha 位图
toGray	: 转为灰度图片
fastBlur	: 快速模糊
renderScriptBlur	: renderScript 模糊图片
stackBlur	: stack 模糊图片
save	: 保存图片
isImage	: 根据文件名判断文件是否为图片
getImageType	: 获取图片类型
compressByScale	: 按缩放压缩
compressByQuality	: 按质量压缩
compressBySampleSize	: 按采样大小压缩
getSize	: 获取图片尺寸

• 意图相关 -> [IntentUtils.java](#)

isIntentAvailable	: 判断意图是否可用
getInstallAppIntent	: 获取安装 App（支持 6.0）的意图
getUninstallAppIntent	: 获取卸载 App 的意图
getLaunchAppIntent	: 获取打开 App 的意图
getLaunchAppDetailsSettingsIntent	: 获取 App 具体设置的意图
getShareTextIntent	: 获取分享文本的意图
getShareImageIntent	: 获取分享图片的意图
getComponentIntent	: 获取其他应用组件的意图
getShutdownIntent	: 获取关机的意图
getCaptureIntent	: 获取拍照的意图

• 键盘相关 -> [KeyboardUtils.java](#) -> [Demo](#)

showSoftInput	: 显示软键盘
hideSoftInput	: 隐藏软键盘
toggleSoftInput	: 切换键盘显示与否状态
isSoftInputVisible	: 判断软键盘是否可见

registerSoftInputChangedListener : 注册软键盘改变监听器
 unregisterSoftInputChangedListener: 注销软键盘改变监听器
 fixAndroidBug5497 : 修复安卓 5497 BUG
 fixSoftInputLeaks : 修复软键盘内存泄漏
 clickBlankArea2HideSoftInput : 点击屏幕空白区域隐藏软键盘

• 语言相关 -> [LanguageUtils.java](#) -> [Demo](#)

applySystemLanguage: 应用系统语言
 applyLanguage : 应用语言

• 日志相关 -> [LogUtils.java](#) -> [Demo](#)

getConfig : 获取 log 配置
 Config.setLogSwitch : 设置 log 总开关
 Config.setConsoleSwitch : 设置 log 控制台开关
 Config.setGlobalTag : 设置 log 全局 tag
 Config.setLogHeadSwitch : 设置 log 头部信息开关
 Config.setLog2FileSwitch : 设置 log 文件开关
 Config.setDir : 设置 log 文件存储目录
 Config.setFilePrefix : 设置 log 文件前缀
 Config.setBorderSwitch : 设置 log 边框开关
 Config.setSingleTagSwitch: 设置 log 单一 tag 开关 (为美化 AS 3.1 的 Logcat)
 Config.setConsoleFilter : 设置 log 控制台过滤器
 Config.setFileFilter : 设置 log 文件过滤器
 Config.setStackDeep : 设置 log 栈深度
 Config.setStackOffset : 设置 log 栈偏移
 Config.setSaveDays : 设置 log 可保留天数
 Config.addFormatter : 新增 log 格式化器
 log : 自定义 tag 的 type 日志
 v : tag 为类名的 Verbose 日志
 vTag : 自定义 tag 的 Verbose 日志
 d : tag 为类名的 Debug 日志
 dTag : 自定义 tag 的 Debug 日志
 i : tag 为类名的 Info 日志
 iTag : 自定义 tag 的 Info 日志
 w : tag 为类名的 Warn 日志
 wTag : 自定义 tag 的 Warn 日志
 e : tag 为类名的 Error 日志
 eTag : 自定义 tag 的 Error 日志
 a : tag 为类名的 Assert 日志
 aTag : 自定义 tag 的 Assert 日志
 file : log 到文件
 json : log 字符串之 json
 xml : log 字符串之 xml

• MetaData 相关 -> [MetaDataUtils.java](#) -> [Demo](#)

getMetaDataInApp : 获取 application 的 meta-data 值
 getMetaDataInActivity: 获取 activity 的 meta-data 值
 getMetaDataInService : 获取 service 的 meta-data 值
 getMetaDataInReceiver: 获取 receiver 的 meta-data 值

• 网络相关 -> [NetworkUtils.java](#) -> [Demo](#)

openWirelessSettings : 打开网络设置界面
 isConnected : 判断网络是否连接
 isAvailable[Async] : 判断网络是否可用
 isAvailableByPing[Async]: 用 ping 判断网络是否可用
 isAvailableByDns[Async] : 用 DNS 判断网络是否可用
 getMobileDataEnabled : 判断移动数据是否打开
 isMobileData : 判断网络是否是移动数据
 is4G : 判断网络是否是 4G
 getWifiEnabled : 判断 wifi 是否打开
 setWifiEnabled : 打开或关闭 wifi
 isWifiConnected : 判断 wifi 是否连接状态
 isWifiAvailable[Async] : 判断 wifi 数据是否可用
 getNetworkOperatorName : 获取移动网络运营商名称
 getNetworkType : 获取当前网络类型
 getIPAddress[Async] : 获取 IP 地址
 getDomainAddress[Async] : 获取域名 IP 地址
 getIpAddressByWifi : 根据 WiFi 获取网络 IP 地址
 getGatewayByWifi : 根据 WiFi 获取网关 IP 地址
 getNetMaskByWifi : 根据 WiFi 获取子网掩码 IP 地址
 getServerAddressByWifi : 根据 WiFi 获取服务端 IP 地址

• 对象相关 -> [ObjectUtils.java](#) -> [Test](#)

isEmpty : 判断对象是否为空
 isEmpty : 判断对象是否非空
 equals : 判断对象是否相等
 requireNonNull: 检查对象非空
 getOrDefault : 获取非空或默认对象
 hashCode : 获取对象哈希值

• 路径相关 -> [PathUtils.java](#) -> [Demo](#)

getRootPath : 获取根路径
 getDataPath : 获取数据路径
 getDownloadCachePath : 获取下载缓存路径
 getInternalAppDataPath : 获取内存应用数据路径
 getInternalAppCodeCacheDir : 获取内存应用代码缓存路径
 getInternalAppCachePath : 获取内存应用缓存路径
 getInternalAppDbsPath : 获取内存应用数据库路径
 getInternalAppDbPath : 获取内存应用数据库路径

<code>getInternalAppFilesPath</code>	: 获取内存应用文件路径
<code>getInternalAppSpPath</code>	: 获取内存应用 SP 路径
<code>getInternalAppNoBackupFilesPath</code>	: 获取内存应用未备份文件路径
<code>getExternalStoragePath</code>	: 获取外存路径
<code>getExternalMusicPath</code>	: 获取外存音乐路径
<code>getExternalPodcastsPath</code>	: 获取外存播客路径
<code>getExternalRingtonesPath</code>	: 获取外存铃声路径
<code>getExternalAlarmsPath</code>	: 获取外存闹铃路径
<code>getExternalNotificationsPath</code>	: 获取外存通知路径
<code>getExternalPicturesPath</code>	: 获取外存图片路径
<code>getExternalMoviesPath</code>	: 获取外存影片路径
<code>getExternalDownloadsPath</code>	: 获取外存下载路径
<code>getExternalDcimPath</code>	: 获取外存数码相机图片路径
<code>getExternalDocumentsPath</code>	: 获取外存文档路径
<code>getExternalAppDataPath</code>	: 获取外存应用数据路径
<code>getExternalAppCachePath</code>	: 获取外存应用缓存路径
<code>getExternalAppFilesPath</code>	: 获取外存应用文件路径
<code>getExternalAppMusicPath</code>	: 获取外存应用音乐路径
<code>getExternalAppPodcastsPath</code>	: 获取外存应用播客路径
<code>getExternalAppRingtonesPath</code>	: 获取外存应用铃声路径
<code>getExternalAppAlarmsPath</code>	: 获取外存应用闹铃路径
<code>getExternalAppNotificationsPath</code>	: 获取外存应用通知路径
<code>getExternalAppPicturesPath</code>	: 获取外存应用图片路径
<code>getExternalAppMoviesPath</code>	: 获取外存应用影片路径
<code>getExternalAppDownloadPath</code>	: 获取外存应用下载路径
<code>getExternalAppDcimPath</code>	: 获取外存应用数码相机图片路径
<code>getExternalAppDocumentsPath</code>	: 获取外存应用文档路径
<code>getExternalAppOobbPath</code>	: 获取外存应用 OBB 路径

• 权限相关 -> [PermissionUtils.java](#) -> [Demo](#)

<code>getPermissions</code>	: 获取应用权限
<code>isGranted</code>	: 判断权限是否被授予
<code>isGrantedWriteSettings</code>	: 判断修改系统权限是否被授予
<code>requestWriteSettings</code>	: 申请修改系统权限
<code>isGrantedDrawOverlays</code>	: 判断悬浮窗权限是否被授予
<code>requestDrawOverlays</code>	: 申请悬浮窗权限
<code>launchAppDetailsSettings</code>	: 打开应用具体设置
<code>permission</code>	: 设置请求权限
<code>rationale</code>	: 设置拒绝权限后再次请求的回调接口
<code>callback</code>	: 设置回调
<code>theme</code>	: 设置主题
<code>request</code>	: 开始请求

• 手机相关 -> [PhoneUtils.java](#) -> [Demo](#)

<code>isPhone</code>	: 判断设备是否是手机
<code>getDeviceId</code>	: 获取设备码
<code>getSerial</code>	: 获取序列号
<code>getIMEI</code>	: 获取 IMEI 码

getMEID : 获取 MEID 码
getIMSI : 获取 IMSI 码
getPhoneType : 获取移动终端类型
isSimCardReady : 判断 sim 卡是否准备好
getSimOperatorName : 获取 Sim 卡运营商名称
getSimOperatorByMnc : 获取 Sim 卡运营商名称
getPhoneStatus : 获取手机状态信息
dial : 跳至拨号界面
call : 拨打 phoneNumber
sendSms : 跳至发送短信界面
sendSmsSilent : 发送短信

• 进程相关 -> [ProcessUtils.java](#) -> [Demo](#)

getForegroundProcessName : 获取前台线程包名
killAllBackgroundProcesses : 杀死所有的后台服务进程
killBackgroundProcesses : 杀死后台服务进程
isMainProcess : 判断是否运行在主进程
getCurrentProcessName : 获取当前进程名称

• 反射相关 -> [ReflectUtils.java](#) -> [Test](#)

reflect : 设置要反射的类
newInstance : 实例化反射对象
field : 设置反射的字段
method : 设置反射的方法
get : 获取反射想要获取的

• 正则相关 -> [RegexUtils.java](#) -> [Test](#)

isMobileSimple : 简单验证手机号
isMobileExact : 精确验证手机号
isTel : 验证电话号码
isIDCard15 : 验证身份证号码 15 位
isIDCard18 : 简单验证身份证号码 18 位
isIDCard18Exact : 精确验证身份证号码 18 位
isEmail : 验证邮箱
isURL : 验证 URL
isZh : 验证汉字
isUsername : 验证用户名
isDate : 验证 yyyy-MM-dd 格式的日期校验，已考虑平闰年
isIP : 验证 IP 地址
isMatch : 判断是否匹配正则
getMatches : 获取正则匹配的部分
getSplits : 获取正则匹配分组
getReplaceFirst : 替换正则匹配的第一部分
getReplaceAll : 替换所有正则匹配的部分

• 资源相关 -> [ResourceUtils.java](#) -> [Demo](#)

copyFileFromAssets: 从 assets 中拷贝文件
readAssets2String : 从 assets 中读取字符串
readAssets2List : 从 assets 中按行读取字符串
copyFileFromRaw : 从 raw 中拷贝文件
readRaw2String : 从 raw 中读取字符串
readRaw2List : 从 raw 中按行读取字符串

• Rom 相关 -> [RomUtils.java](#) -> [Demo](#)

isHuawei : 是否华为
isVivo : 是否 VIVO
isXiaomi : 是否小米
isOppo : 是否 OPPO
isLeeco : 是否乐视
is360 : 是否 360
isZte : 是否中兴
isOneplus : 是否一加
isNubia : 是否努比亚
isCoolpad : 是否酷派
isLg : 是否 LG
isGoogle : 是否谷歌
isSamsung : 是否三星
isMeizu : 是否魅族
isLenovo : 是否联想
isSmartisan: 是否锤子
isHtc : 是否 HTC
isSony : 是否索尼
isGionee : 是否金立
isMotorola : 是否摩托罗拉
getRomInfo : 获取 ROM 信息

• 屏幕相关 -> [ScreenUtils.java](#) -> [Demo](#)

getScreenWidth : 获取屏幕的宽度 (单位: px)
getScreenHeight : 获取屏幕的高度 (单位: px)
getAppScreenWidth : 获取应用屏幕的宽度 (单位: px)
getAppScreenHeight : 获取应用屏幕的高度 (单位: px)
getScreenDensity : 获取屏幕密度
getScreenDensityDpi: 获取屏幕密度 DPI
setFullScreen : 设置屏幕为全屏
setNonFullScreen : 设置屏幕为非全屏
toggleFullScreen : 切换屏幕为全屏与否状态
isFullScreen : 判断屏幕是否为全屏
setLandscape : 设置屏幕为横屏
setPortrait : 设置屏幕为竖屏
isLandscape : 判断是否横屏
isPortrait : 判断是否竖屏

getScreenRotation : 获取屏幕旋转角度
screenShot : 截屏
isScreenLock : 判断是否锁屏
setSleepDuration : 设置进入休眠时长
getSleepDuration : 获取进入休眠时长

• SD 卡相关 -> [SDCardUtils.java](#) -> [Demo](#)

isSDCardEnableByEnvironment: 根据 Environment 判断 SD 卡是否可用
getSDCardPathByEnvironment : 根据 Environment 获取 SD 卡路径
getSDCardInfo : 获取 SD 卡信息

• 服务相关 -> [ServiceUtils.java](#)

getAllRunningServices: 获取所有运行的服务
startService : 启动服务
stopService : 停止服务
bindService : 绑定服务
unbindService : 解绑服务
isServiceRunning : 判断服务是否运行

• Shell 相关 -> [ShellUtils.java](#)

execCmd[Async]: 执行命令

• 尺寸相关 -> [SizeUtils.java](#)

dp2px, px2dp : dp 与 px 转换
sp2px, px2sp : sp 与 px 转换
applyDimension : 各种单位转换
forceGetViewSize : 在 onCreate 中获取视图的尺寸
measureView : 测量视图尺寸
getMeasuredWidth : 获取测量视图宽度
getMeasuredHeight: 获取测量视图高度

• Snackbar 相关 -> [SnackbarUtils.java](#) -> [Demo](#)

with : 设置 snackbar 依赖 view
setMessage : 设置消息
setMessageColor: 设置消息颜色
setBgColor : 设置背景色
setBgResource : 设置背景资源
setDuration : 设置显示时长
setAction : 设置行为

setBottomMargin: 设置底边距
show : 显示 snackbar
showSuccess : 显示预设成功的 snackbar
showWarning : 显示预设警告的 snackbar
showError : 显示预设错误的 snackbar
dismiss : 消失 snackbar
getView : 获取 snackbar 视图
addView : 添加 snackbar 视图

• SpannableString 相关 -> [SpanUtils.java](#) -> [Demo](#)

with : 设置控件
setFlag : 设置标识
setForegroundColor: 设置前景色
setBackgroundColor: 设置背景色
setLineHeight : 设置行高
setQuoteColor : 设置引用线的颜色
setLeadingMargin : 设置缩进
setBullet : 设置列表标记
setFontSize : 设置字体尺寸
setFontProportion : 设置字体比例
setFontXProportion: 设置字体横向比例
setStrikethrough : 设置删除线
setUnderline : 设置下划线
setSuperscript : 设置上标
setSubscript : 设置下标
setBold : 设置粗体
setItalic : 设置斜体
setBoldItalic : 设置粗斜体
setFontFamily : 设置字体系列
setTypeface : 设置字体
setAlign : 设置对齐
setClickSpan : 设置点击事件
setUrl : 设置超链接
setBlur : 设置模糊
setShader : 设置着色器
setShadow : 设置阴影
setSpans : 设置样式
append : 追加样式字符串
appendLine : 追加一行样式字符串
appendImage : 追加图片
appendSpace : 追加空白
create : 创建样式字符串

• SP 相关 -> [SPStaticUtils.java](#) -> [Demo](#)

setDefaultSPUtils: 设置默认 SP 实例
put : SP 中写入数据
getString : SP 中读取 String
getInt : SP 中读取 int

getLong	: SP 中读取 long
getFloat	: SP 中读取 float
getBoolean	: SP 中读取 boolean
getAll	: SP 中获取所有键值对
contains	: SP 中是否存在该 key
remove	: SP 中移除该 key
clear	: SP 中清除所有数据

• SP 相关 -> [SPUtils.java](#)

getInstance	: 获取 SP 实例
Instance.put	: SP 中写入数据
Instance.getString	: SP 中读取 String
Instance.getInt	: SP 中读取 int
Instance.getLong	: SP 中读取 long
Instance.getFloat	: SP 中读取 float
Instance.getBoolean	: SP 中读取 boolean
Instance.getAll	: SP 中获取所有键值对
Instance.contains	: SP 中是否存在该 key
Instance.remove	: SP 中移除该 key
Instance.clear	: SP 中清除所有数据

• 字符串相关 -> [StringUtils.java](#) -> [Test](#)

isEmpty	: 判断字符串是否为 null 或长度为 0
isTrimEmpty	: 判断字符串是否为 null 或全为空格
isSpace	: 判断字符串是否为 null 或全为空白字符
equals	: 判断两字符串是否相等
equalsIgnoreCase	: 判断两字符串忽略大小写是否相等
null2Length0	: null 转为长度为 0 的字符串
length	: 返回字符串长度
upperFirstLetter	: 首字母大写
lowerFirstLetter	: 首字母小写
reverse	: 反转字符串
toDBC	: 转化为半角字符
toSBC	: 转化为全角字符

• 线程相关 -> [ThreadUtils.java](#) -> [Test](#)

isMainThread	: 判断当前是否主线程
getFixedPool	: 获取固定线程池
getSinglePool	: 获取单线程池
getCachedPool	: 获取缓冲线程池
getIoPool	: 获取 IO 线程池
getCpuPool	: 获取 CPU 线程池
executeByFixed	: 在固定线程池执行任务
executeByFixedWithDelay	: 在固定线程池延时执行任务
executeByFixedAtFixRate	: 在固定线程池按固定频率执行任务

`executeBySingle` : 在单线程池执行任务
`executeBySingleWithDelay`: 在单线程池延时执行任务
`executeBySingleAtFixRate`: 在单线程池按固定频率执行任务
`executeByCached` : 在缓冲线程池执行任务
`executeByCachedWithDelay`: 在缓冲线程池延时执行任务
`executeByCachedAtFixRate`: 在缓冲线程池按固定频率执行任务
`executeByIo` : 在 IO 线程池执行任务
`executeByIoWithDelay` : 在 IO 线程池延时执行任务
`executeByIoAtFixRate` : 在 IO 线程池按固定频率执行任务
`executeByCpu` : 在 CPU 线程池执行任务
`executeByCpuWithDelay` : 在 CPU 线程池延时执行任务
`executeByCpuAtFixRate` : 在 CPU 线程池按固定频率执行任务
`executeByCustom` : 在自定义线程池执行任务
`executeByCustomWithDelay`: 在自定义线程池延时执行任务
`executeByCustomAtFixRate`: 在自定义线程池按固定频率执行任务
`cancel` : 取消任务的执行
`setDeliver` : 设置任务结束后交付的线程

• 时间相关 -> [TimeUtils.java](#) -> [Test](#)

`millis2String` : 将时间戳转为时间字符串
`string2Millis` : 将时间字符串转为时间戳
`string2Date` : 将时间字符串转为 `Date` 类型
`date2String` : 将 `Date` 类型转为时间字符串
`date2Millis` : 将 `Date` 类型转为时间戳
`millis2Date` : 将时间戳转为 `Date` 类型
`getTimeSpan` : 获取两个时间差（单位：`unit`）
`getFitTimeSpan` : 获取合适型两个时间差
`getNowMills` : 获取当前毫秒时间戳
`getNowString` : 获取当前时间字符串
`getNowDate` : 获取当前 `Date`
`getTimeSpanByNow` : 获取与当前时间的差（单位：`unit`）
`getFitTimeSpanByNow` : 获取合适型与当前时间的差
`getFriendlyTimeSpanByNow`: 获取友好型与当前时间的差
`getMillis` : 获取与给定时间等于时间差的时间戳
`getString` : 获取与给定时间等于时间差的时间字符串
`getDate` : 获取与给定时间等于时间差的 `Date`
`getMillisByNow` : 获取与当前时间等于时间差的时间戳
`getStringByNow` : 获取与当前时间等于时间差的时间字符串
`getDateByNow` : 获取与当前时间等于时间差的 `Date`
`isToday` : 判断是否今天
`isLeapYear` : 判断是否闰年
`getChineseWeek` : 获取中式星期
`getUSWeek` : 获取美式星期
`getValueByCalendarField` : 根据日历字段获取值
`getChineseZodiac` : 获取生肖
`getZodiac` : 获取星座

• 吐司相关 -> [ToastUtils.java](#) -> [Demo](#)

setGravity : 设置吐司位置
setBgColor : 设置背景颜色
setBgResource : 设置背景资源
setMsgColor : 设置消息颜色
setMsgTextSize : 设置消息字体大小
showShort : 显示短时吐司
showLong : 显示长时吐司
showCustomShort: 显示短时自定义吐司
showCustomLong : 显示长时自定义吐司
cancel : 取消吐司显示

• URI 相关 -> [UriUtils.java](#)

file2Uri: file 转 uri
uri2File: uri 转 file

• 震动相关 -> [VibrateUtils.java](#) -> [Demo](#)

vibrate: 震动
cancel : 取消

• 压缩相关 -> [ZipUtils.java](#) -> [Test](#)

zipFiles : 批量压缩文件
zipFile : 压缩文件
unzipFile : 解压文件
unzipFileByKeyword: 解压带有关键字的文件
getFilePath : 获取压缩文件中的文件路径链表
getComments : 获取压缩文件中的注释链表