Android

Android Https WebView H5

HTTPS Android

AndroidHttps HTTPS

`HttpsSwitcher` HTTPS

NORMAL_HTTP(HTTP) OFFLINE_HTTPS ONLINE_HTTPS

```
/**
* Created by xingerkang on 16/10/17.
* Https
*/
public class HttpsSwitcher {
/**HTTP*/
 public final static int NORMAL_HTTP = 1;
/**HTTPS   */
 public final static int OFFLINE_HTTPS = 2;
/**HTTPS */
 public final static int ONLINE_HTTPS = 3;
/**
* HTTPS
*/
 public static int CURRENT_HTTPS_STATUS = ONLIN
E_HTTPS;
/**
* BaseUrl
*/
 public static final int APPCONFING_DOMAIN = 1;
public static final int QYAPI_DOMAIN = 2;//
 public static final int MYAPI_DOMAIN = 3;//
 public static final int PASSPORTAPP_DOMAIN = 4
;//APP
 public static final int QY_DOMAIN = 5;//
 public static final int GET_KEYWORD_DOMAIN = 6
;//
 public static final int DOWNLOAD_APK = 7;//
 public static final int WX_DOMAIN =  8;//
 public static boolean needHttps(){
if(CURRENT_HTTPS_STATUS==NORMAL_HTTP){
return false;
}
return true;
}
/**
* host
* @param host
 * @return
 */
 public static String getHost(int host){
switch (host){
case APPCONFING_DOMAIN:
return needHttps()?"https://app.chinahr.com/":"
http://appconfig.chinahr.com/";
case QYAPI_DOMAIN:
return needHttps()?"https://app.chinahr.com/":"
http://qy.api.chinahr.com/";
case MYAPI_DOMAIN:
return needHttps()?"https://app.chinahr.com/":"
http://my.api.chinahr.com/";
case PASSPORTAPP_DOMAIN:
return needHttps()?"https://apppassport.chinahr
.com/":"http://passport.app.chinahr.com/";
case QY_DOMAIN:
```

```
return needHttps()?"https://app.chinahr.com/":"
http://qy.app.chinahr.com/";
case GET_KEYWORD_DOMAIN:
return needHttps()?"https://app.chinahr.com/":"
http://suggest.chinahr.com/";
case DOWNLOAD_APK:
return needHttps()?"https://app.chinahr.com/":"
http://download.chinahr.com/";
case WX_DOMAIN:
return needHttps()?"https://api.weixin.qq.com/"
:"https://api.weixin.qq.com/";
}
return "";
}
/**
* HTTPS
*/
 public static void switchHttp(OkHttpClient.Bui
lder builder){
//
  if(CURRENT_HTTPS_STATUS==OFFLINE_HTTPS){
builder.sslSocketFactory(SSLFactory.createSSLSo
cketFactory());
builder.hostnameVerifier(new SSLFactory.TrustAl
lHostnameVerifier());
```

```
}
}
}
```

Android Retrofit Apache HttpClient HttpClient Retrofit

HttpClient APK

```
//HttpsSwitcher.getHost URL

getKeywordDomainService = getRetrofitClient(HttpsSwitcher.getHost(HttpsSwitcher.WX_DOMAIN), null).create(ApiService.class);



httpBuilder.connectTimeout(TIME_OUT_CONNECT, TimeUnit.SECONDS)//
 .writeTimeout(TIME_OUT_WRITE_FIlE, TimeUnit.SECONDS)
.readTimeout(TIME_OUT_READ_FILE, TimeUnit.SECONDS)
.retryOnConnectionFailure(true)//
 .addInterceptor(headerInterceptor);//


//HttpsSwitcher.switchHttp HTTPS

HttpsSwitcher.switchHttp(httpBuilder);


httpClient = httpBuilder.build();
retrofit = new Retrofit.Builder()
.baseUrl(baseUrl)
.addConverterFactory(FastJsonConverterFactory.create())//Gson
 .addCallAdapterFactory(RxJavaCallAdapterFactory.create())//RxJava
 .addConverterFactory(ScalarsConverterFactory.create())
.client(httpClient)//HeaderLoggerOKHttpClient
 .build();
```

ImageLoader

```
ImageLoaderConfiguration configuration = new ImageLoaderConfiguration.Bui
lder(context).
.imageDownloader(new AuthImageDownloader(context)).build();

//Https

public class AuthImageDownloader extends BaseImageDownloader {

private SSLSocketFactory mSSLSocketFactory;
public AuthImageDownloader(Context context) {
super(context);
mSSLSocketFactory = SSLFactory.createSSLSocketFactory();
}
public AuthImageDownloader(Context context, int connectTimeout, int readT
imeout) {
super(context, connectTimeout, readTimeout);
mSSLSocketFactory = SSLFactory.createSSLSocketFactory();
}
@Override
  protected InputStream getStreamFromNetwork(String imageUri, Object
extra) throws IOException {
URL url = null;
try {
url = new URL(imageUri);
} catch (MalformedURLException e) {
}
HttpURLConnection conn = (HttpURLConnection) url.openConnection();
conn.setConnectTimeout(connectTimeout);
conn.setReadTimeout(readTimeout);

if (conn instanceof HttpsURLConnection) {
((HttpsURLConnection)conn).setSSLSocketFactory(mSSLSocketFactory);
((HttpsURLConnection)conn).setHostnameVerifier((DO_NOT_VERIFY));
}
return new BufferedInputStream(conn.getInputStream(), BUFFER_SIZE);
}
// always verify the host - dont check for certificate
  final HostnameVerifier DO_NOT_VERIFY = new HostnameVerifier() {
@Override
  public boolean verify(String hostname, SSLSession session) {
return true;
}
};
}
```

WebView H5

webview android 5.0 WebSettings.MIXED_CONTENT_ALWAYS_ALLOW

```
@Override
public void onReceivedSslError(WebView view, SslErrorHandler handler,
SslError error){
handler.proceed();
}


if (Build.VERSION.SDK_INT >= 21) {
webSettings.setMixedContentMode( WebSettings.MIXED_CONTENT_ALWAYS_ALLOW )
;
}
```