腾讯浏览服务

SDK 文档

腾讯云

【版权声明】

【商标声明】

【服务声明】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

文档目录

SDK 文档
SDK 下载

最近更新时间：2025-06-06 16:31:12

腾讯浏览服务文档 SDK 公网版

腾讯浏览服务为接入方提供如下公网版本 SDK，请根据需要下载对应的 SDK：

SDK 类型	SDK 版本	最近更新日期	SDK 大小	SDK 下载	说明	开发者
综合格式 64位	V1.0.5.6000094	2025.6.6	8MB	单击下载	支持 doc、docx、rtf、ppt、pptx、xls、xlsx、xlsm、csv、pdf、txt、epub、chm 等格式文档	深圳市腾讯计算机系统有限公司
综合格式 64位+32位			10MB	单击下载		

其他相关信息

- 产品简介
- 版本信息
- 常见问题
- 个人信息保护规则
- SDK 合规使用指南
- SDK 接入说明
- License 使用说明

SDK 接入说明

最近更新时间：2025-07-07 17:54:41

SDK 介绍

- 腾讯浏览服务已支持以下基础文档格式的本地打开，接入 SDK 可支持打开文档格式：doc、docx、rtf、ppt、pptx、xls、xlsx、xlsm、csv、pdf、txt、epub、chm。
- 其他兼容了 office 格式、遵循 office 标准的文件，可以尝试把文件后缀名转成以上格式，再调用 SDK 打开。

SDK 接入说明

隐私政策

告知开发者《腾讯浏览服务文档 SDK 合规使用指南》《腾讯浏览服务文档 SDK 个人信息保护规则》。
请开发者及终端用户认真阅读上述规则。如您是开发者，请您确认充分了解并同意本规则后再集成 SDK 产品，如果您不同意上述规则及按照本规则履行对应的用户个人信息保护义务，应立即停止接入及使用 SDK 产品。
为了保证腾讯浏览服务 SDK 的正常使用，SDK 需要在 `AndroidManifest.xml` 声明了如下权限：

权限类型	权限使用说明
android.permission.INITERNET	文档能力使用前的授权鉴权功能（离线版本忽略此项）
android.permission.ACCESS_NETWORK_STATE	

在 SDK 的功能使用期间，SDK 需要获取以下信息数据以支持 SDK 基础功能的正常运行。

信息类型	目的	时机	处理方式
获取设备信息（操作系统版本、CPU 类型、屏幕宽高、屏幕方向、屏幕像素）	加载文档引擎必要参数	SDK 初始化获取	内存内使用，退出程序即销毁
应用信息（宿主应用包名，版本号）	加载文档引擎必要参数	SDK 初始化获取	通过标识化、加密传输和处理的安全处理方式

为实现 SDK 的基础功能，SDK 打开文档可能涉及使用到以下权限，但 SDK 不会主动申请权限，也不会判断应用是否有权限，在需要使用功能的同时，宿主根据业务的判断自己决定是否授予，权限的控制由宿主自主决定。下表列举了各个权限使用涉及的功能及影响。

操作系统	权限名称	使用目的及功能场景（申请时机）	是否必选

Android	网络权限	通过 SDK 打开文档时，需要网络保持连接，支持授权鉴权能力。开发者在调用需要该权限的 SDK 功能时进行调用。	必选
	存储权限	若接入方将文档存储在非 App 私有目录下（例如 SD 卡公共目录），则需要申请存储权限，SDK 才能访问并打开文档；若接入方将文档存储在 App 私有目录下，则无需申请存储权限。开发者在调用需要该权限的 SDK 功能时进行调用。	可选
	剪切板权限	用户主动操作文档内容复制粘贴，需要访问剪切板。开发者在调用需要该权限的 SDK 功能时进行调用。	可选

注意：
请务必确保用户同意隐私政策后，先调用 setLicenseKey 接口设置 licensekey，再调用 initEngine 接口进行初始化。

SDK 接入流程

1. 官网 SDK 下载（按照需要选择一种），集成到项目中。例如：在工程根目录下创建 libs 目录，将 aar 文件放到 libs 目录下。
2. 在 App 模块的 build.gradle 中引入 SDK，例如：

```
implementation fileTree(dir: 'libs', include: ['TbsFileSdk_xxxx.aar'])
```

3. 配置混淆，为了功能的正常使用，需要在 proguard-rules.pro 文件中添加如下配置：

```
-dontwarn com.tencent.tbs.reader.**
-keep class com.tencent.tbs.reader.** {
    *;
}
```

4. 参考下文 接入示例-自定义 layout 方式 实现文档打开代码。

接口介绍

腾讯浏览服务 SDK 不使用 TbsReaderView，统一使用 TbsFileInterfaceImpl 下的接口：

LicenseKey 初始化接口

功能介绍：设置客户端绑定的 LicenseKey，LicenseKey 获取请参见 购买方式 。
场景描述：用户同意隐私政策后，首先调用该接口设置 LicenseKey，才能初始化 SDK。

```
public static void setLicenseKey(String licenseStr)
```

SDK 初始化接口

功能介绍：初始化文件 SDK。
场景描述：该方法每次 App 启动后仅需调用一次，只有成功初始化 SDK 才能调用打开文档的其他接口。

```
/**
* 同步初始化SDK
*/
public static intinitEngine(context appContext)

/**
* 异步初始化SDK
*/
public static void initEngineAsync(Context appContext, final
ITbsReaderCallback callBackListener)
```

initEngineAsync 相关回调值：

actionType	args	result	描述
OPEN_FILEREADER_ASYNC_LOAD_READER_ENTRY_CALLBACK	• 0：加载 SDK 成功 • 非0：加载 SDK 失败	/	异步加载 SDK

格式判断接口

功能介绍：判断该文件的格式是否是腾讯浏览服务 SDK 支持打开的文件格式。
场景描述：在打开文件前调用。

```
/**
* @param fileExt  文件后缀名，如文件名为test.pdf，则只需要传入"pdf"
* @return boolean
*

/public static boolean canOpenFileExt(String fileExt)
```

打开文档接口

功能介绍：打开本地文档。

```
public int openFileReader(Context cx, Bundle param, ITbsReaderCallback
callback, FrameLayout layout)
```

参数配置：

- layout：

  传入 null 则使用默认的 dialog 方式，使用默认标题栏。
  传入一个自定义的 layout，则没有标题栏，layout 内只显示文档内容，宿主可以自己实现标题栏及其功能（和 TbsReaderView 方式相同）。
- param：
  - 必要参数

参数名称	参数类型	参数描述
filePath	String	本地文件路径，SDK 只支持打开一个本地文档，服务器文档需要先下载到本地再调 SDK 打开
fileExt	String	文件后缀名，例如文件名为 test.pdf，则只需要传入"pdf"
tempPath	String	文件临时目录路径（文件打开过程中缓存目录，包括记录上次文档打开位置等，打开文档结束后不会自动删除，如有需要可自行删除，建议指定在沙盒目录下）

  - 可选参数

参数名称	参数类型	参数描述
file_reader_enable_long_press_menu	Boolean	开启长按菜单复制功能，支持 PDF、DOCX、XLSX 和 TXT
file_reader_is_ppt_page_mode_default	Boolean	设置 PPT 打开为翻页模式
file_reader_stream_mode	Boolean	设置为文件流打开模式

file_reader_goto_last_pos	Boolean	打开文件自动跳转到上次浏览结束位置
file_reader_content_bg_color	String	设置文档视图的背景颜色，支持 PDF、DOCX、PPTX，如"#ffffff"

- 默认 dialog 模式特定参数

  默认 dialog 模式提供一些自定义标题栏的参数：

参数名称	参数类型	参数描述
file_reader_top_bar_bg_color	String	设置顶 bar 的颜色，如"#ffffff"
file_reader_top_bar_hight	Int	设置顶 bar 的高度

- 自定义 layout 方式特定参数（老 SDK 的 TbsReaderView 方式）

  如果宿主实现了固定高度的标题栏，需要传入额外的参数设置文档内容显示区域的高度（例如屏幕高度-标题栏高度）：

参数名称	参数类型	描述
set_content_view_height	Int	设置文档内容的高度（默认为整个屏幕高度）

openFileReader 相关回调值：

actionType	args	result	描述
OPEN_FILEREADER_STATUS_UI_CALLBACK	打开文件： Bundle[{typeId = 0, typeDes = fileReaderOpened}] 关闭文件： Bundle[{typeId = 1, typeDes = fileReaderClosed}]	/	文件打开关闭事件
NOTIFY_CANDISPLAY	/	/	文档引擎渲染成功，即将显示文档
READER_EVENT_CLICK	/	/	单击事件

READER_EVENT_SCROLL_BEGIN	/	/	滑动开始事件
READER_EVENT_SCROLL_END	/	/	滑动结束事件
READER_EVENT_SCALE_BEGIN	/	/	缩放开始事件
READER_EVENT_SCALE_END	/	/	缩放结束事件
READER_PAGE_TOAST	Bundle[{cur_page = 当前页码, page_count = 总页码}]	/	获取当前页码和总页码

关闭文档接口

功能介绍：关闭文档，释放相关资源。

```
public void closeFileReader()
```

页面跳转接口

功能介绍：支持 PDF、DOCX、PPTX 跳转到指定页面。

```
public void gotoPosition(Bundle b)
```

使用示例：

```
int curPageIndex = -1;
ITbsReaderCallback callback = new ITbsReaderCallback() {
    @Override
    public void onCallBackAction(Integer actionType, Object args, Object
result) {
        if (ITbsReader.READER_PAGE_TOAST == actionType) {
            // cur_page获取到的是当前真实的页码，需要-1才能得到页面对应的数组下标
            curPageIndex = ((Bundle) args).getInt("cur_page") - 1;
        }
    }
};

// 上一页
```

```
Button pre = view.findViewById(R.id.iv_document_previous_page);
pre.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Bundle b = new Bundle();
        b.putInt("progress", curPageIndex - 1);
        TbsFileInterfaceImpl.getInstance().gotoPosition(b); // 传入页面对
应的数组下标
    }
});

// 下一页
Button next = view.findViewById(R.id.iv_document_next_page);
next.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
      Bundle b = new Bundle();
      b.putInt("progress", curPageIndex + 1);
      TbsFileInterfaceImpl.getInstance().gotoPosition(b);
  }
});
```

其他功能

横屏

在 `AndroidManifest.xml` 设置打开文件的 activity 属性：

```
android:configChanges="orientation|keyboardHidden|navigation|screenSize"
```

自定义 layout 方式需要主动调用接口适配横屏，默认 dialog 模式无需调用该接口：

```
/**
* @param width  layout宽度
* @param height  layout高度
*/
public void onSizeChanged(int width, int height)
```

长按菜单

腾讯浏览服务文档引擎（PDF、DOCX、XLSX、TXT）支持长按复制文本功能，见打开文档接口可选参数。

PPT 翻页模式

参考打开文档接口可选参数，设置 PPT 打开为翻页模式（默认是滑动模式）；参考页面跳转接口设置 PPT 翻页模式页面跳转。

接入示例

腾讯浏览服务 SDK 不使用 TbsReaderView，统一使用 TbsFileInterfaceImpl 下的接口，提供以下两种接入方式：

默认 Dialog 方式

openFileReader 接口 layout 传入 null。

```
//设置回调
ITbsReaderCallback callback = new ITbsReaderCallback() {
    @Override
    public void onCallBackAction(Integer actionType, Object args, Object
result) {
        Log.i(TAG, "actionType=" + actionType + ", args=" + args + ",
result=" + result);
        if (ITbsReader.OPEN_FILEREADER_STATUS_UI_CALLBACK == actionType)
{
            if (args instanceof Bundle) {
                int id = ((Bundle) args).getInt("typeId");
                if (ITbsReader.TBS_READER_TYPE_STATUS_UI_SHUTDOWN == id)
{

TbsFileInterfaceImpl.getInstance().closeFileReader();        //关闭
fileReader
                }
            }
        }
    }
};

// 初始化Engine，见下文
int ret = initEngine();
if(ret != 0) {
    Log.i(TAG, "initEngine失败，不要调用其他接口，ret = " + ret);
```

```
        return;
    }

    //点击打开文档
    findViewById(R.id.btn_dialog).setOnClickListener(new
    View.OnClickListener() {
        @Override
        public void onClick(View v) {
            ret = openFileReader(fileName);
        }
    });

    private int openFileReader(String fileName) {
        //设置参数
        Bundle param = new Bundle();
        param.putString("filePath", filePath); //文件路径
        param.putString("fileExt", extName); // 文件后缀名，如文件名为test.pdf,
    则只需要传入"pdf"
        param.putString("tempPath",
    getExternalFilesDir("temp").getAbsolutePath());

        //调用openFileReader打开文档
        if (TbsFileInterfaceImpl.canOpenFileExt(extName)) { //tbs支持的文档类型
            int ret =
    TbsFileInterfaceImpl.getInstance().openFileReader(this, param, callback,
    null);
            if (ret != 0) {
                Log.i(TAG, "openFileReader失败, ret = " + ret);
            }
        } else {
            //tbs不支持的文档类型
            //...
        }
    }
```

自定义 Layout 方式

openFileReader 接口传入一个 layout。

MainActivity.java

```
public class MainActivity extends AppCompatActivity {
    private final String TAG = "FileSdkDemo";
    private static boolean isInit = false;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        initUI();

        // 初始化Engine，见下文
        int ret = initEngine();

        if(ret != 0) {
            Toast.makeText(getApplicationContext(), "初始化Engine失败 ret
= " + ret, Toast.LENGTH_SHORT).show();
        } else {
            Toast.makeText(getApplicationContext(), "初始化Engine成功",
Toast.LENGTH_SHORT).show();
            isInit = true;
        }
    }

    private void initUI() {
        findViewById(R.id.btn_open_file).setOnClickListener(
                view -> openExternalFilesDirDocument(filePath,extName));
    }

    private void openExternalFilesDirDocument(String filePath, String
extName) {
        if(isInit) {
            if(TbsFileInterfaceImpl.canOpenFileExt(fileExt)) {
                PreviewActivity.start(this, filePath, extName);
            } else {
                // tbs 不支持打开的类型
            }
```

```
        } else {
            Toast.makeText(getApplicationContext(), "Engine未初始化成功，无
法打开文件", Toast.LENGTH_SHORT).show();
        }
    }
```

PreviewActivity.java

```
public class PreviewActivity extends AppCompatActivity {
    private FrameLayout mFlRoot; //内容显示区域
    private RelativeLayout mRl; //自定义标题栏
    private boolean isDestroyed = false;

    public static void start(Context context, String filePath,String
fileExt) {
        context = context.getApplicationContext();
        Intent starter = new Intent(context, PreviewActivity.class);
        starter.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        starter.putExtra("filePath", filePath);
        starter.putExtra("fileExt", fileExt);
        context.startActivity(starter);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_preview);
        String filePath = getIntent().getStringExtra("filePath");
        String fileExt = getIntent().getStringExtra("fileExt");
        initView();
        openFile(filePath, fileExt);
    }

    private void initView() {
        mFlRoot = findViewById(R.id.content);
        mRl = findViewById(R.id.reader_top);
    }
```

```
    private void openFile(String filePath, String fileExt) {
        //设置回调
        ITbsReaderCallback callback = new ITbsReaderCallback() {
            @Override
            public void onCallBackAction(Integer actionType, Object
args, Object result) {
                Log.i(TAG, "actionType=" + actionType + ", args=" + args
+ ", result=" + result);
                if (ITbsReader.OPEN_FILEREADER_STATUS_UI_CALLBACK ==
actionType) {
                    if (args instanceof Bundle) {
                        int id = ((Bundle) args).getInt("typeId");
                        if
(ITbsReader.TBS_READER_TYPE_STATUS_UI_SHUTDOWN == id) {
                            finish();        // 加密文档弹框取消需关闭
activity
                        }
                    }
                }
            }
        };

        //设置参数
        Bundle param = new Bundle();
        param.putString("filePath", filePath);
        param.putString("fileExt", fileExt); // 文件后缀名，如文件名为
test.pdf，则只需要传入"pdf"
        param.putString("tempPath",
getExternalFilesDir("temp").getAbsolutePath());
        //调用openFileReader打开文件
        mFlRoot.post(new Runnable() {
            @Override
            public void run() {
                int height = mFlRoot.getHeight();
                // 自定义 layout 模式必须设置这个值，否则可能导致文档内容显示不全
                param.putInt("set_content_view_height", height);
                int ret =
TbsFileInterfaceImpl.getInstance().openFileReader(
                            PreviewActivity.this, param, callback,
mFlRoot);
```

```
            }
        });
    }

    // 销毁资源使用 onpause + ondestroy 的方式。避免 onDestroy 延迟回调
    private void destroy()  {
        if (isDestroyed) {
            return;
        }
        // 回收资源
        mFlRoot.removeAllViews(); //移除内容显示区域 layout
        TbsFileInterfaceImpl.getInstance().closeFileReader(); //关闭
fileReader
        isDestroyed = true;
    }

    @Override
    protected void onPause() {
        super.onPause();
        if (isFinishing()) {
            destroy();
        }
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        destroy();
    }

    //横竖屏切换
    @Override
    public void onConfigurationChanged(@NonNull Configuration newConfig)
{
        super.onConfigurationChanged(newConfig);
        mFlRoot.getViewTreeObserver().addOnGlobalLayoutListener(new
ViewTreeObserver.OnGlobalLayoutListener() {
          @Override
          public void onGlobalLayout() {
```

```
mFlRoot.getViewTreeObserver().removeOnGlobalLayoutListener(this);
                int w = mFlRoot.getWidth();
                int h = mFlRoot.getHeight();
                TbsFileInterfaceImpl.getInstance().onSizeChanged(w, h);
            }
        });
    }
}
```

activity_preview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical">
    // 自定义标题栏
    <RelativeLayout
      android:id="@+id/reader_top"
      android:layout_width="match_parent"
      android:layout_height="50dp"
      android:background="#576B95">
    </RelativeLayout>

    // 内容显示区域
    <FrameLayout
      android:id="@+id/content"
      android:layout_width="match_parent"
      android:layout_height="match_parent"/>
</LinearLayout>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent">
```

```
    <Button
    android:id="@+id/btn_open_file"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="打开文件"
    android:layout_centerInParent="true"/>
</RelativeLayout>
```

同步初始化 Engine

```
public int initEngine() {
    //设置licenseKey
    TbsFileInterfaceImpl.setLicenseKey("your licenseKey");

    int ret = -1;

    //初始化Engine
    if (!TbsFileInterfaceImpl.isEngineLoaded()) {
        ret = TbsFileInterfaceImpl.initEngine(this);
    }

    return ret;
}
```

异步初始化 Engine

```
public void initEngineAsync() {
    //设置 licenseKey
    TbsFileInterfaceImpl.setLicenseKey("your licenseKey");

    ITbsReaderCallback callback = new ITbsReaderCallback() {
    @Override
    public void onCallBackAction(Integer actionType, Object args, Object
result) {
        Log.i(TAG, "actionType=" + actionType + ", args=" + args + ",
result=" + result);
        // ITbsReader.OPEN_FILEREADER_ASYNC_LOAD_READER_ENTRY_CALLBACK 的
值为7002，不是错误码
```

```
        if (ITbsReader.OPEN_FILEREADER_ASYNC_LOAD_READER_ENTRY_CALLBACK
== actionType) {
            int ret = (int)args; // 错误码为 actionType == 7002时 args 的值
            if (ret == 0) {
                // 初始化成功
            } else {
                // 初始化失败
            }
        }
    }
};
if (!TbsFileInterfaceImpl.isEngineLoaded()) {
    TbsFileInterfaceImpl.initEngineAsync(this, callback);
}
}
```

打开文件流

```
ITbsReaderCallback callback = new ITbsReaderCallback() {
    @Override
    public void onCallBackAction(Integer actionType, Object args, Object
result) {
        // 使用文件流模式会收到该 actionType 回调
        if (ITbsReader.OPEN_FILEREADER_STREAM_MODE_CALLBACK ==
actionType) {
            if (args instanceof ITbsReaderStream) {
                ITbsReaderStream stream = (ITbsReaderStream) args;
                FileInputStream fin = null;
                try {
                    // 实际需要打开的文件流
                    File realFile = new
File(this.getExternalFilesDir("docs") + "/RealTbsTestFile.docx");
                    fin = new FileInputStream(realFile);
                    // 文件流打开真正的错误码
                    int ret = stream.write(fin);
                    fin.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
```

```
            }
        }
    }
};

Bundle param = new Bundle();
param.putBoolean("file_reader_stream_mode", true); // 传入该参数表示使用文件流模式打开
param.putString("filePath", filePath); // 该参数必须设置，模拟文件路径。例如："sdcard/Andoird/data/package_name/files/docs/MockTbsTestFile.docx"

// 本地必须创建一个模拟文件，可以为空文件
File file = new File(filePath);
File fileParent = file.getParentFile();
if(!fileParent.exists()){
    fileParent.mkdirs();
}
file.createNewFile();

param.putString("fileExt", fileExt); // 该参数必须设置，后缀名必须和传入后续要打开的文件流类型一致。如：打开 docx 文件流，就要传入"docx"
param.putString("tempPath", ""); // 该参数无需设置

if (TbsFileInterfaceImpl.canOpenFileExt(fileExt)) {
    // 使用文件流模式，ret 会返回-8。
    int ret = TbsFileInterfaceImpl.getInstance().openFileReader(this, param, callback, null);
}
```

License 使用说明

最近更新时间：2024-06-26 10:38:51

License 介绍

腾讯浏览服务文档 SDK 通过 License 许可证来校验所有功能模块的授权，客户端通过设置对应的 LicenseKey 以获取 SDK 文档能力，一个客户端只能绑定一个 LicenseKey。

License 接入

在调用所有 SDK 接口前，设置客户端对应的 LicenseKey，验证通过即可使用文档功能。如果接入客户端时没有提供正确的 LicenseKey，则授权不通过，SDK 初始化会给出相应报错日志。设置 LicenseKey 的接口如下：

```
public static void setLicenseKey(String licenseStr)
```

注意事项

LicenseKey 生成后请妥善保管，如有泄露请 联系客服 进行更换。

SDK 版本信息

最近更新时间：2025-06-06 16:31:12

V1.0.5.6000094 @ 2025.06.06

bug 及功能优化

- 修复部分文档显示问题
- 修复其他已知问题

V1.0.5.6000030 @ 2023.11.09

bug 及功能优化

- 适配 targetSdkVersion 34
- 修复部分 docx 文档长表格显示不全
- 修复其他已知问题

V1.0.5.6000023 @ 2023.10.16

bug 及功能优化

- 修复部分 xls 文档数值引用问题
- 修复部分 docx 文档背景填充色无法显示
- 修复 ppt 单页模式横竖屏切换显示问题

V1.0.5.6000022 @ 2023.09.08

bug 及功能优化

- 修复部分 xlsx 文档公式计算问题
- 修复部分 pptx 文档版式和渐变背景无法显示
- 修复 Android 12加载组件兼容性问题

V1.0.5.6000021 @ 2023.07.27

bug 及功能优化

- 修复部分 docx 文档表格显示问题
- 修复部分 xlsx 文档表格显示问题
- 修复部分 xlsx 文档公式计算错误问题
- 修复其他已知问题

V1.0.5.6000020 @ 2023.07.10

新增功能

支持自定义文档视图背景颜色

bug 及功能优化

- 修复部分机型 Android 12兼容性问题
- 修复异步加载稳定性问题
- 修复密码弹框文本显示问题
- 修复部分引擎的已知问题

V1.0.5.6000019 @ 2023.06.09

bug 及功能优化

- 修复部分引擎页面跳转问题
- 优化 SDK 加载流程
- 文件打开默认不会自动跳转上次打开位置，需要通过 openFileReader 接口参数启用该功能
- 优化 SDK 重新可用逻辑，资源包消耗完后绑定新资源包可以快速重新生效

V1.0.5.6000018 @ 2023.05.15

新增功能

支持文件流模式打开文档

bug 及功能优化

- 修复部分折叠屏机型兼容性问题
- 修复部分 xlsx 文档图片内容不清晰问题
- 修复部分 docx 文档 emf 图片无法显示问题

V1.0.5.6000017 @ 2023.04.22

bug 及功能优化

- 修复部分机型初始化兼容性问题
- 修复部分 docx 文档表格框显示问题
- 修复部分 rtf 文档显示乱码问题

V1.0.5.6000016 @ 2023.04.13

bug 及功能优化

- 修复部分机型 Android 12兼容性问题
- 修复 SDK 初始化问题
- 优化组件加载

V1.0.5.6000015 @ 2023.03.28

bug 及功能优化

修复一些已知问题

V1.0.5.6000014 @ 2023.03.17

bug 及功能优化

修复一些已知问题

V1.0.5.6000013 @ 2023.03.13

新增接口

判断 SDK 是否加载成功 isEngineLoaded

bug 及功能优化：

- 修复 ppt 文档页面显示空白
- 修复 word 文档内容显示重叠和退出卡顿
- 修复 word 文档排版无效
- 修复 excel 文档共享公式计算

SDK 合规使用指南

最近更新时间：2025-12-23 09:52:02

为帮助使用腾讯浏览服务文档 SDK（以下简称文档 SDK 或 SDK）的开发运营者（以下简称“您”）在符合个人信息保护相关法律法规、政策及标准的规定下合规接入、使用第三方 SDK，深圳市腾讯计算机系统有限公司（以下简称"我们"）特制定《文档 SDK 合规使用指南》（以下简称“指南”），便于您使用文档 SDK 过程中符合相应的合规要求。请您在接入、使用文档 SDK 前，充分阅读和了解本指南内容。

一、接入/升级至满足监管新规的最新 SDK 版本

我们高度重视 SDK 的功能优化、个人信息安全和保护，将适时升级迭代 SDK 版本以提升产品的安全性和稳定性，确保符合相关法律法规及、监管及标准的最新合规要求。强烈建议您升级使用最新版本 SDK，以便保障您正常使用 SDK 最新功能、避免因您更新不及时产生的不利影响（例如 App 被通报或下架等）。
SDK 更新后，我们会及时通过官网公告通知、短信、站内信、邮件或其他适当的方式提醒您更新的内容，以便您及时了解 SDK 最新版本信息。同时，您可以访问或申请 SDK 最新版本链接：腾讯浏览服务-文档 SDK 下载 。

二、App 隐私政策中应披露第三方 SDK 相关情况

请您确保您开发或运营的 App 配备了符合监管要求的《隐私政策》文本。请您务必明确告知终端用户您的 App 集成了第三方 SDK 服务。您应在《隐私政策》中添加关于本 SDK 收集使用个人信息的目的、方式和范围等，并显示本 SDK 的开发运营者名称及隐私政策链接。您应在 App 登录注册页面及 App 首次运行时，通过弹窗、文本链接及附件等简洁明显且易于访问的方式，应当以清晰易懂的语言告知用户《隐私政策》，由用户在充分知情的前提下，作出自愿明确的意思表示。
我们提供以下告知文案示例供您参考，您可以通过文字或表格方式向用户告知。请您理解 SDK 不同版本提供的功能服务及所需的字段信息可能会因开发者的选择或配置不同而存在差异，因此请您参考 SDK 隐私政策及您实际接入使用的 SDK 运行情况向用户进行充分告知并获得用户的同意。
仅 Android 参考示例：

- 第三方 SDK 名称：腾讯浏览服务文档 SDK（Android）。
- 第三方 SDK 提供方的公司名称：深圳市腾讯计算机系统有限公司。
- 使用目的及功能场景：为用户提供稳定、安全、高兼容性的本地文档浏览服务。
- 处理的个人信息类型：设备信息（设备型号、操作系统、CPU 类型）、应用信息（宿主应用包名，版本号）。
- 实现 SDK 产品功能所需的权限：网络权限、存储权限（可选）、剪切板（可选）。
- 第三方 SDK 隐私政策链接：用户规则中心 。
- 第三方 SDK 官网链接（可选披露）：腾讯浏览服务 。

三、获得用户同意后再初始化 SDK

为满足法律法规及监管要求，您应确保在获得用户的同意后再初始化 SDK，并在用户触发 SDK 具体功能服务后通过配置 SDK 的相关参数完成发送请求的调用，此时 SDK 才会按照您设置的配置方式采集功能所需的个人信息或申

请功能所需的权限。为了避免您在获取用户同意前，提前启动 SDK 收集使用用户个人信息，SDK 提供了延迟 SDK 初始化调用的 API 接口、合规初始化技术配置方案，单击 腾讯浏览服务 查看详细操作指引。

1. 确保在用户阅读 App 隐私政策并取得用户授权之后，按 App 功能需要在合适时机调用 init 接口初始化 SDK。反之，如果用户不同意《隐私政策》授权，则不能调用初始化接口。该接口仅进行初始化，不会获取个人信息。
2. 请勿在用户同意隐私政策之前动态申请涉及用户个人信息的敏感设备权限；请勿在用户同意隐私政策前私自采集和上报个人信息（尤其注意 Android_ID、OAID、IMEI、MAC 地址、硬件序列号、应用安装列表等用户信息）。请勿在 App 处于未激活状态时（例如 App 在后台运行），请求 SDK 相关服务。

四、可选信息配置开关

SDK 向您提供了可选权限的控制开关，您可以根据 App 所需的 SDK 功能服务自行配置打开或关闭隐私信息请求开关。

1. 配置可选权限

请您注意，SDK 不强制获取可选权限，即使没有获取可选权限，SDK 提供的基本功能也能正常运行。您可以配置可选权限，以便使用 SDK 提供的其他可选功能。建议调用请求前在合适的时机调用 SDK 提供的方法，在用户授权的情况下获取声明中的权限。

操作系统	权限名称	使用目的及功能场景（申请时机）	是否可选
Android	网络权限	通过 SDK 打开文档时，需要网络保持连接，支持授权鉴权能力。开发者在调用需要该权限的 SDK 功能时进行调用。	必选
	存储权限	若接入方将文档存储在非 App 私有目录下（例如 SD 卡公共目录），则需要申请存储权限，SDK 才能访问并打开文档；若接入方将文档存储在 App 私有目录下，则无需申请存储权限。开发者在调用需要该权限的 SDK 功能时进行调用。	可选
	剪切板	用户主动操作文档内容复制粘贴，需要访问剪切板。开发者在调用需要该权限的 SDK 功能时进行调用。	可选

关于存储权限

若接入方将文档存储在非 App 私有目录下（例如 SD 卡公共目录），则需要申请存储权限，SDK 才能访问并打开文档；若接入方将文档存储在 App 私有目录下，则无需申请存储权限。

关于剪切板权限

接入方可以从功能的层面关闭剪切板的使用，从而不让文档 SDK 申请剪切板权限：关闭长按菜单选择复制的使用。具体关闭方法：

```
param.putBoolean("file_reader_enable_long_press_menu", false);
```

2. 指导建议

请您重点关注，在 App 安装、运行和使用相关功能时，您应遵从国家相关法律法规、监管政策及标准的要求，收集用户个人信息或申请敏感权限，不得存在以下违规行为：
（1）未经用户同意不得收集任何个人信息。
（2）非服务所必需或无合理应用场景下，用户拒绝相关授权申请后，应用不得自动退出或关闭。
（3）在用户明确拒绝权限申请后，APP不应向用户频繁弹窗或反复申请开启与当前服务场景无关的权限、影响用户正常使用，建议掌握合适时机申请敏感权限，不得影响其他功能可用。
（4）不得未明确告知用户索取权限的目的和用途。
（5）App 首次打开或运行中，未见使用权限对应的相关功能或服务时，不应提前向用户弹窗申请开启敏感权限。
（6）不得超出业务功能实际需要过度收集个人信息。

五、用户权利保障机制

如果您需要我们协助来实现您最终用户的个人信息主体权利请求，您可以通过“联系方式”来申请协助。

六、联系方式

我们设立了专门的个人信息保护团队和个人信息保护负责人，如果您和/或终端用户对本规则或个人信息保护相关事宜有任何疑问或投诉、建议时，可以通过以下方式与我们联系：

1. 通过 腾讯客服 或 填写工单 与我们联系。
2. 将问题发送至 `Dataprivacy@tencent.com` 。
3. 邮寄信件至：中国广东省深圳市南山区海天二路33号腾讯滨海大厦数据隐私保护部（收），邮编：518054。

我们将尽快审核所涉问题，并在15个工作日或法律法规规定的期限内予以反馈。

七、注意事项

- 您接入文档 SDK 前的合规自查：

  为确保您就本SDK的使用获得终端用户的授权，且遵守个人信息保护要求和合规流程，我们建议您在接入文档 SDK 前进行合规自查。
  - 请仔细阅读并按本说明指南提示对您 App 的《隐私政策》进行合规自查。
  - 请务必做延迟初始化配置，确保获得用户同意后再初始化 SDK。
  - 当文档 SDK 基于最新的法律法规或监管要求进行更新后，请您在收到版本更新通知时及时将您 App 集成的文档 SDK 升级到最新版本。
  - 其他国家相关法律法规、监管政策及标准的要求。
- 文档 SDK 对您的合规审查：

  您应遵守个人信息保护相关的法律法规、监管政策及标准，并确保合规使用文档 SDK 服务。请您知悉，为确保您切实获得终端用户的授权，且您已满足个人信息保护相关合规要求，文档 SDK 可能视具体情况，在双方订立协议、开展合作前或合作过程中，对您进行必要的个人信息保护合规审查。在该等合规审查过程中，您应当提供必要的配合，包括但不限于：
  - 要求您提供所共享的个人信息的合法来源证明及相关文件。

- 查阅您官网及其他公开渠道可获取的隐私政策文本。
- 试用您的 App 以审查同意授权告知机制及其他合规机制。

如文档 SDK 发现存在不合规情形，您可能会被要求增加或补充相关合规措施，如您未按时增加或补充，文档 SDK 有权拒绝您使用服务。请您知悉，该等合规审查仅属于我们内部必要的合规程序，不构成任何形式的承诺与保证，不具有法律效力。

- 以下合规文件供开发者参考：
  - 《个人信息保护法》
  - 《工业和信息化部关于进一步提升移动互联网应用服务能力的通知》
  - 《工业和信息化部关于开展信息通信服务感知提升行动的通知》
  - 《工业和信息化部关于开展纵深推进 App 侵害用户权益专项整治行动的通知》
  - 《工业和信息化部关于开展 App 侵害用户权益专项整治工作的通知》
  - 《App 违法违规收集使用个人信息行为认定方法》
  - 《网络安全标准实践指南—移动互联网应用程序（App）收集使用个人信息自评估指南》
  - 《常见类型移动互联网应用程序必要个人信息范围规定》
  - 《GB/T 35273-2020信息安全技术 个人信息安全规范》
  - 《网络安全标准实践指南—移动互联网应用程序（App）使用软件开发工具包（SDK）安全指引》
  - 《网络安全标准实践指南—移动互联网应用程序（App）系统权限申请使用指南》