

Android点播SDK集成文档

简介

百家云Android点播SDK是一个支持视频在线播放和离线播放，视频缓存，可自定义UI界面的播放器。播放器继承自Framelayout, 底层采用ijkplayer, 您可以方便地添加到您自己的app当中。推荐使用Android Studio集成。

功能描述

SDK支持Android4.0 (api level 14) 及以上

功能	描述
在线播放	支持百家云后台配置视频播放以及http(s)协议视频url播放
离线播放	支持本地视频绝对路径，加密和不加密详见视频缓存模块
片头片尾	支持视频片头片尾播放，可在百家云后台配置
自定义界面	sdk提供标准界面，用户可自行修改
视频缓存	支持百家云后台配置视频的缓存功能

示例工程





github链接: <https://github.com/baijia/BJVideoPlayerDemo-Android>(apk在apk_bin目录下)

集成SDK

集成前的准备

- 1) 推荐使用最新版Android studio集成SDK [点击下载](#) (需科学上网)
- 2) 在工程根目录下build.gradle添加远程仓库

```
1. maven { url 'https://raw.githubusercontent.com/baijia/maven/master' }
```

对于部分国内用户, 如果github被墙或者访问比较慢, 可以使用我们国内的镜像仓库

```
1. maven { url 'http://git.baijiashilian.com/open-android/maven/raw/master/' }
```

- 3) 在需要集成的module添加如下依赖

```
1. compile 'com.baijia.player:videoplayer:1.7.27'
```

videoplayer默认依赖support包的版本为25, 如与本地support版本冲突, 可以使用如下集成方式

```
1. compile ('com.baijia.player:videoplayer:1.7.27'){
2. exclude group: 'com.android.support'
3. }
```

推荐使用[最新版本](#)

- 4) 在build.gradle中添加ndk过滤

请在过滤so库文件务必同时包含armabi和armeabi-v7a两种类型, 否则某些机型上可能出现音视频不同步的情况

```
1. ndk {
2. abiFilters 'armeabi-v7a', 'armeabi', 'x86' //x86虚拟机测试用, 发版可去掉
3. }
```

- 5) 设置客户专属域名前缀 专属域名从百家云账号中心获取, 需要在进入回放或点播之前设置。例如专属域名为demo123.at.baijiayun.com, 则前缀为demo123, 参考[专属域名说明](#)。

```
1. PlayerConstants.CUSTOM_DOMAIN = "demo123";
```

快速集成

- 1) 在布局文件中添加播放器控件

```
1. <com.baijihulian.player.BJPlayerView
2. android:id="@+id/pv_ac_quiz_play_demo"
3. android:layout_width="match_parent"
4. android:layout_height="match_parent"
5. app:aspect_ratio="fit_parent">
6. </com.baijihulian.player.BJPlayerView>
```

2) 初始化播放器

application初始化

```
1. public class App extends Application {
2.     @Override
3.     public void onCreate() {
4.         super.onCreate();
5.         BJVideoPlayerSDK.getInstance().init(this);
6.     }
7. }
```

manifest.xml文件配置

```
1. <application
2.     android:name=".App"
3.     ...其他配置
```

BJPlayerView初始化

```
1. //变量声明
2. private BJPlayerView playerView;
3. //根据id查找播放器
4. playerView = (BJPlayerView) findViewById(R.id.pv_ac_quiz_play_demo);
5. //以下三个方法分别设置底部、顶部和中部分界面
6. playerView.setBottomPresenter(new BJBottomViewPresenter(playerView.getBottomView()));
7. playerView.setTopPresenter(new BJTopViewPresenter(playerView.getTopView()));
8. playerView.setCenterPresenter(new BJCenterViewPresenter(playerView.getCenterView()));
9. //初始化partnerId, 第一个参数换成您的partnerId
10. playerView.initPartner(123456L, BJPlayerView.PLAYER_DEPLOY_ONLINE);
```

3) 播放

在合适的时机播放视频，比如点击事件

```
1. //第一个参数为百家云后台配置的视频id, 第二个参数为视频token
2. playerView.setVideoId(Long.valueOf(videoId), videoToken);
3. //播放
4. playerView.playVideo();
```

4) 回调接口

回调接口为播放器状态改变之后向上层app的通知，可以在每个回调方法中实现自己的业务逻辑。

如果仅需要实现部分接口，可继承SimpleOnPlayerViewListener。

```

1. playerView.setOnPlayerViewListener(new OnPlayerViewListener() {
2.     @Override
3.     public void onVideoInfoInitialized(BJPlayerView playerView, HttpException exception) {
4.         //TODO: 视频信息初始化结束
5.     }
6.     @Override
7.     public void onPause(BJPlayerView playerView) {
8.         //TODO: video暂停
9.     }
10.    @Override
11.    public void onPlay(BJPlayerView playerView) {
12.        //TODO: 开始播放
13.    }
14.    @Override
15.    public void onError(BJPlayerView playerView, int code) {
16.        //TODO: 播放出错
17.    }
18.    @Override
19.    public void onUpdatePosition(BJPlayerView playerView, int position) {
20.        //TODO: 播放过程中更新播放位置
21.    }
22.    @Override
23.    public void onSeekComplete(BJPlayerView playerView, int position) {
24.        //TODO: 拖动进度条
25.    }
26.    @Override
27.    public void onSpeedUp(BJPlayerView playerView, float speedUp) {
28.        //TODO: 设置倍速播放
29.    }
30.    @Override
31.    public void onVideoDefinition(BJPlayerView playerView, int definition) {
32.        //TODO: 设置清晰度完成
33.    }
34.    @Override
35.    public void onPlayCompleted(BJPlayerView playerView, VideoItem item, SectionItem nextSection) {
36.        //TODO: 当前视频播放完成 [nextSection已被废弃, 请勿使用]
37.    }
38.    @Override
39.    public void onVideoPrepared(BJPlayerView playerView) {
40.        //TODO: 准备好了, 马上要播放
41.        // 可以在这时获取视频时长
42.        playerView.getDuration();
43.    }
44.    @Override
45.    public String getVideoTokenWhenInvalid() {
46.        //TODO 视频token出错, 需要集成方重新获取并传入BJPlayerView。
47.        return "test12345678";
48.    }
49. });

```

5) 生命周期绑定

为防止转屏导致界面重绘, 在manifest.xml中相应Activity加入如下配置

```
1. android:configChanges="keyboardHidden|orientation|screenSize"
```

在Activity或者fragment生命周期回调中加入对应的播放器调用：

```
1. @Override
2. protected void onResume() {
3. //如果要实现后台播放则不用调用onResume()
4. super.onResume();
5. if (playerView != null) {
6. playerView.onResume();
7. }
8. }
9. @Override
10. protected void onPause() {
11. super.onPause();
12. //如果要实现后台播放则不用调用onPause()
13. if (playerView != null) {
14. playerView.onPause();
15. }
16. }
17. @Override
18. protected void onDestroy() {
19. super.onDestroy();
20. if (playerView != null) {
21. playerView.onDestroy();
22. }
23. }
```

至此，点播sdk快速集成完成。

自定义播放器样式

如果您不想使用SDK自带界面，您可以自己开发自己的界面，代替自带界面

1.新建xml布局

在xml播放器控件中添加自定义布局文件，如果不想使用某一块界面可以将其设置为空白的布局

```
1. //顶部布局
2. app:top_controller="@layout/bjplayer_layout_top_controller"
3. //底部布局
4. app:bottom_controller="@layout/bjplayer_layout_bottom_controller"
5. //中部布局
6. app:center_controller="@layout/bjplayer_layout_center_controller"
```

2.实现接口

新建class分别实现TopView、CenterView、BottomView回调接口

接口说明如下

```
1. interface TopView {
2. /**
3. * 绑定presenter和view
4. */
5. void onBind(IPlayer player);
6. /**
7. * 设置视频标题
8. *
9. * @param title 视频标题
10. */
11. void setTitle(String title);
12. /**
13. * 设置了播放器方向
14. */
15. void setOrientation(int orientation);
16. /**
17. * 返回键监听
18. */
19. void setOnBackClickListener(View.OnClickListener listener);
20. }
```

```
1. interface BottomView {
2. /**
3. * 绑定presenter和view
4. */
5. void onBind(IPlayer player);
6. /**
7. * 设置视频长度
8. *
9. * @param duration 播放长度
10. */
11. void setDuration(int duration);
12. /**
13. * 设置播放位置
14. *
15. * @param position 播放位置
16. */
17. void setCurrentPosition(int position);
18. /**
19. * 设置是否在播放
20. *
21. * @param isPlaying true在播放， false 不在播放
22. */
23. void setIsPlaying(boolean isPlaying);
24. /**
25. * 设置方向
26. *
27. * @param orientation 参考android类Configuration的常量
28. * Configuration.ORIENTATION_LANDSCAPE
29. * Configuration.ORIENTATION_PORTRAIT
30. */
31. void setOrientation(int orientation);
32. /**
33. * 更新进度
34. *
35. * @param percent 缓冲进度百分比
36. */
37. void onBufferingUpdate(int percent);
38. /**
39. * 设置进度条是否可以拖动
40. *
41. * @param canDrag true可以拖动， false不可以拖动
42. */
43. void setSeekBarDraggable(boolean canDrag);
44. }
```

```
1. interface CenterView {
2. /**
3. * 绑定presenter和view
4. */
5. void onBind(IPlayer player);
6. /**
7. * 点击返回
8. *
```

```
8.
9. * @deprecated
10. */
11. boolean onBackTouch();
12. /**
13. * 设置播放器方向
14. *
15. * @param orientation 方向int值，同Android系统Confi
16. */
17. void setOrientation(int orientation);
18. /**
19. * 滑动fling手势进度
20. */
21. void showProgressSlide(int delta);
22. /**
23. * 显示loading
24. *
25. * @param message loading信息
26. */
27. void showLoading(String message);
28. /**
29. * 隐藏loading
30. */
31. void dismissLoading();
32. /**
33. * 音量手势显示
34. *
35. * @param volume 当前音量
36. * @param maxVolume 最大音量
37. */
38. void showVolumeSlide(int volume, int maxVolu
39. /**
40. * 亮度显示
41. *
42. * @param brightness 亮度
43. */
44. void showBrightnessSlide(int brightness);
45. /**
46. * 显示错误
47. */
48. void showError(int what, int extra);
49. void showError(int code, String message);
50. /**
51. * 显示警告
52. */
53. void showWarning(String warn);
54. /**
55. * 显示视图
56. */
57. void onShow();
58. /**
59. * 隐藏视图
60. */
```



```
61. void onHide();
62. /**
63. * 视频信息加载完成
64. *
65. * @param videoItem 视频信息model
66. */
67. void onVideoInfoLoaded(VideoItem videoItem);
68. /**
69. * @deprecated
70. */
71. boolean isDialogShowing();
72. /**
73. * 清晰度调整
74. */
75. void updateDefinition();
76. }
```

3.传入接口的实现

将实现类传给BJPlayerView（初始化时）：

```
1. playerView.setBottomPresenter(BottomViewImpl);
2. playerView.setTopPresenter(TopViewImpl);
3. playerView.setCenterPresenter(CenterViewImpl);
```

4.开源了

考虑到自定义播放器样式有一定难度，BJPlayerView的自身实现已经在demo中开源，大家可以参考。

需要注意的是，TopView、CenterView、BottomView是为了自定义播放器样式而抽象的一套接口，配合1）中描述的自定义布局文件可以实现定制化的回调效果。集成时开发者可以选择性实现部分回调接口，如全部空实现则播放器无以上的回调提示。

下载模块

视频缓存(旧版)

视频缓存仅支持下载百家云后台配置的视频，不支持任意视频文件下载。

说明：旧版本在com.baijiahulian的包名下

1.初始化数据库

```
1. /**
2. * 设置用户数据库名称，比如123.db 如果app不使用sdk下载模块或者下载不区分用户则不必调用
3. * 请务必在VideoDownloadManager初始化之前调用
4. */
5. BJVideoPlayerSDK.getInstance().setCurUserDBName(dbName);
```

2.初始化下载管理器

可在Activity、fragment或者application中初始化，传入Context实例

```

1. VideoDownloadManager downloadManager = VideoDownloadService.getDownloadManager(this);
2. //第一个参数为您申请的partnerId
3. downloadManager.initDownloadPartner(32975272, BJPlayerView.PLAYER_DEPLOY_ONLINE, 1);
4. //设置下载目标路径
5. downloadManager.setTargetFolder(Environment.getExternalStorageDirectory().getAbsolutePath() +
   "/aa_video_downloaded/");
6. //设置最大下载并发数，默认3个
7. downloadManager.getThreadPool().setCorePoolSize(4);

```

```

1. /**
2. * 单独初始化下载
3. * @param partnerId 合作ID
4. * @param deploy 服务器环境
5. * @param encryptType 视频是否加密 1: 加密; 0: 不加密
6. */
7. public void initDownloadPartner(long partnerId, int deploy, int encryptType)

```

3.获取某个视频的清晰度

```

1. /**
2. * 获取当前视频的所有清晰度
3. *
4. * @param videoId vid
5. * @param videoToken token
6. * @param listener 监听
7. */
8. public void getVideoDefinitionById(int videoId, String videoToken, OnVideoDefinitionListener listener)

```

4.添加下载任务

不推荐，需要提前获取视频的清晰度。

```

1. /**
2. * @param videoId 视频id
3. * @param videoToken 视频token
4. * @param definitions 清晰度列表，优先匹配第一个清晰度
5. * @param encryptType 是否加密 0 不加密， 1加密
6. * @param listener 获取视频信息回调 可为null
7. */
8. public void addDownloadVideoTask(String fileName, int videoId, String videoToken, List<VideoDefinition>
   definitions, int encryptType, OnVideoInfoGetListener listener){}

```

推荐，传入清晰度list,从前往后匹配清晰度。

```

1. /**
2. * @param fileName
3. * @param videoId 视频id
4. * @param videoToken 视频token
5. * @param definitions 清晰度列表，优先匹配第一个清晰度
6. * @param encryptType 是否加密 0 不加密， 1加密
7. * @param extraInfo 自定义额外信息
8. * @param listener 获取视频信息回调 可为null
9. */
10. public void addDownloadVideoTask(String fileName, int videoId, String videoToken, List<VideoDefinition>
   definitions, int encryptType, String extraInfo, OnVideoInfoGetListener listener){}

```

5.在合适位置释放下载器，比如Activity的onDestroy方法：

```
1. BJVideoPlayerSDK.getInstance().releaseDownloadClient();
```

6.视频缓存(VideoDownloadManager类)

获取所有下载任务

```
1. /**
2.  * 获取所有下载任务
3.  */
4. public List<DownloadInfo> getAllTask()
```

DownloadInfo字段说明如下：

```
1. private int id; //id自增长
2. private String taskKey; //下载的标识键
3. private String url; //文件URL
4. private String targetFolder; //保存文件夹
5. private String targetPath; //保存文件地址
6. private String fileName; //保存的文件名
7. private float progress; //下载进度
8. private long totalLength; //总大小
9. private long downloadLength; //已下载大小
10. private long networkSpeed; //下载速度
11. private int state = 0; //当前状态
12. private BaseRequest request; //当前任务的网络请求
13. @Deprecated
14. private Serializable data; //额外的数据 deprecated
15. private int videoId; //视频id
16. private int videoType; //清晰度
17. private int encryptType; //加密类型 0不加密， 1加密
18. private String videoToken; //视频token， 过期逻辑上层处理
19. private String extraInfo; //额外信息， 客户自己定义， 这里只是转存
```

添加所有任务监听：

```
1. public void addOnAllTaskEndListener(OnAllTaskEndListener allTaskEndListener)
```

移除所有任务监听：

```
1. public void removeOnAllTaskEndListener(OnAllTaskEndListener allTaskEndListener)
```

根据taskKey（下载标识键）获取下载信息

```
1. public DownloadInfo getDownloadInfo(String taskKey)
```

开始所有下载任务

```
1. public void startAllTask()
```

暂停所有下载任务

```
1. public void pauseAllTask()
```

移除所有下载任务

1. //删除所有任务和对应的文件
2. `public void removeAllTaskAndFiles()`

重新下载某个任务

1. `public void restartTask(String taskKey)`

暂停某个任务

1. `public void pauseTask(String taskKey)`

删除某个任务

1. //false只删除任务，不删文件，true删除任务和已下载文件
2. `public void removeTask(String taskKey, boolean isDeleteFile)`

获取下载目录

1. `public String getTargetFolder()`

视频缓存（新版）

重构后的视频下载模块逻辑更加精简，数据缓存改用磁盘缓存，下载模块不再提供线程管理功能，只提供下载的基本功能，集成者可以根据需求在此基础上自行实现并发控制。说明：旧版本在com.baijiayun的包名下

1.manifest.xml中配置DownloadService

1. `<service`
2. `android:name="com.baijiayun.download.DownloadService"`
3. `android:enabled="true"`
4. `android:exported="false"></service>`

2.初始化DownloadManager

1. //初始化下载
2. `manager = DownloadService.getDownloadManager(this);`
3. //设置缓存文件路径
4. `manager.setTargetFolder(Environment.getExternalStorageDirectory().getAbsolutePath() +
"/bb_video_downloaded/");`
5. //读取磁盘缓存的下载任务并初始化加密参数
6. `manager.loadDownloadInfo(32975272, 1);`

参数说明

1. /**
2. * 从cache缓存中读取下载记录并设置加密参数
3. * @param partnerId
4. * @param encryptType 1加密 0不加密
5. */
6. `public void loadDownloadInfo(long partnerId, int encryptType) {`
7. `loadDownloadInfo(partnerId, encryptType, false);`
8. }

3.下载记录支持多用户

下载支持多用户即每个用户的下载记录是独立、互不干扰的。上面初始化manager的同时可以调用setCacheFolderPath(String cacheFolderPath)给当前用户设置缓存文件路径。sdk默认缓存文件路径在传入的targetFolder/cache_info文件夹下。无需按用户区分下载记录，使用默认路径即可。

```

1. /**
2.  * 设置缓存路径
3.  * @param cacheFolderPath 缓存文件绝对路径。##请保证路径合法性##
4.  */
5. public void setCacheFolderPath(String cacheFolderPath){
6.     this.cacheFolder = cacheFolderPath;
7.     cacheManager.setCacheDir(cacheFolder);
8. }

```

切换用户请务必重新传入新的缓存路径。

4.新建一个下载任务

```

1. /**
2.  * 创建点播下载
3.  *
4.  * @param fileName 保存的下载的文件名称
5.  * @param videoid 视频id
6.  * @param token 视频token
7.  * @param definitions 下载的清晰度list,从前往后匹配
8.  * @param encryptType 加密类型 0 不加密, 1加密
9.  * @param extraInfo 额外信息, 客户自己定义, 这里只是转存
10. * @return
11. */
12. downloadManager.newDownloadTask("video_" + videoid, Long.parseLong(videoid), token, definitionList,
13.     0, "haha")
14. .observeOn(AndroidSchedulers.mainThread())
15. .subscribe(new Action1<DownloadTask>() {
16.     @Override
17.     public void call(DownloadTask task) {
18.         //直接开始下载
19.         task.start();
20.     }, new Action1<Throwable>() {
21.         @Override
22.         public void call(Throwable throwable) {
23.             throwable.printStackTrace();
24.         }
25.     });

```

5.DownloadTask接口说明

```

1. public interface DownloadTask {
2.
3.     /**
4.     * 开始任务
5.     */
6.     void start();
7.
8.     /**
9.     * 暂停任务
10.    */
11.    void pause();
12.
13.    /**

```

```
13. /
14. * 重新开始任务
15. */
16. void restart();
17.
18. /**
19. * 取消任务
20. */
21. void cancel();
22.
23. /**
24. * 删除任务和文件
25. */
26. void deleteFiles();
27.
28. /**
29. * 设置下载状态的监听
30. *
31. * @param listener
32. */
33. void setDownloadListener(DownloadListener listener);
34.
35. /**
36. * 获取视频相关信息，数据结构和DownloadInfo类似
37. *
38. * @return
39. */
40. DownloadModel getDownloadInfo();
41.
42. DownloadModel getVideoDownloadInfo();
43.
44. /**
45. * 获取下载状态
46. * New(0), // new -> downloading
47. * Downloading(1), // downloading -> pause, error, finish
48. * Pause(2), // pause -> downloading
49. * Error(3), // error -> new
50. * Finish(4), //
51. * Cancel(5);
52. * @return
53. */
54. TaskStatus getTaskStatus();
55.
56. /**
57. * 获取下载速度，单位为byte
58. * @return
59. */
60. long getSpeed();
61.
62. /**
63. * 获取下载进度
64. * @return
65. */
```

```
66. float getProgress();
67.
68. /**
69. * 获取要下载的总字节数
70. * @return
71. */
72. long getTotalLength();
73.
74. /**
75. * 获取已下载字节数
76. * @return
77. */
78. long getDownloadedLength();
79.
80. /**
81. * 获取下载类别
82. * @return
83. */
84. DownloadType getDownloadType();
85.
86. /**获取文件名称
87. * @return
88. */
89. String getFileName();
90.
91. /**
92. * 获取视频时长
93. * @return
94. */
95. long getVideoDuration();
96. }
```

6. 下载状态监听

给newDownloadTask()获得的DownloadTask设置DownloadListener

```

1. task.setDownloadListener(new DownloadListener() {
2.     @Override
3.     public void onProgress(DownloadTask task) {
4.         //TODO 更新下载进度
5.     }
6.
7.     @Override
8.     public void onError(final DownloadTask task, HttpException e) {
9.         holder.download.setText("出错");
10.        holder.download.setOnClickListener(new View.OnClickListener() {
11.            @Override
12.            public void onClick(View v) {
13.                task.start();
14.            }
15.        });
16.        e.printStackTrace();
17.        //下载地址已失效, 5103(token已失效)
18.        if(e.getCode() == 403 || e.getCode() == 5103){
19.            //TODO 需要用户传入新的token, 重新获取下载链接
20.            newDownloadTask(String.valueOf(task.getDownloadInfo().videoid), "test12345678");
21.        }
22.    }
23.
24.    @Override
25.    public void onPaused(final DownloadTask task) {
26.    }
27.
28.    @Override
29.    public void onStarted(final DownloadTask task) {
30.    }
31.
32.    @Override
33.    public void onFinish(DownloadTask task) {
34.        //TODO 更新UI
35.    }
36.
37.    @Override
38.    public void onDeleted(long videoid) {
39.        //TODO 更新UI
40.    }
41.    });

```

这里要特别注意的一点是OnError回调中, code==403的情况, 视频的下载链接存在一个有效期, 如果视频下载中断后视频链接过期则会返回403。上面的示例代码中onError已经给出了处理方案, 重新调用 `newDownloadTask(String videoid, String token)` 获取视频信息, 但这里的token需要集成者重新传入有效的token。

7.DownloadModel视频信息实体类

DownloadModel的字段和上面DownloadInfo列举的字段基本一致。


```

1. public long videoId; //vid
2. public long sessionId; //sessionId
3. public long roomId; //roomId
4. public String url; //url
5. public VideoDefinition definition; //清晰度
6. public long videoDuration; //视频长度（服务器没传就是0）
7. public FileType fileType; //file type
8. public String targetName; //保存的文件名
9. public String videoName; //视频标题
10. public TaskStatus status = TaskStatus.New; //当前状态
11. public String targetFolder; //保存文件夹
12. public long totalLength; //总大小
13. public long downloadLength; //已下载大小
14. public long speed; //下载瞬时速度
15. public Serializable data; //额外的数据
16. public int encryptType; //加密类型 0不加密，1加密
17. public String videoToken; //视频token，过期逻辑上层处理
18. public String extraInfo; //额外信息，用户想存啥就存啥
19. public String coverUrl; //课程封面url
20. public String lessonTime; //上课时间

```

8.demo中的[SimpleVideoDownloadActivity](#)简单集成了下载功能和UI效果，大家可以参考。

9.Android 8.0兼容

android 8.0之后不再支持应用后台创建service，videoplayer 1.7.13之后sdk把下载保活的DownloadService移出，需要用户自行实现后台下载保活，具体可以参考[CustomDownloadService](#)。之前的DownloadService依旧保留，不过不再兼容8.0系统。

数据迁移

从旧版本下载改用到新版本下载模块，需要兼容用户旧的下载记录，因为旧版本下载记录保存在数据库中，新版本下载记录保存在本地文件中。

1.RecoverDbHelper初始化

```

1. /**
2.  * 初始化
3.  * @param context
4.  * @param recoveryPath 旧的视频保存的文件夹
5.  * @param recoveryCallback callback
6.  */
7. public void init(Context context, String recoveryPath, IRecoveryCallback recoveryCallback){
8. }

```

2.开始迁移

```

1. RecoverDbHelper.getInstance().recoveryDbData();

```

3.示例代码

完整的代码如下，也可在demo中查看。

```

1. manager = DownloadService.getDownloadManager(getApplicationContext());
2. //设置缓存文件路径
3. manager.setTargetFolder(Environment.getExternalStorageDirectory().getAbsolutePath() +
    "/bb_video_downloaded/");
4. //TODO RecoverDbHelper为恢复旧版下载记录的工具类。没有从旧版迁移到新版的需求不用调用此工具类
5. RecoverDbHelper.getInstance().init(getApplicationContext(),
    Environment.getExternalStorageDirectory().getAbsolutePath() + "/aa_video_downloaded/",
6. new IRecoveryCallback() {
7. @Override
8. public void recoverySuccess() {
9. //重新获取taskList, true代表强制刷新
10. manager.loadDownloadInfo(32975272, 1, true);
11. adapter.notifyDataSetChanged();
12. }
13. });
14. //读取磁盘缓存的下载任务
15. manager.loadDownloadInfo(32975272, 1);
16. //TODO 这一句必须在manager.loadDownloadInfo()之后, 确保DownloadManager已初始化完毕
17. RecoverDbHelper.getInstance().recoveryDbData();

```

视频加密

在线播放和离线下载的视频支持自定义ev1格式(百家云自定义的视频格式)和mp4两种格式, 集成时可以选择是否使用加密格式, 服务器端没有对应的加密视频时默认返回mp4格式。之前通过file协议在java层对视频加解密的逻辑已去掉, 视频路径不要再拼接file://。如有视频加密需要请仔细阅读以下步骤。

1.后台配置

加密功能需要在上传视频时配置转码生成加密格式, 具体请联系销售同事配置。只有在后台配置生效之后上传的视频才有对应的加密格式的视频。

2.在线播放

```

1. //1代表使用加密视频,0反之
2. playerView.initPartner(partnerId,BJPlayerView.PLAYER_DEPLOY_ONLINE,1);

```

3.离线下载

```

1. //1代表使用加密视频,0反之
2. manager.loadDownloadInfo(partnerId, 1);

```

记忆播放

记忆播放: 下次打开视频会从上次观看结束的位置继续播放。

1.7.2版本之后才提供此项功能, 该功能默认关闭, 如需启用请看如下集成步骤。

1.开启记忆播放

```

1. playerView.setMemoryPlayEnable(true);

```

参数说明如下

```

1. /**
2. * 是否开启记忆播放功能, 默认关闭
3. * @param enable true开启
4. */
5. public void setMemoryPlayEnable(boolean enable)

```

2. 常用api

1. 清除播放记录

```
1. playerView.getMemoryPlayHelper().removeAll();
```

个性域名

直接对 `PlayerConstants.CUSTOM_DOMAIN` 赋值即可

纯音频

点播支持纯音频播放并提供占位图。对于有转码纯音频的点播，纯音频默认作为一种清晰度供用户手动切换。

1、起播即播放纯音频

```
1. playerView.setAudioFirstWhenEnable(true);
```

```
1. /**
2.  * 设置优先播纯音频
3.  */
4. public void setAudioFirstWhenEnable(boolean audioFirst){
5.     playAudioFirst = audioFirst;
6. }
```

2、更换占位图

如需更换纯音频占位图，直接对 `PlayerConstants.AUDIO_ON_PICTURE`赋值即可。

3、回调

OnPlayerViewListener下的onVideoDefinition

```
1. @Override
2. public void onVideoDefinition(BJPlayerView playerView, int definition) {
3.     //TODO: 设置清晰度完成
4.     if(definition == BJPlayerView.VIDEO_DEFINITION_AUDIO){
5.         //设置为纯音频
6.     }
7. }
```

3、下载

```
1. /**
2.  * true, 如果有音频文件则优先下载音频文件; false 下载视频文件
3.  *
4.  * @param downloadAudio
5.  */
6. manager.setDownloadAudio(isChecked);
```

本地日志

1.7.34版本之后新增关键日志记录到本地功能，默认保存最近7天的日志，日志文件保存目录为/sdcard/Android/packageName/bj_player_log。

1、常用api

```
1. /**
2.  * 设置本地日志文件路径(若设为空字符串则使用默认路径)
3.  */
4. public static void setLogDirPath(String path) {
5. }
```

```
1. /**
2.  * 获取本地日志文件路径
3.  */
4. public static String getLogDirPath()
```

```
1. /**
2.  * 设置保留日志的最大天数
3.  * @param recordDay
4.  */
5. public static void setMaxRecordDay(int recordDay)
```

压缩日志并返回路径

```
1. BJFileLog.zipLogFiles(new BJFileLog.IZipCallback() {
2. @Override
3. public void zipSuccess(String logPath) {
4. //TODO 这里可以上传zip包到服务端存储
5. {
6. //上传逻辑
7. }
8. //TODO 上传成功后可选择删除zip包
9. BJFileLog.deleteZipLogFiles();
10. }
11.
12. @Override
13. public void zipFailed(String errorMsg) {
14.
15. }
16. });
```

常用api说明

播放器(BJPlayerView类)

初始化合作方

```

1. public void initPartner(long partnerId) {
2. //正式服 && 默认获取加密视频
3. initPartner(partnerId, PLAYER_DEPLOY_ONLINE, 1);
4. }
5.
6. public void initPartner(long partnerId, int deploy) {
7. //默认获取加密视频
8. initPartner(partnerId, deploy, 1);
9. }
10. /**
11. * 设置 合作方 ID。 如果没有设置， 无法使用 SDK
12. *
13. * @param partnerId 合作方 ID
14. * @param deploy 运行环境;
15. * <ul>
16. * <li>测试环境: PLAYER_DEPLOY_DEBUG</li>
17. * <li>Beta 环境: PLAYER_DEPLOY_BETA</li>
18. * <li>线上环境: PLAYER_DEPLOY_ONLINE</li>
19. * </ul>
20. * @param encryptType 视频是否加密。1加密；0不加密
21. */
22. public void initPartner(long partnerId, int deploy, int encryptType)

```

设置视频填充颜色

```

1. /**
2. * 设置视频画面四周补边的颜色<br/>
3. *
4. * @param color 颜色值， 缺省值为Color.rgb(255, 0, 0, 0)
5. */
6. public void setVideoEdgePaddingColor(int color)
7.

```

设置片头片尾播放策略

```

1. /**
2. * 设置片头片尾的播放方式， 默认是不播
3. *
4. * @param headTailPlayMethod <ul>
5. * <li>不播: HEAD_TAIL_PLAY_NONE</li>
6. * <li>播一次: HEAD_TAIL_PLAY_ONCE</li>
7. * <li>每次都播: HEAD_TAIL_PLAY_EVERY</li>
8. * </ul>
9. */
10. public void setHeadTailPlayMethod(int headTailPlayMethod)

```

开始播放视频

```

1. /**
2. * <p>播放视频</p>
3. * 会检测网络状况
4. */
5. public synchronized void playVideo()

```

```
1. /**
2. * <p>播放视频</p>
3. * @param netTypeCheck, true检测网络, false不检测
4. */
5. public synchronized void playVideo(boolean netTypeCheck)
```

```
1. /**
2. * <p>播放视频</p>
3. * @param pos 从特定位置播放, 单位为秒
4. */
5. public synchronized void playVideo(int pos)
```

获取视频时长

```
1. /**
2. * 获取视频时长, 需在视频初始化
3. */
4. int getDuration()
```

设置用户自定义信息, 用户您的app的自定义统计

```
1. /**
2. * <p>设置用户自定义信息</p>
3. *
4. * @param userInfo 用户自定义消息
5. */
6. public void setUserInfo(String userName, int userNumber)
```

播放http(s)网络视频或者本地视频

```
1. /**
2. * <p>设置视频源路径</p>
3. *
4. * @param path 视频 url, 可以是网络视频, 也可以是本地视频路径
5. */
6. public void setVideoPath(String path)
```

获取当前视频信息

```
1. /**
2. * 获取视频详细信息
3. *
4. * @return VideoItem 视频信息model
5. */
6. @Override
7. public VideoItem getVideoItem()
```

播放器手势开关控制

```

1. /**
2.  * 是否允许通过触摸手势控制音量
3.  */
4. public void enableVolumeGesture(boolean enable)
5.
6. /**
7.  * 是否允许通过触摸手势控制亮度
8.  */
9. public void enableBrightnessGesture(boolean enable)
10.
11. /**
12.  * 是否允许通过触摸手势拖拽
13.  */
14. public void enableSeekGesture(boolean enable)

```

是否响应横竖屏切换

```

1. //true:横竖屏切换时播放器ui不变化
2. public void setForbidConfiguration(boolean forbidConfiguration)

```

设置视频清晰度

```

1. /**
2.  * <p>设置播放视频清晰度</p>
3.  * @param definition 清晰度
4.  * <ul>
5.  * <li>标清(0): VIDEO_DEFINITION_STD</li>
6.  * <li>高清(1): VIDEO_DEFINITION_HIGH</li>
7.  * <li>超清(2): VIDEO_DEFINITION_SUPER</li>
8.  * <li>720P(3): VIDEO_DEFINITION_720p</li>
9.  * <li>1080P(4): VIDEO_DEFINITION_1080p</li>
10.  * </ul>
11.  */
12. @Override
13. public void setVideoDefinition(int definition)

```

设置播放倍速(标记废弃)

```

1. /**
2.  * <p>设置视频播放倍速</p>
3.  * @param rate 播放速率
4.  * <ul>
5.  * <li>1 倍正常速率: VIDEO_RATE_1_X</li>
6.  * <li>1.1 倍正常速率: VIDEO_RATE_1_1_X</li>
7.  * <li>1.2 倍正常速率: VIDEO_RATE_1_2_X</li>
8.  * <li>1.5 倍速率: VIDEO_RATE_1_5_X</li>
9.  * <li>1.8 倍速率: VIDEO_RATE_1_8_X</li>
10.  * <li>2 倍速率: VIDEO_RATE_2_X</li>
11.  * </ul>
12.  */
13. @Override
14. @Deprecated
15. public void setVideoRate(int rate)

```

设置任意播放倍速【0.5~2.0】

```
1. /**
2. * 设置任意倍速，范围[0.5,2.0].不在此范围则不生效
3. * @param speed
4. */
5. public void setVideoRate(float speed)
```

关闭网络监听

```
1. /**
2. * <p>设置是否开启网络监听</p>
3. *
4. * @param enableNetWatcher true监听 false 不监听
5. */
6. @Override
7. public void setEnableNetWatcher(boolean enableNetWatcher)
```

DownloadManager(下载管理类)

1.读取下载记录缓存

```
1. /**
2. * 从cache缓存中读取下载记录并设置加密参数
3. * @param partnerId
4. * @param encryptType 1加密 0不加密
5. */
6. public void loadDownloadInfo(long partnerId, int encryptType) {
7. loadDownloadInfo(partnerId, encryptType, false);
8. }
9.
```

2.创建点播下载

```
1. public Observable<DownloadTask> newDownloadTask(final String fileName, final long videoId, final String token,final List<VideoDefinition> definitions, final int encryptType, final String extraInfo)
```

参数说明见上方，[点我直达](#)

3.删除任务

```
1. /**
2. * 删除任务 并同时删除文件
3. *
4. * @param task
5. */
6. public void deleteTask(DownloadTask task)
```

4.获取下载列表

```
1. public List<DownloadTask> getAllTasks()
```

错误码对照

错误码	说明
-10000	ijkplayer内部错误,比如文件异常

错误码	说明
-1	无网络连接
-2	移动网络播放
5101、5102、5103	token异常
403	视频地址过期

注：ijkplayer没有给出具体错误码，详见源码的头文件ijkplayer_android_def.h
其余错误码可参考AndroidSDK的MediaPlayer（需科学上网）

常见问题

1) 为什么播本地视频总是提示-10000错误?

-10000错误码为ijkplayer底层报出的错误，请检查本地文件是否为合法视频文件，尝试使用系统播放器能不能正常播放。

2) 为什么我集成之后播放视频一直在loading，没有播放出来?

首先检查网络连接是否正常，再者视频渲染采用surfaceview，初始化需要一定的时间，可在调用播放视频方法的时候尝试加入500ms左右延时。

3) 为什么一直提示我token解析失败?

一个视频id对应一个token，token是百家云后台生成的，您可以联系后台开发人员确定token是否是正确的，再者查看初始化播放器的时候部署环境是否为在线环境playerView.initPartner(123456L, BJPlayerView.PLAYER_DEPLOY_ONLINE);

4) 我不想xml中添加控件，可不可以使用java代码生成播放器?

可以的。BJPlayerView继承自FrameLayout, FrameLayout所有的特性都是支持的。

5) 播放器提示500，什么原因导致?

播放器报500错误码，可能出现的原因比较多，比如connect timeout IO Error等都会导致播放器提示500。这种情况下重试播放即可恢复正常。

6) 为什么某些机器上音视频不同步?

请确认同时引入armeabi-v7a和armeabi两种so库。