```java
package com.catdog2025.activity;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.FrameLayout;
import android.widget.ImageView;
import android.widget.Toast;
import androidx.annotation.Nullable;

import com.bytedance.sdk.openadsdk.AdSlot;
import com.bytedance.sdk.openadsdk.TTAdConstant;
import com.bytedance.sdk.openadsdk.TTAdDislike;
import com.bytedance.sdk.openadsdk.TTAdNative;
import com.bytedance.sdk.openadsdk.TTNativeExpressAd;
import com.bytedance.sdk.openadsdk.TTRewardVideoAd;
import com.catdog2025.R;
import com.catdog2025.config.TTAdManagerHolder;
import com.catdog2025.utils.UIUtils;

import java.util.List;

/**
 * Cat和Dog图片展示页面
 * 显示上方的cat图片和下方的dog图片
 */
public class CatDogActivity extends Activity {
    private static final String TAG = "CatDogActivity";

    private ImageView mCatImageView;
    private ImageView mDogImageView;

    // Cat Banner广告相关
    private FrameLayout mCatBannerContainer;
    private TTNativeExpressAd mCatBannerAd;
    private TTAdNative.NativeExpressAdListener mCatBannerListener;
    private TTNativeExpressAd.ExpressAdInteractionListener mCatBannerInteractionListener;
    private TTAdDislike.DislikeInteractionCallback mCatDislikeCallback;

    // Dog Banner广告相关
    private FrameLayout mDogBannerContainer;
    private TTNativeExpressAd mDogBannerAd;
    private TTAdNative.NativeExpressAdListener mDogBannerListener;
    private TTNativeExpressAd.ExpressAdInteractionListener mDogBannerInteractionListener;
    private TTAdDislike.DislikeInteractionCallback mDogDislikeCallback;

    // 激励广告相关
    private TTRewardVideoAd mTTRewardVideoAd;
    private TTAdNative.RewardVideoAdListener mRewardVideoListener;
    private TTRewardVideoAd.RewardAdInteractionListener mRewardVideoAdInteractionListener;
    private String mPendingPetType; // 等待跳转的宠物类型

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Log.d(TAG, "onCreate: 启动CatDogActivity");
        setContentView(R.layout.activity_cat_dog);

        // 初始化视图
        initViews();

        Log.d(TAG, "onCreate: CatDogActivity初始化完成");

        // 记录激励广告配置信息
        String rewardId = getResources().getString(R.string.reward_media_id);
        Log.d(TAG, "□ 激励广告配置信息:");
        Log.d(TAG, "   广告位ID: " + rewardId);
        Log.d(TAG, "   SDK初始化状态: " + (TTAdManagerHolder.get() != null ? "已初始化" : "未初始化"));
        if (TTAdManagerHolder.get() != null) {
            Log.d(TAG, "   SDK版本: " + TTAdManagerHolder.get().getSDKVersion());
        }
    }

    /**
     * 初始化视图组件
     */
    private void initViews() {
        mCatImageView = findViewById(R.id.iv_cat);
        mDogImageView = findViewById(R.id.iv_dog);

        // 初始化Banner广告容器
        mCatBannerContainer = findViewById(R.id.cat_banner_container);
        mDogBannerContainer = findViewById(R.id.dog_banner_container);

        // 设置图片资源
        mCatImageView.setImageResource(R.drawable.cat);
        mDogImageView.setImageResource(R.drawable.dog);

        // 设置图片点击事件
        mCatImageView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Log.d(TAG, "Cat图片被点击，显示激励广告确认对话框");
                showRewardAdConfirmDialog("cat");
            }
        });

        mDogImageView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Log.d(TAG, "Dog图片被点击，显示激励广告确认对话框");
                showRewardAdConfirmDialog("dog");
            }
        });

        // 初始化广告监听器
        initAdListeners();

        // 加载Banner广告
        loadCatBannerAd();
        loadDogBannerAd();

        Log.d(TAG, "initViews: 图片设置完成，开始加载Banner广告");
    }

    @Override
    protected void onResume() {
        super.onResume();
        Log.d(TAG, "onResume: 页面恢复");
    }

    @Override
    protected void onPause() {
        super.onPause();
        Log.d(TAG, "onPause: 页面暂停");
    }

    /**
     * 初始化广告监听器
     */
    private void initAdListeners() {
        // 猫咪Banner广告监听器
        mCatBannerListener = new TTAdNative.
NativeExpressAdListener() {
            @Override
            public void onError(int code, String msg) {
                Log.d(TAG, "Cat banner load fail: errCode: " + code + ",
errMsg: " + msg);
            }

            @Override
            public void onNativeExpressAdLoad(List<
TTNativeExpressAd> list) {
                if (list != null && list.size() > 0) {
                    Log.d(TAG, "Cat banner load success");
                    mCatBannerAd = list.get(0);
                    showCatBannerAd();
                } else {
                    Log.d(TAG, "Cat banner load success, but list is null");
                }
            }
        };

        mCatBannerInteractionListener = new TTNativeExpressAd.
ExpressAdInteractionListener() {
            @Override
            public void onAdClicked(View view, int type) {
                Log.d(TAG, "Cat banner clicked");
            }

            @Override
            public void onAdShow(View view, int type) {
                Log.d(TAG, "Cat banner showed");
            }

            @Override
            public void onRenderFail(View view, String msg, int code) {
                Log.d(TAG, "Cat banner render fail: " + msg);
            }

            @Override
            public void onRenderSuccess(View view, float width, float
height) {
                Log.d(TAG, "Cat banner render success");
            }
        };

        mCatDislikeCallback = new TTAdDislike.DislikeInteractionCallback() {
            @Override
            public void onShow() { }

            @Override
            public void onSelected(int position, String value, boolean enforce) {
                if (mCatBannerContainer != null) {
                    mCatBannerContainer.removeAllViews();
                }
                Log.d(TAG, "Cat banner closed");
            }

            @Override
            public void onCancel() { }
        };

        // 狗狗Banner广告监听器
        mDogBannerListener = new TTAdNative.NativeExpressAdListener() {
            @Override
            public void onError(int code, String msg) {
                Log.d(TAG, "Dog banner load fail: errCode: " + code + ", errMsg: " + msg);
            }

            @Override
            public void onNativeExpressAdLoad(List<TTNativeExpressAd> list) {
                if (list != null && list.size() > 0) {
                    Log.d(TAG, "Dog banner load success");
                    mDogBannerAd = list.get(0);
                    showDogBannerAd();
                } else {
                    Log.d(TAG, "Dog banner load success, but list is null");
                }
            }
        };

        mDogBannerInteractionListener = new TTNativeExpressAd.ExpressAdInteractionListener() {
            @Override
            public void onAdClicked(View view, int type) {
                Log.d(TAG, "Dog banner clicked");
            }

            @Override
            public void onAdShow(View view, int type) {
                Log.d(TAG, "Dog banner showed");
            }

            @Override
            public void onRenderFail(View view, String msg, int code) {
                Log.d(TAG, "Dog banner render fail: " + msg);
            }

            @Override
            public void onRenderSuccess(View view, float width, float height) {
                Log.d(TAG, "Dog banner render success");
            }
        };

        mDogDislikeCallback = new TTAdDislike.DislikeInteractionCallback() {
            @Override
            public void onShow() { }

            @Override
            public void onSelected(int position, String value, boolean enforce) {
                if (mDogBannerContainer != null) {
                    mDogBannerContainer.removeAllViews();
                }
                Log.d(TAG, "Dog banner closed");
            }

            @Override
            public void onCancel() { }
        };
    }

    /**
     * 加载猫咪Banner广告
     */
    private void loadCatBannerAd() {
        String catBannerId = getResources().getString(R.string.cat_banner_media_id);
        // 创建AdSlot对象
        AdSlot adSlot = new AdSlot.Builder()
                .setCodeId(catBannerId) // 使用Cat专用Banner广告位ID
                .setImageAcceptedSize(UIUtils.dp2px(this, 350f), UIUtils.dp2px(this, 120f))
                .build();

        // 创建TTAdNative对象
        TTAdNative adNativeLoader = TTAdManagerHolder.get().createAdNative(this);

        // 加载广告
        if (adNativeLoader != null) {
            adNativeLoader.loadBannerExpressAd(adSlot, mCatBannerListener);
            Log.d(TAG, "开始加载Cat Banner广告，广告位ID: " + catBannerId);
        }
    }

    /**
     * 加载狗狗Banner广告
     */
    private void loadDogBannerAd() {
        String dogBannerId = getResources().getString(R.string.dog_banner_media_id);
        // 创建AdSlot对象
        AdSlot adSlot = new AdSlot.Builder()
                .setCodeId(dogBannerId) // 使用Dog专用Banner广告位ID
                .setImageAcceptedSize(UIUtils.dp2px(this, 350f), UIUtils.dp2px(this, 120f))
                .build();

        // 创建TTAdNative对象
        TTAdNative adNativeLoader = TTAdManagerHolder.get().createAdNative(this);

        // 加载广告
        if (adNativeLoader != null) {
            adNativeLoader.loadBannerExpressAd(adSlot, mDogBannerListener);
            Log.d(TAG, "开始加载Dog Banner广告，广告位ID: " + dogBannerId);
        }
    }

    /**
     * 显示猫咪Banner广告
     */
    private void showCatBannerAd() {
        if (mCatBannerAd != null && mCatBannerContainer != null) {
            mCatBannerAd.setExpressInteractionListener(mCatBannerInteractionListener);
            mCatBannerAd.setDislikeCallback(this, mCatDislikeCallback);

            View bannerView = mCatBannerAd.getExpressAdView();
            if (bannerView != null) {
                mCatBannerContainer.removeAllViews();
                mCatBannerContainer.addView(bannerView);
                Log.d(TAG, "Cat Banner广告显示成功");
            }
        }
    }

    /**
```

```java
     * 显示狗狗Banner广告
     */
    private void showDogBannerAd() {
        if (mDogBannerAd != null && mDogBannerContainer != null) {
            mDogBannerAd.setExpressInteractionListener(mDogBannerInteractionListener);
            mDogBannerAd.setDislikeCallback(this, mDogDislikeCallback);

            View bannerView = mDogBannerAd.getExpressAdView();
            if (bannerView != null) {
                mDogBannerContainer.removeAllViews();
                mDogBannerContainer.addView(bannerView);
                Log.d(TAG, "Dog Banner广告显示成功");
            }
        }
    }

    /**
     * 显示激励广告确认对话框
     * @param petType 宠物类型
     */
    private void showRewardAdConfirmDialog(String petType) {
        Log.d(TAG, "□ 显示激励广告确认对话框");
        Log.d(TAG, "   宠物类型: " + petType);

        mPendingPetType = petType;
        String petName = "cat".equals(petType) ? "小猫" : "小狗";

        Log.d(TAG, "   宠物名称: " + petName);

        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("□ 解锁" + petName + "沟通器");
        builder.setMessage("观看一个短视频广告即可解锁" + petName + "沟通功能，与您的宠物开始对话吧！");
        builder.setIcon("cat".equals(petType) ? R.drawable.cat : R.drawable.dog);

        builder.setPositiveButton("观看广告", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Log.d(TAG, "□ 用户点击'观看广告'按钮");
                Log.d(TAG, "   即将开始加载激励广告...");
                loadAndShowRewardAd();
            }
        });

        builder.setNegativeButton("稍后再说", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Log.d(TAG, "□ 用户点击'稍后再说'按钮");
                Log.d(TAG, "   用户取消观看激励广告");
                Toast.makeText(CatDogActivity.this, "需要观看广告才能使用沟通功能哦", Toast.LENGTH_SHORT).show();
            }
        });

        builder.setCancelable(false); // 不允许点击外部取消
        AlertDialog dialog = builder.create();
        dialog.show();

        Log.d(TAG, "□ 激励广告确认对话框已显示");
    }

    /**
     * 显示激励广告重试对话框
     * @param message 提示信息
     */
    private void showRewardAdRetryDialog(String message) {
        Log.d(TAG, "□ 显示激励广告重试对话框");

        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("广告加载失败");
        builder.setMessage(message + "\n\n• 重试：重新加载广告\n• 取消：返回主页面");
        builder.setCancelable(false);

        //重试按钮
        builder.setPositiveButton("重试", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Log.d(TAG, "□ 用户选择重试广告");
                dialog.dismiss();
                //重新加载广告
                loadAndShowRewardAd();
            }
        });

        //取消按钮（返回主页面，保护广告收入）
        builder.setNegativeButton("取消", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Log.d(TAG, "□ 用户取消广告，返回主页面（保护广告收入）");
                dialog.dismiss();
                Toast.makeText(CatDogActivity.this, "取消广告，请稍后再试", Toast.LENGTH_SHORT).show();
                //清空待处理状态，返回主页面
                mPendingPetType = null;
            }
        });

        AlertDialog dialog = builder.create();
        dialog.show();

        Log.d(TAG, "□ 激励广告重试对话框已显示");
    }

    /**
     * 显示激励广告失败选择对话框
     */
    private void showRewardAdFailDialog() {
```

```java
        Log.d(TAG, "□ 显示激励广告失败选择对话框");

        String petName = "cat".equals(mPendingPetType) ? "小猫" : "小狗";

        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("广告暂时无法加载");
        builder.setMessage("抱歉，当前暂时无法加载广告\n\n您可以选择：\n• 重新尝试加载广告\n• 稍后再试（返回主页面）");
        builder.setCancelable(false);

        // 重试按钮
        builder.setPositiveButton("重试", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Log.d(TAG, "□ 用户选择重试广告（无广告可用情况）");
                dialog.dismiss();
                loadAndShowRewardAd();
            }
        });

        // 稍后再试按钮（保护广告收入）
        builder.setNegativeButton("稍后再试", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Log.d(TAG, "□ 用户选择稍后再试（保护广告收入）");
                dialog.dismiss();
                //Toast.makeText(CatDogActivity.this, "请稍后再试观看广告解锁功能", Toast.LENGTH_SHORT).show();
                // 清空待处理状态，不解锁功能
                mPendingPetType = null;
            }
        });

        AlertDialog dialog = builder.create();
        dialog.show();

        Log.d(TAG, "□ 激励广告失败选择对话框已显示");
    }

    /**
     * 加载并展示激励广告
     */
    private void loadAndShowRewardAd() {
        Log.d(TAG, "=== 开始激励广告流程 ===");

        // 检查SDK初始化状态
        if (TTAdManagerHolder.get() == null) {
            Log.e(TAG, "TTAdManagerHolder为空，SDK未正确初始化");
            Toast.makeText(this, "广告SDK未初始化，请重启应用", Toast.LENGTH_LONG).show();
            return;
        }
        Log.d(TAG, "□ TTAdManagerHolder已初始化");

        // 初始化监听器
        initRewardAdListeners();
        Log.d(TAG, "□ 激励广告监听器已初始化");

        // 创建AdSlot对象
        String rewardId = getResources().getString(R.string.reward_media_id);
        Log.d(TAG, "□ 获取激励广告位ID: " + rewardId);

        AdSlot adSlot = new AdSlot.Builder()
                .setCodeId(rewardId)
                .setOrientation(TTAdConstant.ORIENTATION_VERTICAL)
                .build();
        Log.d(TAG, "□ AdSlot对象创建成功，方向: 竖屏");

        // 创建TTAdNative对象
        TTAdNative adNativeLoader = TTAdManagerHolder.get().createAdNative(this);
        if (adNativeLoader == null) {
            Log.e(TAG, "□ createAdNative返回null，无法创建广告加载器");
            Toast.makeText(this, "广告加载器创建失败", Toast.LENGTH_LONG).show();
            return;
        }
        Log.d(TAG, "□ TTAdNative广告加载器创建成功");

        // 加载激励广告
        Log.d(TAG, "□ 开始加载激励广告...");
        adNativeLoader.loadRewardVideoAd(adSlot, mRewardVideoListener);
        //Toast.makeText(this, "正在加载广告...", Toast.LENGTH_SHORT).show();
        Log.d(TAG, "□ 激励广告加载请求已发送，等待回调...");
    }

    /**
     * 初始化激励广告监听器
     */
    private void initRewardAdListeners() {
        Log.d(TAG, "□ 开始初始化激励广告监听器...");

        // 广告加载监听器
        mRewardVideoListener = new TTAdNative.RewardVideoAdListener() {
            @Override
            public void onError(int code, String msg) {
                Log.e(TAG, "□ 激励广告加载失败详情:");
                Log.e(TAG, "   错误码: " + code);
                Log.e(TAG, "   错误信息: " + msg);
                Log.e(TAG, "   广告位ID: " + getResources().getString(R.string.reward_media_id));
                Log.e(TAG, "   宠物类型: " + mPendingPetType);

                // 分析错误原因
                analyzeRewardAdError(code, msg);

                // 检查是否有本地配置可用
                if (TTAdManagerHolder.isLocalConfigSupported()) {
                    Log.i(TAG, "□□ 检测到本地配置支持，
可能从本地配置重试");
                showRewardAdRetryDialog("广告加载失败，是否重试？");
            } else {
                //Log.w(TAG, "□□ 无本地配置支持，提供用户选择");
                // 显示用户选择对话框，而不是直接跳转
                showRewardAdFailDialog();
            }
        }

        @Override
        public void onRewardVideoAdLoad(TTRewardVideoAd ttRewardVideoAd) {
            Log.d(TAG, "□ 激励广告加载成功！");
            Log.d(TAG, "   广告对象: " + (ttRewardVideoAd != null ? "不为空" : "为空"));
            if (ttRewardVideoAd != null) {
                Log.d(TAG, "   广告类型: 激励视频");
                Log.d(TAG, "   是否已缓存: " + (ttRewardVideoAd.getMediationManager() != null));
            }
            mTTRewardVideoAd = ttRewardVideoAd;
            // 加载成功后立即展示
            Log.d(TAG, "□ 准备展示激励广告...");
            showRewardAd();
        }

        @Override
        public void onRewardVideoCached() {
            Log.d(TAG, "□ 激励广告缓存成功（方法1）");
        }

        @Override
        public void onRewardVideoCached(TTRewardVideoAd ttRewardVideoAd) {
            Log.d(TAG, "□ 激励广告缓存成功（方法2）");
            Log.d(TAG, "   缓存的广告对象: " + (ttRewardVideoAd != null ? "不为空" : "为空"));
            mTTRewardVideoAd = ttRewardVideoAd;
            }
        };

        // 广告展示监听器
        mRewardVideoAdInteractionListener = new TTRewardVideoAd.RewardAdInteractionListener() {
            @Override
            public void onAdShow() {
                Log.d(TAG, "□ 激励广告开始展示");
                Log.d(TAG, "   当前时间: " + System.currentTimeMillis());
                Log.d(TAG, "   宠物类型: " + mPendingPetType);
            }

            @Override
            public void onAdVideoBarClick() {
                Log.d(TAG, "□ 激励广告被点击");
            }

            @Override
            public void onAdClose() {
                Log.d(TAG, "□ 激励广告关闭");
                Log.d(TAG, "   用户可能未看完广告");
            }

            @Override
            public void onVideoComplete() {
                Log.d(TAG, "□ 激励广告视频播放完成");
                Log.d(TAG, "   用户已观看完整个视频");
            }

            @Override
            public void onVideoError() {
                Log.e(TAG, "□ 激励广告视频播放错误");
                Log.e(TAG, "   可能是网络问题或视频文件损坏");
                Toast.makeText(CatDogActivity.this, "视频播放出错", Toast.LENGTH_SHORT).show();
            }

            @Override
        public void onRewardVerify(boolean rewardVerify, int rewardAmount, String rewardName, int errorCode, String errorMsg) {
            Log.d(TAG, "□ 激励广告奖励验证");
            Log.d(TAG, "   验证结果: " + rewardVerify);
            Log.d(TAG, "   奖励数量: " + rewardAmount);
            Log.d(TAG, "   奖励名称: " + rewardName);
            Log.d(TAG, "   错误码: " + errorCode);
            Log.d(TAG, "   错误信息: " + errorMsg);
        }

        @Override
        public void onRewardArrived(boolean isRewardValid, int rewardType, Bundle extraInfo) {
            Log.d(TAG, "□ 激励广告奖励到达");
            Log.d(TAG, "   奖励有效: " + isRewardValid);
            Log.d(TAG, "   奖励类型: " + rewardType);
            Log.d(TAG, "   额外信息: " + (extraInfo != null ? extraInfo.toString() : "无"));
            Log.d(TAG, "   目标宠物: " + mPendingPetType);

            if (isRewardValid) {
                // 奖励验证成功，跳转到录音界面
                String petName = "cat".equals(mPendingPetType) ? "小猫" : "小狗";
                Log.d(TAG, "□ 奖励验证成功，准备跳转到" + petName + "录音界面");
                Toast.makeText(CatDogActivity.this, "广告观看完成！解锁" + petName + "沟通器成功！", Toast.LENGTH_LONG).show();
                startRecordPlayActivity(mPendingPetType);
            } else {
                Log.w(TAG, "□□ 奖励验证失败，用户可能未完整观看广告");
                Toast.makeText(CatDogActivity.this, "奖励验证失败，请重新尝试", Toast.LENGTH_SHORT).show();
            }
        }

        @Override
```

```
            public void onSkippedVideo() {
                Log.w(TAG, "□□ 激励广告被跳过");
                Log.w(TAG, "  用户未看完整个广告，无法获得奖励");
                Toast.makeText(CatDogActivity.this, "需要看完整个广告才能解锁功能", Toast.LENGTH_SHORT).show();
            }
        };

        Log.d(TAG, "□ 激励广告监听器初始化完成");
    }

    /**
     * 分析激励广告加载错误
     * @param code 错误代码
     * @param msg 错误信息
     */
    private void analyzeRewardAdError(int code, String msg) {
        Log.d(TAG, "□ 激励广告错误分析:");

        // 常见错误代码分析
        switch (code) {
            case 40001:
                Log.d(TAG, "  原因: 配置拉取失败（网络问题或服务器异常）");
                Log.d(TAG, "  建议: 本地配置和兜底机制将发挥作用");
                break;
            case 40002:
                Log.d(TAG, "  原因: 广告位配置错误");
                Log.d(TAG, "  建议: 检查广告位ID是否正确");
                break;
            case 40003:
                Log.d(TAG, "  原因: 无广告填充");
                Log.d(TAG, "  建议: 正常现象，可启用降级策略");
                break;
            case 40004:
                Log.d(TAG, "  原因: 广告已被过滤");
                Log.d(TAG, "  建议: 内容安全过滤，可重试");
                break;
            case 20005:
                Log.d(TAG, "  原因: 全部代码位请求失败（聚合广告位无可用资源）");
                Log.d(TAG, "  说明: 所有聚合的广告平台都无法返回广告");
                break;
            case -8:
                Log.d(TAG, "  原因: 网络超时");
                Log.d(TAG, "  建议: 检查网络连接或使用本地配置");
                break;
            case -9:
                Log.d(TAG, "  原因: 网络错误");
                Log.d(TAG, "  建议: 本地配置将提供保障");
                break;
            default:
                Log.d(TAG, "  原因: 其他错误 (" + code + ")");
                Log.d(TAG, "  描述: " + msg);
                break;
        }

        // 兜底机制状态检查
        boolean localConfigSupported = TTAdManagerHolder.isLocalConfigSupported();
        boolean fallbackSupported = TTAdManagerHolder.isFallbackSupported();

        Log.d(TAG, "  本地配置: " + (localConfigSupported ? "□ 可用" : "□ 不可用"));
        Log.d(TAG, "  自定义兜底: " + (fallbackSupported ? "□ 可用" : "□ 不可用"));

        if (!localConfigSupported && !fallbackSupported) {
            Log.i(TAG, "  说明: 当前为标准SDK，基础功能完全正常");
        }
    }

    /**
     * 展示激励广告
     */
    private void showRewardAd() {
        Log.d(TAG, "□ === 开始展示激励广告 ===");

        if (mTTRewardVideoAd == null) {
            Log.e(TAG, "□ 激励广告对象为空，无法展示");
            Log.e(TAG, "   可能原因：广告加载失败或尚未完成");
            Toast.makeText(this, "广告加载失败，请重试", Toast.LENGTH_SHORT).show();
            return;
        }

        Log.d(TAG, "□ 激励广告对象检查通过");
        Log.d(TAG, "   广告对象状态: 不为空");
        Log.d(TAG, "   MediationManager: " + (mTTRewardVideoAd.getMediationManager() != null ? "已创建" : "未创建"));

        // 设置展示监听器并展示广告
        Log.d(TAG, "□ 设置广告展示监听器...");
        mTTRewardVideoAd.setRewardAdInteractionListener(mRewardVideoAdInteractionListener);

        Log.d(TAG, "□ 准备调用showRewardVideoAd...");
        Log.d(TAG, "   Activity状态: " + (this.isFinishing() ? "正在结束" : "正常"));
        Log.d(TAG, "   宠物类型: " + mPendingPetType);

        try {
            mTTRewardVideoAd.showRewardVideoAd(this);
            Log.d(TAG, "□ showRewardVideoAd调用成功，等待广告展示...");
        } catch (Exception e) {
            Log.e(TAG, "□ showRewardVideoAd调用异常: " + e.getMessage());
            e.printStackTrace();
            Toast.makeText(this, "广告展示失败: " + e.getMessage(), Toast.LENGTH_LONG).show();
        }
    }

    /**
     * 启动录音播放界面
     * @param petType 宠物类型：\"cat\" 或 \"dog\"
     */
    private void startRecordPlayActivity(String petType) {
        Intent intent = new Intent(this, RecordPlayActivity.class);
        intent.putExtra("pet_type", petType);
        startActivity(intent);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();

        // 销毁Banner广告
        if (mCatBannerAd != null) {
            mCatBannerAd.destroy();
            Log.d(TAG, "Cat Banner广告已销毁");
        }
        if (mDogBannerAd != null) {
            mDogBannerAd.destroy();
            Log.d(TAG, "Dog Banner广告已销毁");
        }

        // 销毁激励广告
        if (mTTRewardVideoAd != null && mTTRewardVideoAd.
getMediationManager() != null) {
            mTTRewardVideoAd.getMediationManager().destroy();
            Log.d(TAG, "激励广告已销毁");
        }

        Log.d(TAG, "onDestroy: 页面销毁");
    }
}

```