```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:background="@color/white"
    android:fitsSystemWindows="true"
    android:orientation="vertical">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <include layout="@layout/activity_back_header_right_img"></include>

    </RelativeLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:paddingLeft="5dp"
        android:paddingTop="10dp"
        android:paddingRight="5dp"
        android:paddingBottom="10dp">

        <TextView
            android:id="@+id/message_list_name_type"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="5dp"
            android:background="@drawable/module_patient_type_round_blue"
            android:gravity="center"
            android:text="咨"
            android:textColor="@color/module_item_main_blue"
            android:textSize="15sp" />

        <Button
            android:id="@+id/message_entrust"
            android:layout_width="wrap_content"
            android:layout_height="30dp"
            android:layout_marginLeft="5dp"
            android:layout_marginRight="5dp"
            android:background="@drawable/btn_send_selector"
            android:gravity="center"
            android:stateListAnimator="@null"
            android:text="委托管理"
            android:textAllCaps="false"
            android:textColor="@color/white"
            android:visibility="visible" />

        <TextView
            android:id="@+id/message_list_restrict"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginRight="10dp"
```

```
        android:layout_weight="1"
        android:gravity="right"
        android:text="剩余5小时25分钟"
        android:textColor="@color/blue"
        android:textSize="14sp" />

</LinearLayout>

<include layout="@layout/view_line" />

<com.lcodecore.tkrefreshlayout.TwinklingRefreshLayout
    android:id="@+id/message_layout_refresh"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1">

    <com.yanzhenjie.recyclerview.SwipeRecyclerView
        android:id="@+id/message_layout_recycler"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:overScrollMode="never"></com.yanzhenjie.recyclerview.SwipeRecyclerView>

</com.lcodecore.tkrefreshlayout.TwinklingRefreshLayout>

<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <LinearLayout
        android:id="@+id/ll_voice"

        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom"
        android:background="#e7e7e7"
        android:gravity="bottom"
        android:orientation="vertical">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <ImageView
                android:id="@+id/iv_voice"

                android:layout_width="40dp"
                android:layout_height="40dp"
                android:layout_margin="@dimen/dp5"
                android:src="@mipmap/voice" />

            <EditText
                android:id="@+id/et_msg"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
```

```
        android:layout_gravity="center_vertical"
        android:layout_marginLeft="@dimen/dp5"
        android:layout_marginTop="@dimen/dp5"
        android:layout_marginRight="@dimen/dp10"
        android:layout_marginBottom="@dimen/dp5"
        android:layout_weight="1"
        android:background="@drawable/chat_msg_shape"
        android:hint="消息"
        android:maxLines="5"
        android:padding="5dp" />

    <Button
        android:id="@+id/btn_send"
        android:layout_width="65dp"
        android:layout_height="40dp"
        android:layout_gravity="center_vertical"
        android:layout_marginTop="5dp"
        android:layout_marginRight="5dp"
        android:layout_marginBottom="5dp"
        android:background="@drawable/btn_send_selector"
        android:gravity="center"
        android:text="@string/send"
        android:textAllCaps="false"
        android:textColor="@color/white"
        android:textSize="18sp" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:paddingLeft="@dimen/dp5"
    android:paddingTop="@dimen/dp5"
    android:paddingRight="@dimen/dp5"
    android:paddingBottom="@dimen/dp10">

    <ImageView
        android:id="@+id/iv_camera"
        android:layout_width="0dp"
        android:layout_height="@dimen/chat_icon_size"
        android:layout_gravity="center_vertical"
        android:layout_weight="1"
        android:src="@mipmap/chat_camera" />

    <ImageView
        android:id="@+id/iv_pic"
        android:layout_width="0dp"
        android:layout_height="@dimen/chat_icon_size"
        android:layout_gravity="center_vertical"
        android:layout_weight="1"
        android:src="@mipmap/chat_pic" />


</LinearLayout>
```

```
</LinearLayout>

<LinearLayout
    android:id="@+id/ll_message"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom"
    android:background="#e7e7e7"
    android:gravity="bottom"
    android:orientation="vertical"
    android:visibility="gone">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_vertical"
        android:orientation="horizontal">

        <ImageView
            android:id="@+id/iv_message"

            android:layout_width="40dp"
            android:layout_height="40dp"
            android:layout_margin="@dimen/dp5"
            android:src="@mipmap/keyboard" />

        <com.palline.merit.frame.voice.view.AudioRecorderButton
            android:id="@+id/audio_btn"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center_vertical"
            android:layout_marginLeft="@dimen/dp5"
            android:layout_marginTop="@dimen/dp5"
            android:layout_marginRight="@dimen/dp10"
            android:layout_marginBottom="@dimen/dp5"
            android:layout_weight="1"
            android:background="@drawable/btn_recorder_normal"
            android:enabled="true"
            android:gravity="center"
            android:minHeight="0dp"
            android:padding="5dp"
            android:text="@string/str_recorder_normal"
            android:textColor="#727272" />


    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:paddingLeft="@dimen/dp5"
        android:paddingTop="@dimen/dp5"
        android:paddingRight="@dimen/dp5"
        android:paddingBottom="@dimen/dp10"
        android:visibility="invisible">
```

```
            <ImageView
                android:layout_width="0dp"
                android:layout_height="@dimen/chat_icon_size"
                android:layout_gravity="center_vertical"
                android:layout_weight="1"
                android:src="@mipmap/chat_camera" />

            <ImageView
                android:layout_width="0dp"
                android:layout_height="@dimen/chat_icon_size"
                android:layout_gravity="center_vertical"
                android:layout_weight="1"
                android:src="@mipmap/chat_pic" />


        </LinearLayout>

    </LinearLayout>
  </FrameLayout>
</LinearLayout>
```

```
package com.palline.merit.mvvm.consultation.chat;

import android.Manifest;
import android.app.Activity;
import android.arch.lifecycle.MutableLiveData;
import android.arch.lifecycle.Observer;
import android.arch.lifecycle.ViewModelProviders;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Color;
import android.graphics.drawable.AnimationDrawable;
import android.media.MediaPlayer;
import android.os.Build;
import android.os.Bundle;
import android.os.CountDownTimer;
import android.os.Handler;
import android.support.annotation.Nullable;
import android.support.v4.app.ActivityCompat;
import android.support.v4.widget.SwipeRefreshLayout;
import android.support.v7.widget.DefaultItemAnimator;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.OrientationHelper;
import android.support.v7.widget.PopupMenu;
import android.support.v7.widget.SimpleItemAnimator;
import android.text.Editable;
import android.text.InputFilter;
import android.text.TextUtils;
import android.text.TextWatcher;
import android.util.Log;
import android.util.SparseArray;
import android.view.Gravity;
import android.view.LayoutInflater;
```

```
import android.view.MenuItem;
import android.view.View;
import android.view.WindowManager;
import android.view.inputmethod.InputMethodManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.PopupWindow;
import android.widget.TextView;

import com.bumptech.glide.Priority;
import com.bumptech.glide.request.RequestOptions;
import com.lcodecore.tkrefreshlayout.RefreshListenerAdapter;
import com.lcodecore.tkrefreshlayout.TwinklingRefreshLayout;
import com.lcodecore.tkrefreshlayout.header.HeadRefreshView;
import com.luck.picture.lib.PictureSelector;
import com.luck.picture.lib.config.PictureConfig;
import com.luck.picture.lib.config.PictureMimeType;
import com.luck.picture.lib.entity.LocalMedia;
import com.palline.merit.R;
import com.palline.merit.frame.base.BaseActivity;
import com.palline.merit.frame.glide.GlideRoundTransform;
import com.palline.merit.frame.recycler.ViewHolder;
import com.palline.merit.frame.retrofit.BaseObserver;
import com.palline.merit.frame.retrofit.RetrofitHelper;
import com.palline.merit.frame.retrofit.RetrofitService;
import com.palline.merit.frame.tool.AppUrl;
import com.palline.merit.frame.tool.Constant;
import com.palline.merit.frame.tool.GsonUtil;
import com.palline.merit.frame.tool.InputFilterUtil;
import com.palline.merit.frame.tool.MyToastUtil;
import com.palline.merit.frame.tool.SPUtils;
import com.palline.merit.frame.tool.ThreadUtils;
import com.palline.merit.frame.tool.TimeUtils;
import com.palline.merit.frame.tool.ToastUtil;
import com.palline.merit.frame.tool.Utils;
import com.palline.merit.frame.voice.MediaPlayManager;
import com.palline.merit.frame.voice.VoiceBean;
import com.palline.merit.frame.voice.VoiceModel;
import com.palline.merit.frame.voice.view.AudioRecorderButton;
import com.palline.merit.frame.websocket.ICallback;
import com.palline.merit.frame.websocket.WsManager;
import com.palline.merit.mvc.adapter.ChatAdapter;
import com.palline.merit.mvc.adapter.MessageDetailAdapter;
import com.palline.merit.mvc.model.ChatHealthFileModel;
import com.palline.merit.mvc.model.ChatItem;
import com.palline.merit.mvc.model.ImagePath;
import com.palline.merit.mvc.presenter.ChatItemImpl;
import com.palline.merit.mvc.presenter.ChatPresenter;
import com.palline.merit.mvc.presenter.ChatPresenterImpl;
import com.palline.merit.mvc.view.ChatView;
import com.palline.merit.mvc.view.PhotoActivity;
import com.palline.merit.mvp.main.problem.healthfilehome.HealthFileHomeActivity;
import com.palline.merit.mvp.main.problem.inhospital.inhospitaldeatil.InHospitalDeatilActivity;
```

```
import
com.palline.merit.mvp.main.problem.medicalexaminationreport.medicalexaminationreportdeatil.MedicalExaminationReportDetailActivity;
import com.palline.merit.mvp.main.problem.outpatientservice.model.OutpatientModel;
import com.palline.merit.mvp.main.problem.outpatientserviceparticulars.OutpatientServiceParticularsActivity;
import com.palline.merit.mvvm.consultation.checksendcontent.CheckSendContentActivity;
import com.palline.merit.mvvm.consultation.entrustdoctorstomanage.view.EntrustDoctorsToManageActivity;
import com.palline.merit.mvvm.consultation.haveanappointmentwithadoctor.view.HaveAnAppointmentWithADoctorActivity;
import com.palline.merit.mvvm.consultation.transfefromonehospitaltoanother.TransferFromOneHospitalToAnotherActivity;
import com.palline.merit.test.ChatModel;
import com.palline.merit.test.ChatViewModel;
import com.palline.merit.test.Data;
import com.palline.merit.test.HistoryActivity;
import com.yanzhenjie.recyclerview.SwipeRecyclerView;

import org.greenrobot.eventbus.EventBus;
import org.greenrobot.eventbus.Subscribe;
import org.greenrobot.eventbus.ThreadMode;

import java.io.File;
import java.util.ArrayList;
import java.util.List;

import io.reactivex.android.schedulers.AndroidSchedulers;
import io.reactivex.disposables.Disposable;
import io.reactivex.schedulers.Schedulers;
import okhttp3.HttpUrl;
import okhttp3.MediaType;
import okhttp3.MultipartBody;
import okhttp3.RequestBody;
import okhttp3.ResponseBody;
import retrofit2.Call;

import static com.palline.merit.mvc.presenter.ChatItemImpl.MessageStatus.SEND_SUCCEED;

public class ChatActivity extends BaseActivity implements TextWatcher, View.OnClickListener, SwipeRefreshLayout.OnRefreshListener,
ChatView, ChatAdapter.OnImageItemClickListener {
    private EditText mEtMsg;
    private Button mBtnSend;
    private ChatPresenterImpl mChatPresenter;
    private ArrayList<ChatItemImpl> mList;
    private TwinklingRefreshLayout mRefresh;
    public SwipeRecyclerView mRecycler;
    private int visitisvalid;
    private int patientId;
    private long begintime;
    private int residueduration;
    private int isStartCountdown;
    private int residueDegree;
    private String visitType;
    private int consultChatId;
    public MessageDetailAdapter messageDetailAdapter;
    private int themeId = R.style.picture_default_style;
    private TextView back_header_txt;
    private boolean isPhoto = true;
    private int doctorId;
    private int chatType;
```

```
    TextView mMessage_list_name_type;
    TextView mMessage_list_restrict;
    CountDownTimer countDownTimer;
    //退出activity时关闭所有定时器，避免造成资源泄漏。
      private SparseArray<CountDownTimer> countDownMap;
    private TextView mRightText;
    private Intent mIntent;
    private Button message_entrust;
    private Handler handler = new Handler() {
    };
    private int isEntrust;
    private LinearLayout mLlVoice;
    private LinearLayout mLlMessage;
    private AudioRecorderButton audio_btn;
    private ImageView mIvVoice;
    private ImageView mIvMessage;
    private View anim;
    private View anim2;
    private RetrofitService mRetrofitService;
    private ImageView mBackHeaderTextRight;
    private MenuItem mMenuItem;
    private ChatViewModel mChatViewModel;
    private PopupWindow window;
    private TextView tv_set_up_the_entrusted_doctor;
    private TextView mTvSetUpTheEntrustedDoctor;
    private TextView mTvManagementOfCommissionedDoctors;
    private TextView mTvViewDelegationMessage;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_chai);

        mChatViewModel = ViewModelProviders.of(this).get(ChatViewModel.class);
        MutableLiveData<Data<ChatModel>> chatModelDataMutableLiveData = mChatViewModel.getChatModelDataMutableLiveData();
        chatModelDataMutableLiveData.observe(this, new Observer<Data<ChatModel>>() {
            @Override
            public void onChanged(@Nullable Data<ChatModel> chatModelData) {
                String errorMsg = chatModelData.getErrorMsg();

                if (errorMsg != null) {
                    MyToastUtil.show(errorMsg);

                } else {
                    ChatModel data = chatModelData.getData();

                }
            }
        });
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            getWindow().getDecorView().setSystemUiVisibility(View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN |
View.SYSTEM_UI_FLAG_LIGHT_STATUS_BAR);
        }
        mRetrofitService = RetrofitHelper.getInstance().getServer();
        bindViews();
        initEvent();
```

```
        countDownMap = new SparseArray<>();
        setRefreshSetting();
        setRefreshListener();
        mRefresh.startRefresh();
        Intent intent = getIntent();
        //病人信息
            String name = intent.getStringExtra(Constant.PATIENT_NAME);
        back_header_txt.setText(name);
        isEntrust = intent.getIntExtra(Constant.IS_ENTRUST, 0);
        patientId = intent.getIntExtra(Constant.PATIENT_ID, 0);
        begintime = intent.getLongExtra(Constant.BEGINTIME, 0);
        residueduration = intent.getIntExtra(Constant.RESIDUEDURATION, 0);
        isStartCountdown = intent.getIntExtra(Constant.PATIENT_START_CHAT, 0);
        residueDegree = intent.getIntExtra(Constant.PATIENT_RESIDUE_DEGREE, 0);
        visitType = intent.getStringExtra(Constant.PATIENT_VISIT_TYPE);
        visitisvalid = intent.getIntExtra(Constant.VISITISVALID, 0);
        String patientImg = intent.getStringExtra(Constant.PATIENT_IMAG);
        chatType = intent.getIntExtra(Constant.PATIENT_TYPE, 0);
        consultChatId = intent.getIntExtra(Constant.PATIENT_CONSULTCHATID, 0);
        //医生信息
            doctorId = SPUtils.getInteger(Constant.ACCOUNTID, 0);
        mChatPresenter = new ChatPresenter(this);
        mChatPresenter.init(consultChatId, chatType);
        WsManager.getInstance().setIsChat(this);
        EventBus.getDefault().register(this);
        setChatTypeText();
        setRestrictText();

        handler.postDelayed(new Runnable() {

            @Override
            public void run() {
                //重新发起一个定时任务
                        ThreadUtils.runOnMainThread(new Runnable() {
                    @Override
                    public void run() {
                        mChatPresenter.init(consultChatId, chatType);
                        //isEntrust字段：1委托
                                if (isEntrust == 1) {
                            mChatPresenter.getChatHistoryTimingRefresh(SPUtils.getString(Constant.TOKEN, Constant.NOSPSTRING), doctorId,
patientId, chatType, 0, consultChatId);
                        } else {
                        }
                    }
                });
                handler.postDelayed(this, 10000);
            }
        }, 10000);
        initData();

        int page = Constant.PAGE;

        mChatViewModel.myhistoryconsultlist(SPUtils.getString(Constant.TOKEN, Constant.NOSPSTRING),
SPUtils.getInteger(Constant.ACCOUNTID, Constant.ERROR_INT), patientId, page, Constant.LIST_NUMBER);
```

```
    }

    private void initEvent() {
        audio_btn.setOnAudioFinishRecorderListener(new AudioRecorderButton.OnAudioFinishRecorderListener() {
            @Override
            public void onFinish(float sec, String filePath) {
                showCustomProgress();
                postVoice(filePath);
            }
        });
    }

    public void postVoice(String path) {
        File file = new File(path);
        RequestBody requestFile =
                RequestBody.create(MediaType.parse("image/png"), file);
        Log.a("TAG", "file.getname..." + file.getName());
        MultipartBody.Part body = MultipartBody.Part.createFormData("file", file.getName(), requestFile);
        Call<ResponseBody> call = mRetrofitService.postVoiceTest(AppUrl.POST_VOICE + SPUtils.getString(Constant.TOKEN,
Constant.NOSPSTRING) + "/uploadaudio", body);
        final HttpUrl url = call.request().url();
        Log.a("json", url.toString());
        mRetrofitService.postVoice(AppUrl.POST_VOICE + SPUtils.getString(Constant.TOKEN, Constant.NOSPSTRING) + "/uploadaudio",
body)
                .subscribeOn(Schedulers.ic())
                .observeOn(AndroidSchedulers.mainThread())
                .subscribe(new BaseObserver<ResponseBody>() {
                    @Override
                    public void onSubscribe(Disposable d) {
                        super.onSubscribe(d);
                    }

                    @Override
                    public void onNext(final ResponseBody responseBody) {
                        super.onNext(responseBody);
                        try {
                            final String path = responseBody.string().toString();
                            Log.a("json", "path..." + path);
                            VoiceBean voiceBean = GsonUtil.parseJsonToBean(path, VoiceBean.class);
                            if (TextUtils.equals(voiceBean.getCode(), "00007")) {
                                sendVoice("{\"audioUrl\":\"" + voiceBean.getAudioUrl() + "\",\"length\":\"" + voiceBean.getLength() + "\"}");
                            } else {
                                ToastUtil.showMyToast(voiceBean.getDesc());
                            }
                        } catch (Exception e) {
                            e.printStackTrace();
                            ToastUtil.showMyToast("数据异常");
                        }
                    }

                    @Override
                    public void onError(Throwable e) {
                        super.onError(e);
                        Log.a("TAG", e.getMessage().toString());
                        ToastUtil.showMyToast(e.getMessage());
                    }
```

```
            @Override
            public void onComplete() {
                super.onComplete();
            }
        });
}

private void initData() {
    if (isEntrust == 1) {
        message_entrust.setVisibility(View.VISIBLE);
    } else {
        if (getIntent().getIntExtra(Constant.PATIENT_TYPE, 0) == 1) {
            List<String> dataList = SPUtils.getDataList(Constant.HOSPITAL_PRIVILEGE_LIST);
            for (int i = 0; i < dataList.size(); i++) {
                if (dataList.get(i).equals("is_revisit_entrustment")) {

                    //显示委托设置医生按钮，目前的方案是任何人都可以设置
                            message_entrust.setVisibility(View.VISIBLE);
                }
            }
        } else {
            List<String> dataList = SPUtils.getDataList(Constant.HOSPITAL_PRIVILEGE_LIST);
            for (int i = 0; i < dataList.size(); i++) {
                if (dataList.get(i).equals("is_online_entrustment")) {

                    //显示委托设置医生按钮，目前的方案是任何人都可以设置
                            message_entrust.setVisibility(View.VISIBLE);
                }
            }
        }

    }
}

private void setRestrictText() {
    mMessage_list_restrict.setVisibility(View.VISIBLE);
    if (visitisvalid == 0) {
        mMessage_list_restrict.setText("结束");
        mMessage_list_restrict.setTextColor(Utils.getResourceColor(R.color.module_item_main_red));
    } else {
        mMessage_list_restrict.setTextColor(Utils.getResourceColor(R.color.android_text));
        if (TextUtils.equals("时间", visitType)) {
            mMessage_list_restrict.setVisibility(View.VISIBLE);
            long residueDuration = residueduration * 1000;
            if (isStartCountdown == 0) {
                if (residueDuration == 0) {
                    mMessage_list_restrict.setTextColor(Utils.getResourceColor(R.color.module_item_main_red));
                    mMessage_list_restrict.setText("结束");
                } else {
                    mMessage_list_restrict.setText("剩余" + TimeUtils.formatDuring(residueDuration));
                }
            } else {
                //记录时间点
                        long timeStamp = TimeUtils.getWebsiteDatetime() - begintime;
                long time = residueDuration - timeStamp;
```

```
        Log.a("TAG", "startitem..time" + time);
        if (getIntent().getBooleanExtra("StartChatActivity", false)) {
            time = begintime;
        }
        if (time > 0) {//判断倒计时是否结束
                    countDownTimer = new CountDownTimer(time, 1000) {
                public void onTick(long millisUntilFinished) {
                    mMessage_list_restrict.setText("剩余" + TimeUtils.formatDuring(millisUntilFinished));
                }

                public void onFinish() {
                    //倒计时结束
                            mMessage_list_restrict.setText("结束");
                    mMessage_list_restrict.setTextColor(Utils.getResourceColor(R.color.module_item_main_red));
                }
            }.start();
            countDownMap.put(mMessage_list_restrict.hashCode(), countDownTimer);
        } else {
            mMessage_list_restrict.setText("结束");
            mMessage_list_restrict.setTextColor(Utils.getResourceColor(R.color.module_item_main_red));
        }
    }
} else if (TextUtils.equals("次数", visitType)) {
    mMessage_list_restrict.setVisibility(View.VISIBLE);
    mMessage_list_restrict.setText("剩余" + residueDegree + "次");
} else if (TextUtils.equals("时间/次数", visitType)) {
    mMessage_list_restrict.setVisibility(View.VISIBLE);
    long residueDuration = residueduration * 1000;
    final long residueDegree = ChatActivity.this.residueDegree;
    if (isStartCountdown == 0) {
        if (residueDuration == 0 || residueDegree == 0) {
            mMessage_list_restrict.setTextColor(Utils.getResourceColor(R.color.module_item_main_red));
            mMessage_list_restrict.setText("结束");
        } else {
            mMessage_list_restrict.setText("剩余" + TimeUtils.formatDuring(residueDuration) + "/" + residueDegree + "次");
        }
    } else {
        //记录时间点
                long timeStamp = TimeUtils.getWebsiteDatetime() - begintime;
        long time = residueDuration - timeStamp;
        Log.a("TAG", "startitem..time" + time);
        if (time > 0) {//判断倒计时是否结束
                    countDownTimer = new CountDownTimer(time, 1000) {
                public void onTick(long millisUntilFinished) {
                    mMessage_list_restrict.setText("剩余" + TimeUtils.formatDuring(millisUntilFinished) + "/" + residueDegree + "次");
                }

                public void onFinish() {
                    //倒计时结束
                            mMessage_list_restrict.setText("结束");
                    mMessage_list_restrict.setTextColor(Utils.getResourceColor(R.color.module_item_main_red));
                }
            }.start();
            countDownMap.put(mMessage_list_restrict.hashCode(), countDownTimer);
        } else {
```

```
                    mMessage_list_restrict.setText("结束");
                    mMessage_list_restrict.setTextColor(Utils.getResourceColor(R.color.module_item_main_red));
                }
                if (residueDegree == 0) {
                    mMessage_list_restrict.setText("结束");
                    mMessage_list_restrict.setTextColor(Utils.getResourceColor(R.color.module_item_main_red));
                    countDownTimer.cancel();
                }
            }
        } else {
            mMessage_list_restrict.setVisibility(View.INVISIBLE);
        }
    }
}

public void setRefreshSetting() {
    HeadRefreshView headRefreshView = new HeadRefreshView(this);
    mRefresh.setHeaderView(headRefreshView);
    mRefresh.setEnableLoadmore(false);
}

public void setRefreshListener() {
    mRefresh.setOnRefreshListener(new RefreshListenerAdapter() {
        @Override
        public void onRefresh(final TwinklingRefreshLayout refreshLayout) {
            super.onRefresh(refreshLayout);
            mChatPresenter.getChatHistory(SPUtils.getString(Constant.TOKEN, Constant.NOSPSTRING), doctorId, patientId, chatType, 0,
consultChatId);
        }

        @Override
        public void onLoadMore(TwinklingRefreshLayout refreshLayout) {
            super.onLoadMore(refreshLayout);
        }
    });
}

private void bindViews() {
    mBackHeaderTextRight = (ImageView) findViewById(R.id.back_header_text_right);
    mBackHeaderTextRight.setImageResource(R.drawable.dt_course_normal);
    mBackHeaderTextRight.setOnClickListener(this);

    audio_btn = (AudioRecorderButton) findViewById(R.id.audio_btn);
    if (ActivityCompat.checkSelfPermission(this, Manifest.permission.RECORD_AUDIO) != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.RECORD_AUDIO}, 0);
    }
    mLlVoice = (LinearLayout) findViewById(R.id.ll_voice);
    mLlMessage = (LinearLayout) findViewById(R.id.ll_message);

    mIvVoice = (ImageView) findViewById(R.id.iv_voice);
    mIvVoice.setOnClickListener(this);
    mIvMessage = (ImageView) findViewById(R.id.iv_message);
    mIvMessage.setOnClickListener(this);
    message_entrust = (Button) findViewById(R.id.message_entrust);
    message_entrust.setOnClickListener(this);
    mMessage_list_name_type = (TextView) findViewById(R.id.message_list_name_type);
```

```
mMessage_list_restrict = (TextView) findViewById(R.id.message_list_restrict);

ImageView iv_pic = (ImageView) findViewById(R.id.iv_pic);
iv_pic.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (isPhoto) {
            isPhoto = false;
            selectPhoto();
        }
    }
});
ImageView iv_camera = (ImageView) findViewById(R.id.iv_camera);
iv_camera.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (isPhoto) {
            isPhoto = false;
            setCamera();
        }
    }
});
ImageView back_header_back = (ImageView) findViewById(R.id.back_header_back);
back_header_back.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        hintKeyBoard();
        finish();
    }
});
back_header_txt = (TextView) findViewById(R.id.back_header_txt);
mEtMsg = (EditText) findViewById(R.id.et_msg);
mEtMsg.addTextChangedListener(new TextWatcher() {

    private CharSequence temp;

    private int editStart;

    private int editEnd;

    @Override

    public void onTextChanged(CharSequence s, int start, int before, int count) {

        temp = s;

    }

    @Override

    public void beforeTextChanged(CharSequence s, int start, int count, int after) {


    }
```

```
        @Override
        public void afterTextChanged(Editable s) {

            editStart = mEtMsg.getSelectionStart();
            editEnd = mEtMsg.getSelectionEnd();

            if (temp.length() >= 1000) {//条件判断可以实现其他功能
                        MyToastUtil.show("你输入的字数已经超过了！");
                s.delete(editStart - 1, editEnd);
                int tempSelection = editStart;
            }
        }
    });
    //然后给editText设置filter，这里给editText设置了两个filter第一个是屏蔽表情，第二个是设置用户输入多少字数的限制
        mEtMsg.setFilters(new InputFilter[]{InputFilterUtil.emojiFilter, new InputFilter.LengthFilter(1000)});
    //防止Activity自动恢复状态时，etMsg有历史文字
        String msg = mEtMsg.getText().toString().trim();
    mBtnSend = (Button) findViewById(R.id.btn_send);
    if (TextUtils.isEmpty(msg)) {
        //设置button为不可用
            mBtnSend.setEnabled(false);
    }
    mBtnSend.setOnClickListener(this);
    mRefresh = (TwinklingRefreshLayout) findViewById(R.id.message_layout_refresh);
    mRecycler = (SwipeRecyclerView) findViewById(R.id.message_layout_recycler);
    //给EditText添加文字改变监听
        mEtMsg.addTextChangedListener(this);
}

private void setChatTypeText() {
    //设置诊断类型 （1复诊、2咨询、3、智能问诊）
        if (chatType == 1) {
        mMessage_list_name_type.setText(R.string.message_chat_type_one);
        mMessage_list_name_type.setTextColor(Utils.getResourceColor(R.color.module_item_main_red));
        mMessage_list_name_type.setBackgroundResource(R.drawable.module_patient_type_round_red);
    } else if (chatType == 2) {
        mMessage_list_name_type.setText(R.string.message_chat_type_two);
        mMessage_list_name_type.setTextColor(Utils.getResourceColor(R.color.module_item_main_blue));
        mMessage_list_name_type.setBackgroundResource(R.drawable.module_patient_type_round_blue);
    } else if (chatType == 3) {
        mMessage_list_name_type.setText(R.string.message_chat_type_three);
        mMessage_list_name_type.setTextColor(Utils.getResourceColor(R.color.module_item_main_green));
        mMessage_list_name_type.setBackgroundResource(R.drawable.module_patient_type_round_green);
    } else if (chatType == 4) {
        mMessage_list_name_type.setText(R.string.message_chat_type_four);
        mMessage_list_name_type.setTextColor(Utils.getResourceColor(R.color.module_item_main_orange));
        mMessage_list_name_type.setBackgroundResource(R.drawable.module_patient_type_round_orange);
```

```
    } else {
        mMessage_list_name_type.setVisibility(View.INVISIBLE);
        mMessage_list_name_type.setTextColor(Utils.getResourceColor(R.color.module_item_main_blue));
        mMessage_list_name_type.setBackgroundResource(R.drawable.module_patient_type_round_blue);
    }
}

private void selectPhoto() {
    PictureSelector.create(this)
            .openGallery(PictureMimeType.ofImage())
            .theme(themeId)
            .maxSelectNum(9)
            .minSelectNum(1)
            .imageSpanCount(3)// 每行显示个数
                    //此参数为0时可多选，为1时为单选，其他亦多选
                    .selectionMode(0)
            .previewImage(true)
            .previewVideo(false)
            .enablePreviewAudio(false) // 是否可播放音频
                    .isCamera(true)
            .enableCrop(false)
            .compress(true)// 是否压缩
                    .minimumCompressSize(100)// 小于100kb的图片不压缩
                    .cropCompressQuality(90)
            .glideOverride(160, 160)
            .previewEggs(true)
            .withAspectRatio(1, 1)
            .hideBottomControls(true)
            .isGif(false)
            .freeStyleCropEnabled(true)
            .circleDimmedLayer(false)
            .showCropFrame(true)
            .showCropGrid(false)
            .openClickSound(true)
            .forResult(PictureConfig.CHOOSE_REQUEST);
}

private void setCamera() {
    PictureSelector.create(this)
            .openCamera(PictureMimeType.ofImage())
            .enableCrop(false)
            .compress(true)// 是否压缩
                    .minimumCompressSize(100)// 小于100kb的图片不压缩
                    .cropCompressQuality(90)
            .freeStyleCropEnabled(true)
            .withAspectRatio(1, 1)
            .forResult(PictureConfig.CHOOSE_REQUEST);
}

public void send(String msg) {
    /**
     * 1. 获取消息（V）
         * 2. 清空EditText（V）
         * 3. 立即将新消息添加到大的集合中（F）
         * 4. 然后更新界面（V）
         * 5. 给消息添加监听器，监听消息发送的状态的变化（F）
```

```
            * 6. 发送消息（F）
            */

        final ChatItem chatItem = new ChatItem(0, 0, "P", "D", chatType, patientId, SPUtils.getInteger(Constant.ACCOUNTID, 0), msg, 1,
SPUtils.getString(Constant.NAME, Constant.NOSPSTRING), SPUtils.getString(Constant.DOCTOR_ICON, Constant.NOSPSTRING),
TimeUtils.getNowString(), consultChatId, 2);
        if (messageDetailAdapter != null && messageDetailAdapter.messageList != null) {
            chatItem.setChatId(-1);
            messageDetailAdapter.setAdd(chatItem, mRecycler);
        }
        WsManager.getInstance().sendReq(chatItem, new ICallback() {
            @Override
            public void onSuccess(Object o) {
                dismissCustomProgress();
                ChatItemImpl detailBean = (ChatItemImpl) o;
                ChatItemImpl.MessageStatus messageStatus = detailBean.getMessageStatus();
                messageStatus = SEND_SUCCEED;
            }

            @Override
            public void onFail(String msg) {
                dismissCustomProgress();
                //消息发送失败
                        ThreadUtils.runOnMainThread(new Runnable() {
                    @Override
                    public void run() {
                        MyToastUtil.show("发送失败");
                    }
                });
                chatItem.setSendFail(true);
                if (messageDetailAdapter != null && messageDetailAdapter.messageList != null) {
                    chatItem.setStatus(0);
                    messageDetailAdapter.setState(chatItem, mRecycler);
                }

            }
        });
    }

    @Override
    public void setAdapter(ArrayList<ChatItemImpl> list, boolean isRefreshToBottom) {
        mList = list;
        if (messageDetailAdapter == null) {
            mRefresh.finishRefreshing();
            messageDetailAdapter = new MessageDetailAdapter(this, false, list, SPUtils.getInteger(Constant.ACCOUNTID, 0));
            LinearLayoutManager mLayoutManager = new LinearLayoutManager(this);
            mLayoutManager.setOrientation(OrientationHelper.VERTICAL);
            mRecycler.setLayoutManager(mLayoutManager);
            SimpleItemAnimator itemAnimator = new DefaultItemAnimator();
            itemAnimator.setSupportsChangeAnimations(false);
            mRecycler.setItemAnimator(itemAnimator);
            mRecycler.setAdapter(messageDetailAdapter);
            mRecycler.scrollToPosition(list.size() - 1);
            setItemClick();
        } else {
            for (int i = 0; i < list.size(); i++) {
```

```
            int chatId = ((ArrayList<ChatItemImpl>) messageDetailAdapter.messageList).get(messageDetailAdapter.messageList.size() -
1).getChatId();
            if (list.get(i).getChatId() == chatId) {
                if (list.size() > messageDetailAdapter.messageList.size()) {
                    if (((ChatItemImpl) messageDetailAdapter.messageList.get(i)).getVoiceState()) {
                        list.get(i).setVoiceState(true);
                    }
                    messageDetailAdapter.messageList = list;
                    mRefresh.finishRefreshing();
                    messageDetailAdapter.notifyDataSetChanged();
                }
            }
        }
    }
    if (!isRefreshToBottom) {
        messageDetailAdapter.messageList = list;
        mRefresh.finishRefreshing();
        messageDetailAdapter.notifyDataSetChanged();
    }
}

private void setItemClick() {
    messageDetailAdapter.setOnItemClickListener(new MessageDetailAdapter.OnItemClickListener() {
        @Override
        public void onTextClick(int position) {

        }

        @Override
        public void onImageClick(int position) {

            Intent intent = new Intent(ChatActivity.this, PhotoActivity.class);
            if (position < mList.size()) {
                intent.putExtra("chatImage", mList.get(position).getContent());
                startActivity(intent);
            } else {
                MyToastUtil.show("当前网络不好，请重新刷新此界面");
            }
        }

        @Override
        public void onHeadClick(int position) {

        }

        @Override
        public void onHealthFileClick(int position) {
            ChatHealthFileModel chatHealthFileModel = GsonUtil.parseJsonToBean(mList.get(position).getContent(),
ChatHealthFileModel.class);
            onInitData(String.valueOf(chatHealthFileModel.getHospitalDataName()), String.valueOf(chatHealthFileModel.getLsh()),
chatHealthFileModel.getType());
        }

        @Override
        public void onVoiceClick(ViewHolder holder, String content, final int position) {
            //播放动画
```

```
            if (anim != null) {
        anim.setBackgroundResource(R.drawable.adj);
        anim = null;
    }
    // 播放动画
            if (anim2 != null) {
        anim2.setBackgroundResource(R.drawable.voice_left);
        anim2 = null;
    }
    anim = holder.itemView.findViewById(R.id.id_recorder_anim);
    anim.setBackgroundResource(R.drawable.voice_anim);
    final AnimationDrawable animationDrawable = (AnimationDrawable) anim.getBackground();
    animationDrawable.start();
    // 播放音频
            VoiceModel voiceModel = GsonUtil.parseJsonToBean(content, VoiceModel.class);
    if (voiceModel != null) {
        MediaPlayManager.play(voiceModel.getAudioUrl(), new MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mediaPlayer) {
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        for (int i = 0; i < messageDetailAdapter.messageList.size(); i++) {
                            ((ChatItemImpl) messageDetailAdapter.messageList.get(i)).setVoiceState(false);
                        }
                        Log.a("debug", "取消" + position);
                        messageDetailAdapter.notifyDataSetChanged();
                    }
                });

                // 播放动画
                            if (anim != null) {
                    anim.setBackgroundResource(R.drawable.adj);
                    anim = null;
                }
            }
        });
    }
}

@Override
public void onReceiverVoiceClick(ViewHolder holder, String content, final int position) {
    // 播放动画
            if (anim != null) {
        anim.setBackgroundResource(R.drawable.adj);
        anim = null;
    }
    // 播放动画
            if (anim2 != null) {
        anim2.setBackgroundResource(R.drawable.voice_left);
        anim2 = null;
    }

    anim2 = holder.itemView.findViewById(R.id.id_recorder_anim);
    anim2.setBackgroundResource(R.drawable.voice_receiver_anim);
    final AnimationDrawable animationDrawable = (AnimationDrawable) anim2.getBackground();
```

```
            animationDrawable.start();

            // 播放音频
                    VoiceModel voiceModel = GsonUtil.parseJsonToBean(content, VoiceModel.class);
            MediaPlayManager.play(voiceModel.getAudioUrl(), new MediaPlayer.OnCompletionListener() {
                @Override
                public void onCompletion(MediaPlayer mediaPlayer) {
                    runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            for (int i = 0; i < messageDetailAdapter.messageList.size(); i++) {
                                ((ChatItemImpl) messageDetailAdapter.messageList.get(i)).setVoiceState(false);
                            }
                            Log.d("debug", "取消" + position);
                            messageDetailAdapter.notifyDataSetChanged();
                        }
                    });
                    // 播放动画
                                if (anim2 != null) {
                        anim2.setBackgroundResource(R.drawable.voice_left);
                        anim2 = null;
                    }
                }
            });
        }
    });
}

public void onInitData(String hospitalDataName, String lsh, int type) {
    switch (type) {
        case 1:
            mIntent = new Intent(ChatActivity.this, OutpatientServiceParticularsActivity.class);
            break;
        case 2:
            mIntent = new Intent(ChatActivity.this, InHospitalDeatilActivity.class);
            break;
        case 3:
            mIntent = new Intent(ChatActivity.this, MedicalExaminationReportDetailActivity.class);
            break;
    }
    final OutpatientModel.RstBean rstBean = new OutpatientModel.RstBean();
    rstBean.setHospitalDataName(hospitalDataName);
    rstBean.setLsh(lsh);
    rstBean.setType(type);
    mIntent.putExtra(Constant.OUTPATIENT_MODEL, rstBean);
    startActivity(mIntent);
}

@Override
public void setReceiver(ChatItemImpl item, int senderId) {
    if (senderId == patientId) {
    }
}

@Override
public void planImagePath(String path) {
```

```
        final ChatItem chatItem = new ChatItem(0, 0, "P", "D", chatType, patientId, SPUtils.getInteger(Constant.ACCOUNTID, 0), path, 2,
SPUtils.getString(Constant.NAME, Constant.NOSPSTRING), SPUtils.getString(Constant.DOCTOR_ICON, Constant.NOSPSTRING),
TimeUtils.getNowString(), consultChatId, 2);
        if (messageDetailAdapter != null && messageDetailAdapter.messageList != null) {
            chatItem.setChatId(-1);
            messageDetailAdapter.setAdd(chatItem, mRecycler);
        }
        WsManager.getInstance().sendReq(chatItem, new ICallback() {
            @Override
            public void onSuccess(Object o) {
                ChatItemImpl detailBean = (ChatItemImpl) o;
                ChatItemImpl.MessageStatus messageStatus = detailBean.getMessageStatus();
                messageStatus = SEND_SUCCEED;
            }

            @Override
            public void onFail(String msg) {
                dismissCustomProgress();
                ThreadUtils.runOnMainThread(new Runnable() {
                    @Override
                    public void run() {
                        MyToastUtil.show("图片发送失败");
                    }
                });
                //图片消息发送失败
                        chatItem.setSendFail(true);
                if (messageDetailAdapter != null && messageDetailAdapter.messageList != null) {
                    chatItem.setStatus(0);
                    messageDetailAdapter.setState(chatItem, mRecycler);
                }
            }
        });
    }

    @Override
    public void onInit(int size) {
        if (mMenuItem != null) {
            mMenuItem.setTitle("健康档案（" + size + "）");
        }
    }

    @Override
    public void onRefresh() {

    }

    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {

    }
```

```
@Override
public void afterTextChanged(Editable s) {
    //每输入一个字符，就调用一次，如果输入字符后，内容为空
        if (TextUtils.isEmpty(s.toString())) {
        mBtnSend.setEnabled(false);
    } else {
        mBtnSend.setEnabled(true);
    }
}

@Override
public void onClick(View v) {
    Intent intent;
    PopupMenu popupMenu = new PopupMenu(mContext, v);

    switch (v.getId()) {
        case R.id.tv_set_up_the_entrusted_doctor:
            intent = new Intent(ChatActivity.this, ExpandableListActivity.class);
            intent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
            intent.putExtra(Constant.CHAT_TYPE, chatType);
            startActivity(intent);
            break;
        case R.id.tv_management_of_commissioned_doctors:
            intent = new Intent(ChatActivity.this, EntrustDoctorsToManageActivity.class);
            intent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
            intent.putExtra(Constant.CHAT_TYPE, chatType);
            startActivity(intent);
            break;
        case R.id.tv_view_delegation_message:
            Intent messageEntrustIntent = new Intent(ChatActivity.this, CheckSendContentActivity.class);
            messageEntrustIntent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
            messageEntrustIntent.putExtra(Constant.CHAT_TYPE, chatType);
            messageEntrustIntent.putExtra("patientId", patientId);
            startActivity(messageEntrustIntent);
            break;
        case R.id.back_header_text_right:
            popupMenu.setGravity(Gravity.RIGHT);
            popupMenu.getMenuInflater().inflate(R.menu.chat_menu, popupMenu.getMenu());
            //弹出式菜单的菜单项点击事件
                    popupMenu.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener() {

                @Override
                public boolean onMenuItemClick(MenuItem item) {
                    Intent intent;
                    switch (item.getItemId()) {
                        case R.id.mune_transfer_from_one_hospital_to_another:
                            intent = new Intent(ChatActivity.this, TransferFromOneHospitalToAnotherActivity.class);
                            intent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
                            intent.putExtra(Constant.CHAT_TYPE, chatType);
                            startActivity(intent);
                            break;
                        case R.id.mune_health:
                            intent = new Intent(ChatActivity.this, HealthFileHomeActivity.class);
                            intent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
                            intent.putExtra(Constant.CHAT_TYPE, chatType);
                            startActivity(intent);
```

```
                    break;
                case R.id.mune_history:
                    intent = new Intent(mContext, HistoryActivity.class);
                    intent.putExtra(Constant.PATIENT_ID, patientId);
                    startActivity(intent);
                    break;
                case R.id.mune_register:
                    intent = new Intent(mContext, HaveAnAppointmentWithADoctorActivity.class);
                    intent.putExtra(Constant.PATIENT_ID, patientId);
                    intent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
                    startActivity(intent);
                    break;
            }
            return false;
        }
    });
    popupMenu.show();
    mMenuItem = popupMenu.getMenu().getItem(0);
    popupMenu.setGravity(Gravity.CENTER);
    mChatPresenter.init(consultChatId, chatType);
    break;
case R.id.message_entrust:
    showPopUpWindows();


//            popupMenu.setGravity(Gravity.RIGHT);
//            popupMenu.getMenuInflater().inflate(R.menu.entrusted_management, popupMenu.getMenu());
//            //弹出式菜单的菜单项点击事件
//            popupMenu.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener() {
//                @Override
//                public boolean onMenuItemClick(MenuItem item) {
//                    Intent intent;
//                    switch (item.getItemId()) {
//                        case R.id.set:
////                            Intent setEntrustDoctorIntent = new Intent(ChatActivity.this, SetEntrustDoctorActivity.class);
////                            setEntrustDoctorIntent.putExtra("consultChatId", consultChatId);
////                            setEntrustDoctorIntent.putExtra(Constant.CHAT_TYPE, chatType);
////                            setEntrustDoctorIntent.putExtra("isEntrust", isEntrust);
////                            startActivityForResult(setEntrustDoctorIntent, 11);
//                            intent = new Intent(ChatActivity.this, ExpandableListActivity.class);
//                            intent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
//                            intent.putExtra(Constant.CHAT_TYPE, chatType);
//                            startActivity(intent);
//                            break;
//                        case R.id.administration:
//                            intent = new Intent(ChatActivity.this, EntrustDoctorsToManageActivity.class);
//                            intent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
//                            intent.putExtra(Constant.CHAT_TYPE, chatType);
//                            startActivity(intent);
//                            break;
//                        case R.id.look:
//                            Intent messageEntrustIntent = new Intent(ChatActivity.this, CheckSendContentActivity.class);
//                            messageEntrustIntent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
//                            messageEntrustIntent.putExtra(Constant.CHAT_TYPE, chatType);
//                            messageEntrustIntent.putExtra("patientId", patientId);
//                            startActivity(messageEntrustIntent);
```

```
//                        break;
//                    }
//                    return false;
//                }
//            });
//            popupMenu.show();
//            popupMenu.setGravity(Gravity.CENTER);
            break;
        case R.id.btn_send:
            String msg = mEtMsg.getText().toString();
            mEtMsg.getText().clear();
            send(msg);
            break;
        case R.id.right_txt:
            intent = new Intent(this, HealthFileHomeActivity.class);
            intent.putExtra(Constant.CONSULT_CHAT_ID, consultChatId);
            intent.putExtra(Constant.CHAT_TYPE, chatType);
            startActivity(intent);
            break;
        case R.id.iv_voice:
            if (ActivityCompat.checkSelfPermission(this, Manifest.permission.RECORD_AUDIO) !=
PackageManager.PERMISSION_GRANTED) {
                ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.RECORD_AUDIO}, 0);
            } else {
                if (mLlVoice.getVisibility() == View.VISIBLE) {
                    mLlVoice.setVisibility(View.INVISIBLE);
                    mLlMessage.setVisibility(View.VISIBLE);
                } else {
                    mLlMessage.setVisibility(View.INVISIBLE);
                    mLlVoice.setVisibility(View.VISIBLE);
                }
            }
            break;
        case R.id.iv_message:
            if (mLlVoice.getVisibility() == View.VISIBLE) {
                mLlVoice.setVisibility(View.INVISIBLE);
                mLlMessage.setVisibility(View.VISIBLE);
            } else {
                mLlVoice.setVisibility(View.VISIBLE);
                mLlMessage.setVisibility(View.INVISIBLE);
            }
            break;
    }
}

@Override
public void onImageItemClick(int position, List<ImagePath> imagePathList) {

}

@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    //isPhoto用来设置是否可拍照和进入相册，避免两个冲突
        isPhoto = true;
    if (resultCode == Activity.RESULT_OK) {
```

```
        switch (requestCode) {
            case PictureConfig.CHOOSE_REQUEST:
                List<LocalMedia> selectList = PictureSelector.obtainMultipleResult(data);
                for (int i = 0; i < selectList.size(); i++) {
                    LocalMedia media = selectList.get(i);
                    String path = media.getCompressPath();
                    mChatPresenter.postImage(path);
                }
                //5. 显示到界面
                        RequestOptions options = new RequestOptions()
                    .transform(new GlideRoundTransform())
                    .error(R.mipmap.head_doctor)
                    .dontAnimate()
                    .priority(Priority.HIGH);
                break;
            case 11:
                isEntrust = 1;
                break;
        }
    }
}

@Subscribe(threadMode = ThreadMode.MAIN)
public void onEvent(ChatItemImpl chatItem) {
    dismissCustomProgress();
    //如果发送人是当前的聊天病人则刷新当前界面
        if (chatItem.getSenderId() == patientId || chatItem.getSenderId() == SPUtils.getInteger(Constant.ACCOUNTID, 0)) {
        //如果reduceTime是1，咨询次数减一次
            if (chatItem.getReduceTime() == 1) {
            residueDegree--;
            if (residueDegree == 0) {
                mMessage_list_restrict.setText("结束");
                mMessage_list_restrict.setTextColor(Color.RED);
            } else {
                mMessage_list_restrict.setText("剩余" + residueDegree + "次");
            }
        }
        if (messageDetailAdapter != null && messageDetailAdapter.messageList != null && chatItem.getReceiverType().equals("P")) {
            chatItem.setStatus(1);
            messageDetailAdapter.setState(chatItem, mRecycler);
        } else if (messageDetailAdapter != null && messageDetailAdapter.messageList != null) {
            messageDetailAdapter.setAdd(chatItem, mRecycler);
        }
    }
}

@Subscribe(threadMode = ThreadMode.MAIN)
public void onEvent(VoiceBean voiceBean) {
    MediaPlayManager.release();
    for (int i = 0; i < messageDetailAdapter.messageList.size(); i++) {
        ((ChatItemImpl) messageDetailAdapter.messageList.get(i)).setVoiceState(false);
    }
    messageDetailAdapter.notifyDataSetChanged();
}

public void sendVoice(String msg) {
```

```
        /**
         * 1. 获取消息（V）
            * 2. 清空EditText（V）
            * 3. 立即将新消息添加到大的集合中（F）
            * 4. 然后更新界面（V）
            * 5. 给消息添加监听器，监听消息发送的状态的变化（F）
            * 6. 发送消息（F）
            */

        final ChatItem chatItem = new ChatItem(0, 0, "P", "D", chatType, patientId, SPUtils.getInteger(Constant.ACCOUNTID, 0), msg, 4,
SPUtils.getString(Constant.NAME, Constant.NOSPSTRING), SPUtils.getString(Constant.DOCTOR_ICON, Constant.NOSPSTRING),
TimeUtils.getNowString(), consultChatId, 2);
        if (messageDetailAdapter != null && messageDetailAdapter.messageList != null) {
            chatItem.setChatId(-1);
            messageDetailAdapter.setAdd(chatItem, mRecycler);
        }
        WsManager.getInstance().sendReq(chatItem, new ICallback() {
            @Override
            public void onSuccess(Object o) {
                dismissCustomProgress();
                ChatItemImpl detailBean = (ChatItemImpl) o;
                ChatItemImpl.MessageStatus messageStatus = detailBean.getMessageStatus();
                messageStatus = SEND_SUCCEED;
            }

            @Override
            public void onFail(String msg) {
                dismissCustomProgress();
                //消息发送失败
                        ThreadUtils.runOnMainThread(new Runnable() {
                    @Override
                    public void run() {
                        MyToastUtil.show("发送失败");
                    }
                });
                chatItem.setSendFail(true);
                if (messageDetailAdapter != null && messageDetailAdapter.messageList != null) {
                    chatItem.setStatus(0);
                    messageDetailAdapter.setState(chatItem, mRecycler);
                }

            }
        });
    }

    public void showPopUpWindows() {
        double width = getScreenWidth();
        View view = LayoutInflater.from(this).inflate(R.layout.item_popup_window_entrusted_management, null);
        if (window == null) {
            window = new PopupWindow(view, (int) width, WindowManager.LayoutParams.WRAP_CONTENT);
            window.setFocusable(true);//获取焦点
                    window.setOutsideTouchable(true);//获取外部触摸事件
                    window.setTouchable(true);//能够响应触摸事件
                    window.setAnimationStyle(R.style.popwindow_anim_style);
            window.setBackgroundDrawable(getResources().getDrawable(R.color.transparent));
            window.setOnDismissListener(new PopupWindow.OnDismissListener() {
```

```
            @Override
            public void onDismiss() {
                setBackgroundAlpha(1f);
            }
        });
        viewBindViews(view);
    }
    window.showAtLocation(message_entrust, Gravity.BOTTOM, 0, 0);
    setBackgroundAlpha(0.7f);
}

private void viewBindViews(View view) {
    mTvSetUpTheEntrustedDoctor = (TextView) view.findViewById(R.id.tv_set_up_the_entrusted_doctor);
    mTvManagementOfCommissionedDoctors = (TextView) view.findViewById(R.id.tv_management_of_commissioned_doctors);
    mTvViewDelegationMessage = (TextView) view.findViewById(R.id.tv_view_delegation_message);

    mTvSetUpTheEntrustedDoctor.setOnClickListener(this);
    mTvManagementOfCommissionedDoctors.setOnClickListener(this);
    mTvViewDelegationMessage.setOnClickListener(this);
}

@Override
public void onDestroy() {
    super.onDestroy();
    //医生读了病人的消息
        mChatPresenter.doctorReadPatient(SPUtils.getString(Constant.TOKEN, Constant.NOSPSTRING),
SPUtils.getInteger(Constant.ACCOUNTID, 0), patientId, getIntent().getIntExtra(Constant.PATIENT_TYPE, 0), consultChatId);
    EventBus.getDefault().unregister(this);
    cancelAllTimers();
}

/**
 * 隐藏键盘
 */
public void hintKeyBoard() {
    //拿到InputMethodManager
    InputMethodManager imm = (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
    //如果window上view获取焦点 && view不为空
        if (imm.isActive() && getCurrentFocus() != null) {
        //拿到view的token 不为空
            if (getCurrentFocus().getWindowToken() != null) {
            //表示软键盘窗口总是隐藏，除非开始时以SHOW_FORCED显示。
                imm.hideSoftInputFromWindow(getCurrentFocus().getWindowToken(), InputMethodManager.HIDE_NOT_ALWAYS);
        }
    }
}

/**
 * 清空资源
 */
public void cancelAllTimers() {
    if (countDownMap == null) {
        return;
    }
    for (int i = 0, length = countDownMap.size(); i < length; i++) {
```

```
            CountDownTimer cdt = countDownMap.get(countDownMap.keyAt(i));
            if (cdt != null) {
                cdt.cancel();
            }
        }
    }

    @Override
    public boolean isImmersionBarEnabled() {
        return false;
    }

    @Override
    public void onPause() {
        super.onPause();
        if (audio_btn != null) {
            try {
                audio_btn.send();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }

    @Override
    public boolean isSetBarAlpha() {
        return false;
    }
}
```