写接口的方法情况有以下两种：
1、调用某方法实现接口方法后的事件处理（接口方法一般无参）
2、实现接口后和回掉数据处理（例如一个方法传入一个URL,通过网络获取数据返回一个对象，那么这个接口就是一个有参的实现方法）

具体的步骤：
1、创建实现某业务方法
例1：调用该方法实现数据库的查询，比如：业务方法只是想在回调方法中做提示或其它操作
public void get(接口类 接口类名){}

例2：调用该方法需要给一个或多个参数，最终返回一个数据
public void get(String S, 接口类 接口类名){
return 返回数据；
}

2、创建内部接口及接口方法
public interface OnResultListener {
void onResult(); 业务完成后的事件处理方法

void onResult(String s); 有返回数据的接口方法
}
3、分析业务的类型
如果业务是例1则考虑用事件处理方法，

4，具体实现的方法
（1）定义自定义接口

```
public OnResultListener onResultListener;
    public void setOnSelectedListener(OnResultListener onResultListener) {
            this.onResultListener = onResultListener;
    }
    public interface OnResultListener {
            void onSucceed(String s);                    //成功的方法
            void onDefeated(String s);              //失败的方法
    }
```

（2）在调用方法中声明接口参数

```
public static void getqingqou(Context context,String url,final OnResultListener onResultListener)
{
        在方法需要的地方调用接口中的方法
    if (onResultListener != null) {
            onResultListener.onSucceed(string);
    }
}
```

（3）使用的方法：调用方法 new一个接口

```
Wangluoqingqou.getqingqou(getApplicationContext(), mUrl, new OnResultListener() {

                    @Override
                    public void onSucceed(String s) {
```

```
                mTEXT.setText(s+"请求成功了");
            }

            @Override
            public  void  onDefeated(String  s)  {
            }
        });
```

自定义接口回调方法用法二：　在自定义类的方法中使用如在onTouchEvent方法中使用回调方法

```
    /**
     * 点击图标的回调方法定义
     */
    public  OnTasktListener onTasktListener;
    public void setOnSelectedListener(OnTasktListener onTasktListener) {
        this.onTasktListener = onTasktListener;
    }
    public interface OnTasktListener {
        void onSucceed();
    }

//在onTouchEvent里回调自定义接口的方法
@Override
public boolean onTouchEvent(MotionEvent event) {
if (event.getAction() == MotionEvent.ACTION_UP) {
if (getCompoundDrawables()[2] != null) {
boolean touchable = event.getX() > (getWidth()getTotalPaddingRight())&&
(event.getX() < ((getWidth() - getPaddingRight())));
        if (touchable) {
                    //里面写上自己想做的事情在这里写一个回调方法
                    if(onTasktListener!=null){
                        onTasktListener.onSucceed();
                    }
        }
return super.onTouchEvent(event);
}
```

使用：

```
ClearEditText username = (ClearEditText) findViewById(R.id.username);
username.setOnSelectedListener(new ClearEditText.OnTasktListener() {
@Override
public void onSucceed() {
        showToast("我去不会成功了吧");
        }
```

```
});
```

```
public class ClearEditText extends android.support.v7.widget.AppCompatEditText {
    /**
     * 右侧的图标控件
     */
    private Drawable mClearDrawable;
    /**
     * 点击图标的回调方法定义
     */
    public OnTasktListener onTasktListener;
    public void setOnSelectedListener(OnTasktListener onTasktListener) {
        this.onTasktListener = onTasktListener;
    }
    public interface OnTasktListener {
        void onSucceed();
    }
```

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    if (event.getAction() == MotionEvent.ACTION_UP) {
        if (getCompoundDrawables()[2] != null) {

            boolean touchable = event.getX() > (getWidth() - getTotalPaddingRight()
                    && (event.getX() < ((getWidth() - getPaddingRight())));

            if (touchable) {
                //里面写上自己想做的事情在这里写一个回调方法
                if(onTasktListener!=null){
                    onTasktListener.onSucceed();
                }
            }
        }
    }

    return super.onTouchEvent(event);
}
```

```
    ClearEditText username = (ClearEditText) findViewById(R.id.username);

    username.setOnSelectedListener(new ClearEditText.OnTasktListener() {
        @Override
        public void onSucceed() {
            showToast("我去不会成功了吧");
        }
    });
```

```
//*********************************************
控件点击一下改变控件的背景颜色并改变控件大小，再点击一下再次改变控件的背景颜色，
再点击一下改变控件的背景颜色并改变控件大小
//*********************************************
        private int   status = 0;

        final Button button = (Button) findViewById(R.id.butt);
        button.setOnClickListener(new OnClickListener() {

        @Override
        public void onClick(View v) {
                switch (status) {
                        case 0:    //默认是0点击后改为1改变颜色值
                                button.setBackgroundColor(Color.RED);
                                status = 1;
                                setLayout(button,55,55);
                                break;
                        case 1:    //为1时点击后改为2改颜色值
                                button.setBackgroundColor(Color.BLACK);
                                Toast.makeText(getApplicationContext(), "现在是2了", 1).show();
                                status = 2;
                                break;
                        case 2:    //为2时点击后恢复默认值
                                button.setBackgroundColor(Color.GREEN);
                                setLayout(button,45,45);
                                Toast.makeText(getApplicationContext(), "现在是0了", 1).show();
                                status = 0;
                                break;
                                }
                        }
                });
        }

        //动态改变控件的宽高值
        public void setLayout(Button button,int i,int j){
                RelativeLayout.LayoutParams rl = (LayoutParams) button.getLayoutParams();
                rl.width =dip2px(getApplicationContext(), i);    //接收px值    需要将dp值 转为px值
                rl.height=dip2px(getApplicationContext(), j);      //接收px值
                button.setLayoutParams(rl);
        }


         /**
```

```
  * 根据手机的分辨率从 dp 的单位 转成为 px(像素)     一般用于代码中设置的值
  */
public   int dip2px(Context context, float dpValue) {
      final float scale = context.getResources().getDisplayMetrics().density;
      return (int) (dpValue * scale + 0.5f);
}

/**
  * 根据手机的分辨率从 px(像素) 的单位 转成为 dp
  */
public   int px2dip(Context context, float pxValue) {
      final float scale = context.getResources().getDisplayMetrics().density;
      return (int) (pxValue / scale + 0.5f);
}
```