WebView的内存泄露： 参考文档

- 首先应该做的就是不能在xml中定义webview节点，而是在需要的时候动态生成。即：可以在使用WebView的地方放置一个LinearLayout类似ViewGroup的节点，然后在要使用WebView的时候，动态生成即：

```
WebView      mWebView = new WebView(getApplicationgContext());
LinearLayout mll      = findViewById(R.id.xxx);
mll.addView(mWebView);
```

- 然后一定要在onDestroy()方法中显式的调用

```
protected void onDestroy() {
    super.onDestroy();
    mWebContainer.removeAllViews();
    mWebView.clearHistory();
    mWebView.clearCache(true);
    mWebView.loadUrl("about:blank");
    mWebView.freeMemory(); //new code
    mWebView.pauseTimers(); //new code
    mWebView.destroy()
    mWebView = null;
}
```

内存泄露的一些建议

- In summary, to avoid context-related memory leaks, remember the following:

  Do not keep long-lived references to a context-activity (a reference to an activity should have the same life cycle as the activity itself)，

  Try using the context-application instead of a context-activity

  Avoid non-static inner classes in an activity if you don’t control their life cycle, use a static inner class and make a weak reference to the activity inside，

  And remember that a garbage collector is not an insurance against memory leaks. Last but not least, we try to make such leaks harder to make happen whenever we can.

WebView设置标题的方式：

- onReceivedTitle进入页面是有用，但是对于某些android版本， WebView.goBack()无效，不如onPageFinished来得安全

```

 WebView mWebView = (WebView) findViewById(R.id.mwebview);
    mWebView.setWebViewClient(new WebViewClient() {
        @Override
        public void onPageFinished(WebView view, String url) {
            ExperimentingActivity.this.setTitle(view.getTitle());
        }
    });



webview.setWebChromeClient(new WebChromeClient() {
    @Override
    public void onReceivedTitle(WebView view, String title) {
        super.onReceivedTitle(view, title);
        if (!TextUtils.isEmpty(title)) {
            YourActivity.this.setTitle(title);
        }
```

```
    }
});
```

This doesn't seem to be called after WebView.goBack() is called. So, if you need to handle back navigation I think onPageFinished() might be a better choice.

webview 要打开JS才能使得网页内部效果有效，其次如果要做一些手势的监听，可以在自定义CustomWebView里面做，

WebView内部View点击事件的监听

```
private class MyWebViewClient extends WebViewClient {

  @Override

    public boolean shouldOverrideUrlLoading(WebView view, String url) {

            if (!TextUtils.isEmpty(url) && url.contains(GO_TO_PERSON_INFO)) {

 // 这里做可以监听，因为每个点击的View是可以带有Url的，如果Url的值符合监听要求，就做特殊处理

}}
```

Webview关于Js代码的回调 参考文档：http://blog.csdn.net/wangtingshuai/article/details/8631835

```
mWebView.loadUrl("javascript:YH.appCallback('"
        + upLoadActionMap.get("actionId") +"','" + callBackjson + "')");
```

Webview 如何监听Js对话框的弹出与处理

public boolean onJsAlert (WebView view, String url, String message, JsResult result) Added in API level 1

Tell the client to display a javascript alert dialog. If the client returns true, WebView will assume that the client will handle the dialog. If the client returns false, it will continue execution.这种情况下，恐怕只能用Activity Context

```
@Override
public boolean onJsAlert(WebView view, String url, String message, final JsResult result) {
    AlertDialog.Builder b = new AlertDialog.Builder(getActivity());
    b.setTitle("警告");
    b.setMessage(message);
    b.setPositiveButton("确认", new AlertDialog.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            result.confirm();
        }
    });
    b.setCancelable(true);
    b.create();
    b.show();
    return true;
};
```

public boolean onJsConfirm (WebView view, String url, String message, JsResult result) Added in API level 1

Tell the client to display a confirm dialog to the user. If the client returns true, WebView will assume that the client will handle the confirm dialog and call the appropriate JsResult method. If the client returns false, a default value of false will be returned to javascript. The default behavior is to return false.

```
@Override
public boolean onJsConfirm(WebView view, String url, String message, final JsResult result) {
    AlertDialog.Builder b = new AlertDialog.Builder(getActivity());
    b.setTitle("确认");
    b.setMessage(message);
    b.setPositiveButton("确认",
            new AlertDialog.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    result.confirm();
                }
            });
    b.setNegativeButton("取消",
            new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    result.cancel();
                }
            });
    b.setCancelable(true);
    b.create();
    b.show();

    return true;
};
```