

转载请注明链接：https://blog.csdn.net/feather_wch/article/details/79900081

Android换肤： AppCompatActivity从XML创建View的原理分析

版本：2018/4/11-1

本质原理：

1. Activity是通过 Factory 进行View的创建
2. 自定义 Factory 就能拦截创建过程，创建自己的 View

1、AppCompatActivity的OnCreate流程

```

//AppCompatActivity.java
protected void onCreate(@Nullable Bundle savedInstanceState) {
    final AppCompatActivity delegate = getDelegate();
    //1. 初始化LayoutInflater, 并且设置过Factory(没有设置过就新建)
    delegate.installViewFactory();
    //2. 执行正常的onCreate流程
    delegate.onCreate(savedInstanceState);
    //xxx
    super.onCreate(savedInstanceState);
}
//AppCompatActivityImplV9.java
public void installViewFactory() {
    LayoutInflater inflater = LayoutInflater.from(mContext);
    //1. 没有Factory, 系统会创建一个Factory去进行XML到View的转换
    if (inflater.getFactory() == null) {
        LayoutInflaterCompat.setFactory2(inflater, this);
    } else {
        if (!(inflater.getFactory2() instanceof AppCompatActivityImplV9)) {
            Log.i(TAG, "The Activity's LayoutInflater already has a Factory installed" + " so v
        }
    }
}
//LayoutInflaterCompat.java
public static void setFactory2(@NonNull LayoutInflater inflater, @NonNull LayoutInflater.Factory
    //1. 能将Factory接口绑定到创建View的LayoutInflater(IMPL类型为LayoutInflaterCompatBaseImpl)
    IMPL.setFactory2(inflater, factory);
}
//LayoutInflaterCompat.java内部类LayoutInflaterCompatBaseImpl:
static class LayoutInflaterCompatBaseImpl {
    //xxx
    public void setFactory2(LayoutInflater inflater, LayoutInflater.Factory2 factory) {
        inflater.setFactory2(factory);
    }
}
//LayoutInflater.java-完成Factory的创建
public void setFactory2(Factory2 factory) {
    //xxx
    if (mFactory == null) {
        mFactory = mFactory2 = factory;
    } else {
        mFactory = mFactory2 = new FactoryMerger(factory, factory, mFactory, mFactory2);
    }
}

```

2、AppCompatActivity的OnCreate中setContentView()的流程

```

//AppCompatActivityImplV9.java
    public void setContentView(int resId) {
        //xxx
        //1. 获取到父容器Content
        ViewGroup contentParent = (ViewGroup) mSubDecor.findViewById(android.R.id.content);
        contentParent.removeAllViews();
        //2. 通过LayoutInflater加载布局文件
        LayoutInflater.from(mContext).inflate(resId, contentParent);
        mOriginalWindowCallback.onContentChanged();
    }

//LayoutInflater.java
    public View inflate(@LayoutRes int resource, @Nullable ViewGroup root) {
        return inflate(resource, root, root != null);
    }

//LayoutInflater.java
    public View inflate(@LayoutRes int resource, @Nullable ViewGroup root, boolean attachToRoot) {
        final Resources res = getContext().getResources();
        //xxx
        final XmlResourceParser parser = res.getLayout(resource);
        //1. 重点
        return inflate(parser, root, attachToRoot);
    }

//LayoutInflater.java
    public View inflate(XmlPullParser parser, @Nullable ViewGroup root, boolean attachToRoot) {
        ...
        final String name = parser.getName();//控件名
        //1. 将XmlPullParser转换为View的属性AttributeSet,给其他方法使用
        final AttributeSet attrs = Xml.asAttributeSet(parser);
        //2. Temp是XML文件中的根布局(name为"LinearLayout"等等)
        final View temp = createViewFromTag(root, name, inflaterContext, attrs);
        //3. 将XML根布局中temp下面所有的子View都进行加载
        rInflateChildren(parser, temp, attrs, true);
        //4. 将根布局tmp中找到的所有View贴到root中(content view)
        if (root != null && attachToRoot) {
            root.addView(temp, params);
        }
        ...
    }

/**=====
 * 通过提供的属性AttributeSet attrs, 创建View
 * // LayoutInflater.java
 *=====*/
View createViewFromTag(View parent, String name, Context context, AttributeSet attrs, boolean attachToRoot) {
    //1. 彩蛋?<blink>标签会进行闪烁
    if (name.equals(TAG_1995)) {
        // Let's party like it's 1995!
        return new BlinkLayout(context, attrs);
    }
    //2. 通过Factory创建View
    View view;
    view = mFactory2.onCreateView(parent, name, context, attrs);

```

```

        //xxx
        return view;
    }

//AppCompatActivityImplV9.java
    public final View onCreateView(View parent, String name, Context context, AttributeSet attr
        ...
        //创建View
        return onCreateView(parent, name, context, attrs);
    }

//AppCompatActivityImplV9.java
    public View onCreateView(View parent, final String name, @NonNull Context context, @NonNull /
        ...
        return mAppCompatActivityInflater.onCreateView(parent, name, context, attrs, inheritContext,
            IS_PRE_LOLLIPOP, /* Only read android:theme pre-L (L+ handles this anyway) */
            true, /* Read read app:theme as a fallback at all times for legacy reasons */
            VectorEnabledTintResources.shouldBeUsed() /* Only tint wrap the context if enab
        );
    }

//AppCompatActivityInflater.java-最终完成从XML到View的转变
    public final View onCreateView(View parent, final String name, Context context, AttributeSet

        View view = null;
        switch (name) {
            case "TextView":
                view = new AppCompatActivityTextView(context, attrs);
                break;
            case "ImageView":
                view = new AppCompatActivityImageView(context, attrs);
                break;
            case "Button":
                view = new AppCompatActivityButton(context, attrs);
                break;
            case "EditText":
                view = new AppCompatActivityEditText(context, attrs);
                break;
            case "Spinner":
                view = new AppCompatActivitySpinner(context, attrs);
                break;
            case "ImageButton":
                view = new AppCompatActivityImageButton(context, attrs);
                break;
            case "CheckBox":
                view = new AppCompatActivityCheckBox(context, attrs);
                break;
            case "RadioButton":
                view = new AppCompatActivityRadioButton(context, attrs);
                break;
            case "CheckedTextView":
                view = new AppCompatActivityCheckedTextView(context, attrs);
                break;
            case "AutoCompleteTextView":
                view = new AppCompatActivityAutoCompleteTextView(context, attrs);
                break;

```

```
        case "MultiAutoCompleteTextView":
            view = new AppCompatMultiAutoCompleteTextView(context, attrs);
            break;
        case "RatingBar":
            view = new AppCompatRatingBar(context, attrs);
            break;
        case "SeekBar":
            view = new AppCompatSeekBar(context, attrs);
            break;
    }
    ...
    return view;
}
```

3、自定义Activity中通过Factory对控件的创建进行拦截，实现“换肤”效果：

```

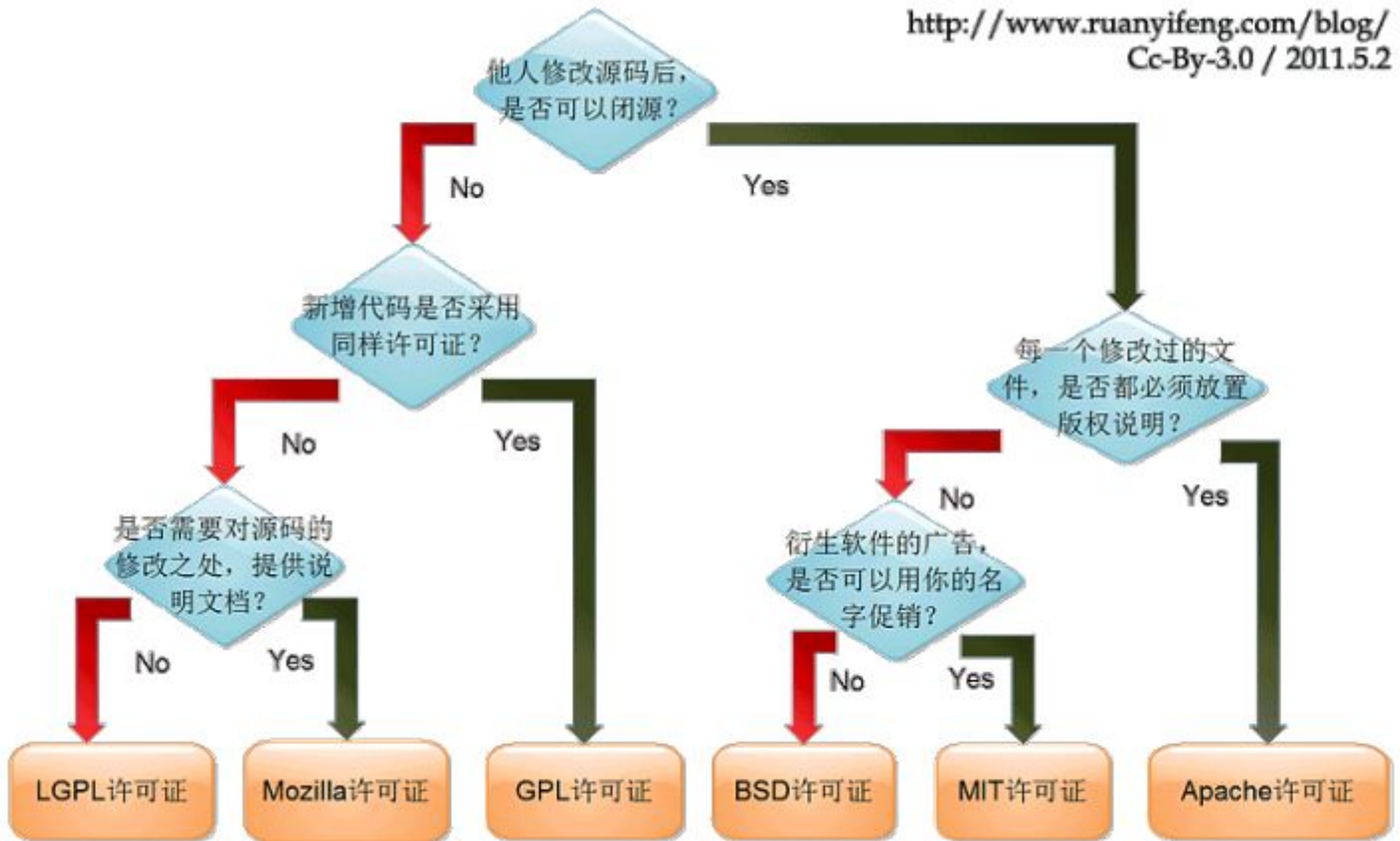
public class SkinActivity extends AppCompatActivity{
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        LayoutInflaterCompat.setFactory2(LayoutInflater.from(this), new LayoutInflater.Factory2() {
            @Override
            public View onCreateView(View parent, String name, Context context, AttributeSet attrs) {
                AppCompatActivity delegate = getDelegate();
                View view = delegate.onCreateView(parent, name, context, attrs);
                return view;
            }
        });

        @Override
        public View onCreateView(String name, Context context, AttributeSet attrs) {
            View view = null;
            switch (name) {
                case "TextView":
                    view = new AppCompatActivityTextView(context, attrs);
                    break;
                case "ImageView":
                    view = new AppCompatActivityImageView(context, attrs);
                    break;
                case "Button":
                    view = new AppCompatActivityButton(context, attrs);
                    break;
                case "EditText":
                    view = new AppCompatActivityEditText(context, attrs);
                    break;
                //...
            }
            return view;
        }
    }
}

```

学习和参考资料



1. [Android 探究 LayoutInflater setFactory](#)
2. [Github:Android-Skin\(换肤库\)](#)