1. <Text>组件，它专门用来显示文本。
2. Image 显示图像, 属性source设置图片地址对象
   1. let pic = {  
       uri: 'http://upload.wikimedia.org/wikipedia/commons/d/de/Bananavarieties.jpg'  
      }
   2. <Image>0.50及以上不再支持包裹内容；
3. View 常用作其他组件的容器，来帮助控制布局和样式。
4. 样式:可以传入一个数组——在数组中位置居后的样式对象比居前的优先级更高，这样你可以间接实现样式的继承。
5. 使用StyleSheet.create来集中定义组件的样式
6. TextInput,在应用中输入文本的组件,
   1. 常用的配置: 自动完成、自动大小写、占位文字，以及多种不同的键盘类型（如纯数字键盘）
   2. onChangeText事件读取用户的输入.目前唯一从TextInput中取值的方法.即使用onChangeText写入state，然后从this.state中取出值。
   3. 其他的事件:onSubmitEditing和onFocus
7. 有些属性仅在multiline为true或者false的时候有效.
   1. 例如:当multiline=false时,为元素的某一边添加边框样式(borderBottomColor, borderLeftWidth等)将不会生效.为了能够实现效果可以使用一个View来包裹TextInput.
   2. TextInput在安卓上默认有一个底边框，同时会有一些padding。如果要想使其看起来和iOS上尽量一致，则需要设置padding: 0，同时设置underlineColorAndroid="transparent"来去掉底边框。
   3. 又，在安卓上如果设置multiline = {true}，文本默认会垂直居中，可设置textAlignVertical: 'top'样式来使其居顶显示。
   4. 又又，在安卓上长按选择文本会导致windowSoftInputMode设置变为adjustResize，这样可能导致绝对定位的元素被键盘给顶起来。要解决这一问题你需要在AndroidManifest.xml中明确指定合适的windowSoftInputMode( https://developer.android.com/guide/topics/manifest/activity-element.html )值，或是自己监听事件来处理布局变化。
8. StatusBar
   1. StatusBar.setHidden(true) 隐藏状态栏
9. FlatList
   1. simpleDemo

<FlatList

data={[{key: 'a'}, {key: 'b'}]}

renderItem={({item}) => <Text>{item.key}</Text>}

/>

* 1. 设置extraData={this.state} 正确触发FlatList的更新

//this.refs.\_flatlist.scrollToEnd(); //滚动到末尾，如果不设置getItemLayout属性的话，可能会比较卡

this.refs.\_flatlist.scrollToIndex({viewPosition: 0, index: 18}); //viewPosition 0位于顶部，1位于底部，0.5位于中间。

//this.refs.\_flatlist.scrollToOffset({animated: true, offset: 200}); //滚动到指定的偏移的位置