hansontable的渲染定义方式有多种，常见的有 NumericRenderer、TextRenderer、CheckboxRenderer，但他们都是通过registerRenderer来渲染 的，registerRenderer是hansontable的渲染公用方法，其他的渲染都是在此基础上扩展的。

那么我们需要自定义一个文本渲染该如何实现呢？请跟我来：

## 定义自己的渲染方法

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/mafan121/article/details/48491207)

1. var MyRenderer = function (instance, td, row, col, prop, value, cellProperties) {
2. Handsontable.renderers.TextRenderer.apply(this, arguments);
3. $(td).css("text-align", cel[k].align);
4. $(td).css("vertical-align", cel[k].valign);
5. }）;

这里的instance是hansontable的核心方法接口对象，td是一个渲染的单元格，value是单元格的值，cellProperties是单元格的渲染方法对象，其中包含一个renderer属性，定义了该用那种渲染方式。

Handsontable.renderers.TextRenderer.apply(this, arguments);

这句话不可少。其含义是将自定义方法中的配置信息通过hansontable的Text渲染应用到当前window对象上。this表示当前window对象，arguments表示渲染方法中的7个参数，是一个参数数组。

## 在单元格属性配置中调用自定义的渲染方法

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/mafan121/article/details/48491207)

1. cells: function (row, col, prop) {
2. var cellProperties = {};
3. if (row === 0 || this.instance.getData()[row][col] === 'readOnly') {
4. cellProperties.readOnly = true;
5. }
6. if (row === 0) {
7. cellProperties.renderer = firstRowRenderer;//调用首行渲染方法
8. }
9. else {
10. cellProperties.renderer = MyRenderer;//调用自定义渲染方法
11. }
12. return cellProperties;
13. }

或者直接在cells中调用renderer属性

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/mafan121/article/details/48491207)

1. cells: function(row, col, prop) {
2. this.renderer = myRenderer;
3. }

这样就能使用自定义的渲染方法了。

## 为hansontable中的元素添加事件。

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/mafan121/article/details/48491207)

1. Handsontable.Dom.removeEvent(document, eventName, function(){});
2. Handsontable.Dom.addEvent(document, eventName, function(){});

document是dom元素，eventName是事件名，function(){}是事件处理函数。