Render Tree and Renderer

1. **What is Render Tree?**

   This tree is of visual elements in the **order** in which they will be displayed. It is the visual representation of the document. The purpose of this tree is to enable painting the contents in their correct order.

2. **definition of the Renderer**

   — Firefox calls the elements in the render tree "frames". WebKit uses the term renderer or render object.

   — A renderer knows how to lay out and paint itself and its children.

3. **How does renderer work?**

   — Each renderer represents a rectangular area usually corresponding to a node's CSS box. It includes geometric(几何的) information like width, height and position.

   — The box type is affected by the "display" value of the style attribute that is relevant to the node.

   — The element type is also considered: for example, form controls and tables have special frames.

DEFINITION OF WEBKIT'S RENDEROBJECT:

WebKit's RenderObject class

```
class RenderObject{
  virtual void layout();
  virtual void paint(PaintInfo);
  virtual void rect repaintRect();
  Node* node;  //the DOM node
  RenderStyle* style;  // the computed style
  RenderLayer* containgLayer; //the containing z-index layer
}
```

CREATE NODE DEPEND ON DISPLAY ATTRIBUTE:

what type of renderer should be create

```
RenderObject* RenderObject::createObject(Node* node, RenderStyle* style)
{
    Document* doc = node->document();
    RenderArena* arena = doc->renderArena();
    ...
    RenderObject* o = 0;
    switch (style->display()) {
        case NONE:
            break;
        case INLINE:
            o = new (arena) RenderInline(node);
            break;
        case BLOCK:
            o = new (arena) RenderBlock(node);
            break;
        case INLINE_BLOCK:
            o = new (arena) RenderBlock(node);
            break;
        case LIST_ITEM:
            o = new (arena) RenderListItem(node);
            break;
        ...
    }
    return o;
}
```

The render tree relation to the DOM tree

1. **Non-visual DOM elements will not be inserted in the render tree.**
   — The renderers correspond to DOM elements, but the relation is not one to one.

   — An example is the "head" element. Also elements whose display value was assigned to "none" will not appear in the tree (whereas elements with "hidden" visibility will appear in the tree).

2. **Some DOM elements correspond to several visual objects.**
   — These are usually elements with complex structure that cannot be described by a single rectangle.

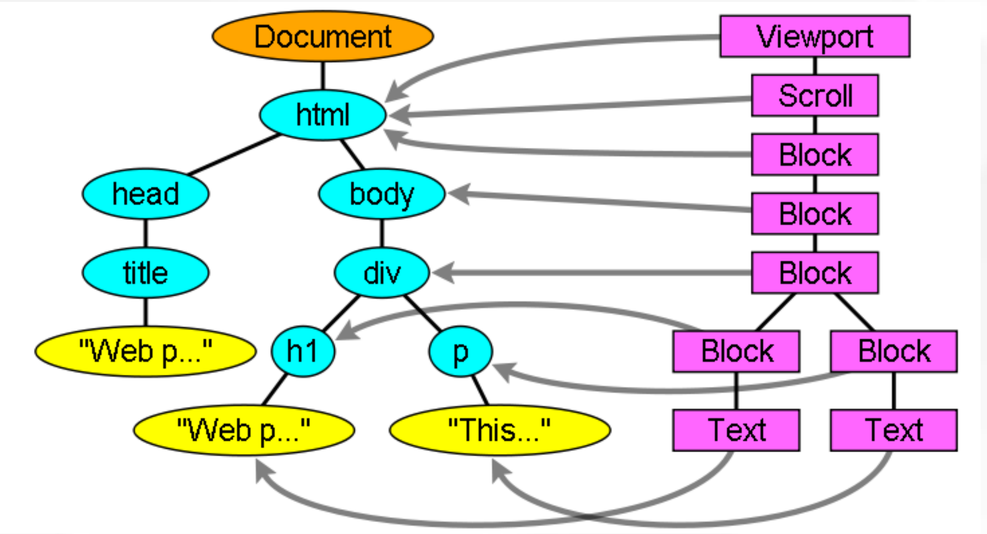
   — For example, the "select" element has three renderers: one for the display area, one for the drop down list box and one for the button.

   — Another example of multiple renderers is broken HTML. According to the CSS spec an inline element must contain either only block elements or only inline elements. In the case of mixed content, anonymous block renderers will be created to wrap the inline elements.

3. **Some render objects correspond to a DOM node but not in the same place in the tree.**
   Floats and absolutely positioned elements are out of flow, placed in a different part of the tree, and mapped to the real frame. A placeholder frame is where they should have been.

The "Viewport" is the initial containing block.

In WebKit it will be the "RenderView" object



THE RENDER TREE AND THE CORRESPONDING DOM TREE