岗位职责及介绍

大家好，我是工业物联网信息能力中心的毛浪，我的岗位是前端UI工程师。

> 前端UI这个岗位细分的话，它其实包含两个方面：一是UI设计方面，二是前端开发方面。
>
> - 先总的来说，前端UI最终实现web或移动互联网网页视觉的呈现与用户交互，视觉呈现就是我们点开一个网站或APP所看到的所有东西，包括颜色，布局，特效动画、各种组件等等。用户交互就是，比如看到一个按钮组件、表单组件你可以点击，系统能给予你相应回馈，能增删改查一系列信息或数据的。
> - 职责细分来讲
>   - UI设计（或称界面设计）是指对软件的人机交互、操作逻辑、界面美观的整体设计。UI设计分为实体UI和虚拟UI，互联网常用的UI设计是虚拟UI，UI即User Interface(用户界面)的简称。我们日常生活中所用到的：手机、电脑、电视、车载系统、iPad、ATM机、工业中控系统.....只要是带有电子屏幕的显示设备，都有需要UI设计。[1]
>   - 前端开发是创建WEB页面或APP等前端界面呈现给用户的过程，通过HTML，CSS及JavaScript以及衍生出来的各种技术、框架、解决方案，来实现互联网产品的用户界面视觉静态呈现和动态交互功能。简单来说对于用户界面的最终实现UI设计定义了蓝图，前端开发负责编码实现。[2]

大概介绍了一下前端UI这个岗位是做什么的，接下来我想与大家分享一下，近期结项项目中的几个案例实现。
首先，项目介绍

项目介绍

> 项目名称：web自由组态
> 项目概述：web自由组态的实现能使用户通过类似“搭积木”的简单方式来完成自己所需要的软件功能，而不需要编写计算机程序，它能将自动化过程和设备中采集的各种信息以图形化等更易于理解的方式进行显示，也能对信息执行必要的分析，发出指令等。主要为可视化数据监控屏的应用。

下面这几张图就是典型可视化数据监控屏应用场景：也能完美适应pc端和移动端屏幕
图片1
图片2

> web自由组态本质上就是实现各种数据可视化应用场景搭建的工具软件，例如图片上的数据、图表、表单等用户均可通过自定义编辑实现。拿图表组件来举例，类似于制图软件，用户通过从组件库中拖拽出对应图表比如柱状图，通过设置柱状图的宽高、位置、最大最小刻度、主次刻度等分、刻度线及柱体颜色等等样式定制最终达成目标效果。

项目案例

接下来我将介绍项目开发中遇到的几个案例

> 案例1：数据组件的定制化使用实现
>
> - 案例分析：
>   - 案例需求：实现例如折线图、柱状图、仪表盘、环形图、饼图、天气API等数据组件的动态化定制。
>   - 竞品实现方案：对比繁易组态、BY组态， 对于数据组件的实现单纯采用静态图片代替，简单来说就是拖拽出来的数据组件只是一张固定死的图片。这就会产生一下几个问题：
>   - 1、在组件大小变换方面：图片缩放会导致失真模糊，竞品1采取禁止数据组件进行大小变换，竞品2不做任何处理，且天气API数据组件的最终渲染大小与用户定义不符。
>   - 2、在实时渲染方面：静态图片并不能满足更随用户设置而实时变化渲染效果。
>   - 项目实施方案：引入第三方库Echarts（响应式数据可视化框架）进行功能模块开发。
>   - 实施方案难点：底层原理上不能将Echarts渲染出的数据图形纳入Konvajs生成的图形容器中，致使数据图形无法渲染和控制，组件拖拽、变换、导航操作等问题成为实现难点。
>   - 难点问题解决：数据组件渲染实现使用CSS定位计算及层叠顺序，数据组件控制实现使用Konva间接图形对象控制。
> - 案例实现：
>   - 渲染实现：
>   - 相对定位：对元素设置相对定位后可以通过设置垂直或水平位置，让这个元素“相对于”它的起点进行移动，值得注意的是在使用相对定位时，无论是否进行移动，元素仍然占据原来的空间。所以相对定位一般结合绝对定位来使用，对设置了绝对定位的元素的父元素使用相对定位。
>   - 绝对定位：设置为绝对定位的元素框从文档流完全删除，元素不保留原来的空间，并相对于其包含块进行定位。由于绝对定位的元素与文档流无关，所以它们可以覆盖页面上的其它元素。于是将元素引入三维坐标，通过定义元素在z轴上位置来控制元素的层叠堆放顺序。
>   - 层叠：设置元素的层叠索引，基于同一个层叠上下文比较，可以看出Element#4,Element#5,Element#6都是Element#3的子元素，这三个子元素同属于一个层叠上下文，它们进行比较。然后它们和Element#3作为一个整体与Element#3的兄弟元素进行比较。所以可以看出尽管Elment#4的z-index(层叠索引)为6，但是他还是在Element#1的下面。
>   - 控制实现：
>   - 在组件基础数据结构中加入Konva矩形图形对象，只设置矩形宽高数据以实现隐藏效果。动态添加echarts图表容器标签，并初始化位置及大小，通过监听隐形矩形对象的位置变化、图形变换来间接改变echarts图表容器位置及大小，最终实现跟随用户设置实时渲染数据组件。

案例2：组件拖拽及变换限制实现

- 案例分析：
  - 案例需求：实现适用于各类型组件的拖拽（位移）和变换（旋转变换、缩放变换）限制功能。
  - 案例描述：可以看到PPT上的两幅动图，简单来说所谓拖拽及变换限制就是组件位置不可超出可视区域，组件变换例如旋转，缩放也不可超出可视区域。
  - 竞品实现方案：竞品1对于超出可视区域的组件未做处理，竞品2对组件超出可视区域后能正常显示。对比分析，竞品1的方案组件不能正常显示，当组件完全超出可视区域将造成组件丢失。竞品2的方案设置组件编辑时能超出可视区域，于场景正式发布后只显示可视区域内的内容，是一种良好的解决方案。
  - 项目实施方案：项目一期采用限制组件不可超出可视区范围的方案，二期则考虑朝竞品2的方向优化。
    - 实施方案难点：方案难点在于处理多个组件同时拖拽变换边缘点数据计算问题。
    - 难点问题解决：计算组件位移及变换后的正盒子数据。
- 案例实现:
  1. 勾股定理：计算斜边长：a^2 + b^2 = c^2；
  2. Math.atan2函数：atan2() 返回从原点(0,0)到(x,y)点的线段与x轴正方向之间的平面角度(弧度值)
  3. 三角函数及其诱导公式：con a = l临边/斜边；sin a = 对边/斜边；cos(π/2 + a) = -sin a；sin(π/2 + a) = cos a;
     已知旋转盒的(x,y)坐标和旋转角度rad；计算旋转后的盒子四个顶点的坐标值，取四个点坐标值中最小的x，y作为正盒子的定位坐标点，最大x与最小x之间的差值作为正盒子的宽，高同理。调用Konva转换器类Transformer的boundBoxFunc函数计算正盒子的x,y坐标值是否为负，横纵坐标分别加宽高是否大于画布宽高，即超出范围，返回旧盒子坐标及大小。

```
function getCorner(pivotX, pivotY, diffX, diffY, angle) {
    const distance = Math.sqrt(diffX * diffX + diffY * diffY);
    angle += Math.atan2(diffY, diffX);

    const x = pivotX + distance * Math.cos(angle);
    const y = pivotY + distance * Math.sin(angle);
    return { x: x, y: y };
  }
  function getClientRect(rotatedBox) {
    const { x, y, width, height } = rotatedBox;
    const rad = rotatedBox.rotation;

    const p1 = getCorner(x, y, 0, 0, rad);
    const p2 = getCorner(x, y, width, 0, rad);
    const p3 = getCorner(x, y, width, height, rad);
    const p4 = getCorner(x, y, 0, height, rad);

    const minX = Math.min(p1.x, p2.x, p3.x, p4.x);
    const minY = Math.min(p1.y, p2.y, p3.y, p4.y);
    const maxX = Math.max(p1.x, p2.x, p3.x, p4.x);
    const maxY = Math.max(p1.y, p2.y, p3.y, p4.y);

    return {
      x: minX,
      y: minY,
      width: maxX - minX,
      height: maxY - minY,
    };
  }
  shape.setAbsolutePosition(newAbsPos);
```

个人理解与思考

> 1. 工作方面：
>   对于工作，我认为一个软件开发的流程中，在确保项目可行性的前提下，项目需求的分析是重中之重。因为它具有决策性、方向性、策略性的作用。在软件开发过程中它的作用要远远大于程序设计。许多大型应用系统的失败，最后归结到需求分析的失败：要么获取需求的方法不当，使得需求分析不到位或不彻底，导致开发者反复多次的进行需求分析，致使设计、编码、测试无法顺利进行；要么与客户配合不好，导致客户对需求不确认或客户需求不断变化，同样致使设计、编码、测试无法顺利进行。而对于处于开发阶段的程序设计者来说，能做到的就是对现有需求的统一性、正确性、透彻性理解，朝正确的方向开发程序。

> 2. 个人成长方面：

咳咳，众所周知，程序员的日常工作是很枯燥的啊，除了写代码就还是写代码，仅个人观点来说就是少有条件能让我们有机会感悟人生和成长啊，除了被代码搞崩溃时可能会自我安慰式的去看一两篇毒鸡汤

以毒攻毒啊。咳那回归正题：

> 首先，我觉得人要会给自己找麻烦，我们可能对现有的工作得心应手，每天也循环重复着一些自己熟悉得不能再熟悉得工作，也可能现有的工作中没有什么问题能难得倒你，即使会有一点小摩擦，但基本上过的比较舒适，那么这个时候我们就该警醒了，这代表我们正在陷入自己的舒适圈中。人还是要学会给自己找麻烦，而有麻烦就意味着有问题，而正是问题激发我们去学习，去实践，去观察。创造始于问题，有了问题才会有思考，有思考才会有新的认知，才不会止步不前。不要视麻烦如洪水猛兽，要欣然接受麻烦，在处理麻烦的过程中获取新知，积累经验，开拓阅历。麻烦的到来代表我们拥有了一个自我提升的机会，而对于机会，人就应该踊跃抓取。如果光是比如只掌握着现有的知识，循环着现有的思维模式和观点，安于呆在自己的舒适圈中，就难免逐渐麻木，最后转眼时光流逝岁月匆匆，在想改变，已然来不及了。
> 从自身出发，目前的我们迫切的需要自身的成长，特别是能力和经验的增长，毕竟现代社会内卷的这么厉害，自身没有拥有与时间流逝而相应增长的能力的话，也怕前浪真的被后浪拍死在沙滩上啊，是吧。当然无论是主动找的还是被动接受的麻烦，他来了，可能该烦还是烦，但像我上述所说那样转变思维，我们能欣然接受，能积极主动的去成长，何乐而不为呢。

> 其次就是责任，责任表现在工作和生活中方方面面，对于工作的责任，我体会最深的就是凡是承诺的工作和任务，一定要保质保量、如期完成。不给自己找借口，找理由，哪怕需要加班也要保证准时完工，至少保证不能因个人的失误而造成团队的损失。这就意味着需要我们提高自己的专注力、执行力，做事注重效率，切忌拖延，不懂就问不可盲目自信。除开个人的责任，还有团队所共同承担的责任也与每个人息息相关，拒绝个人英雄主义，或以自我为中心，重视团队协作，互帮互助，要在维护团队利益的前提下，发挥个人的最大价值，承担起个人与团队的责任，与团队共进退才能赢得最终的胜利。

总结

正确全面的理解自己的岗位职责，有助于减少工作的冲突与摩擦，有利于我们工作的顺利进展。
web自由组态项目顺利结项，开发过程中学到不断学习到新的知识和框架，积累了有关H5canvas类型项目的实战经验。让我感觉收货颇丰。借此机会也向领导、恩师和部门同事表达感谢，感谢理解和包容，我们是一个和谐快乐的团队。
致此，感谢聆听，欢迎各位评委的提问。谢谢。

1. 1、项目前期参与产品构思，提出产品界面规划，把握产品最终界面实现效果。
   2、项目过程中收集和分析用户对于GUI的需求，根据产品原型图进行界面视觉设计和交互逻辑设计，对页面进行优化，使用户操作更趋于人性化。
   3、后期结合用户体验，优化完善设计，制定可行的产品品质提升方案。
   4、日常维护现有的应用产品，积累通用类软件或互联网应用产品的人机交互方面的理解和认识，具有敏锐的用户体验观察力，富有创新精神。

5、好的UI设计不仅是让软件变得有个性有品位，还要让软件的操作变得舒适简单、自由，充分体现软件的定位和特点。 ↩

2. 1、项目前期参与项目会议、定义整体产品体验和技术方案。
   2、项目过程中负责前端编码技术规范制定，文档撰写及实现界面交互(即代码输出)，负责前端页面技术难点攻克及不同平台的兼容性调试，包括主流PC浏览器及手机浏览器的兼容。
   3、后期负责持续优化前端体验和页面响应速度，负责整体web前端用户体验及改进工作。可以说前端是最贴近用户的程序员，前端的能力就是能让产品从90分进化到100分，甚至更好。
   4、日常维护和升级现有网站项目，快速定位并修复现有网站缺陷。也负责对Web与移动互联网前沿技术的持续研究与新技术创新。 ↩