# 前端工作流程及错误调试技巧

[1. 完整项目人员分配 1](#_Toc27974)

[2. 前端工作对象 2](#_Toc16934)

[4. 项目流程图 3](#_Toc32129)

[5. 常见错误整理 3](#_Toc15470)

[6.网站渲染流程 4](#_Toc4057)

### 完整项目人员分配

1.1产品需求设计/产品--负责

1. 根据市场调研的需求，写需求文档，
2. 利用交互设计工具，完成 产品原型设计<设计师能读懂的框图>

1.2UI--

1）根据需求及UI完成psd ,jpg <最终网页让客户到的效果图>

\*UED-[用户体验设计师](https://baike.baidu.com/item/%E7%94%A8%E6%88%B7%E4%BD%93%E9%AA%8C%E8%AE%BE%E8%AE%A1%E5%B8%88" \t "https://baike.baidu.com/item/UED/_blank)(User Experience Designer)

1.3前端工程师--

1. 根据ui设计的psd图，完成兼容性良好，性能优良的html网页；
2. 根据原型、流程图、需求文档等结合后端提供接口、数据完成网站交互功能

3）优化网站性能，重视用户体验

1.4后台开发人员--

1)根据数据库，及三方接口封装成符合页面数据需要和页面流转的接口

1. 将接口体现在swagger，及其他接口文档上；方便前后端联调前期的分离开发
2. 配合前端开发人员，修改接口，完善功能
3. 测试阶段，在终端、移动端测试审核项目完整流程
4. 要让技术和业务结合起来；

1.5 测试人员--

1. 根据需求文档和测试环境的项目进行各种测试
2. 编写测试案例，测试分类：冒烟测试，功能测试，ui测试 ？？？

1.6 项目管理人员--

1）和各方人员进行资源调配，及负责功能的验收

2）把控项目整个的流程

3）对于项目有技术的远见和整合能力

### 前端工作对象

需求/产品

流程图/原型

需求文档

UI

PSD图

Jpg/png图

静态html

后端

约定的链接（参数）

接口文档

/Swagger

需求/产品

流程图/原型

需求文档

交互效果的完整项目

**3.**参数传递过程

（url参数 session缓存 接口入参 接口出参 ）

session缓存

url参数

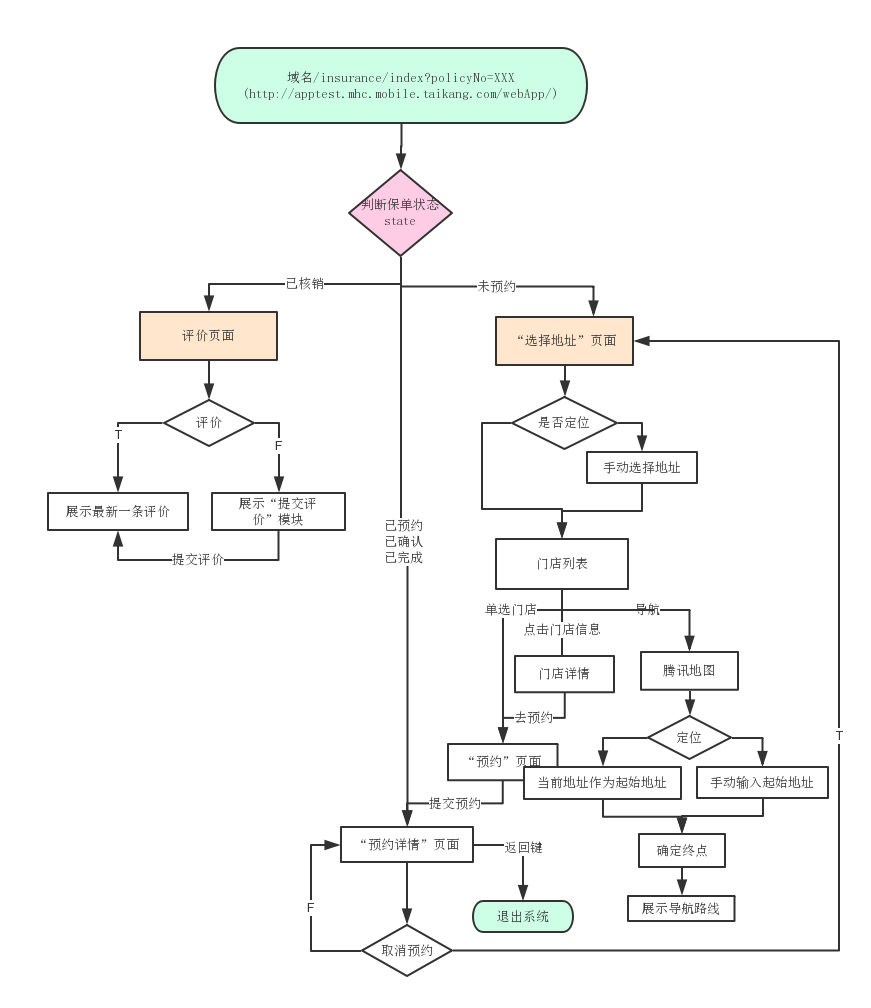
接口入参

接口出参

页面展示

### 项目流程图

属于自己的项目流程跳转图

Eg:

### 常见错误整理

200-服务器成功返回网页--

200 （成功） 服务器已成功处理了请求。通常，这表示服务器提供了请求的网页。 201 （已创建） 请求成功并且服务器创建了新的资源。 202 （已接受） 服务器已接受请求，但尚未处理。 203 （非授权信息） 服务器已成功处理了请求，但返回的信息可能来自另一来源。 204 （无内容） 服务器成功处理了请求，但没有返回任何内容。 205 （重置内容） 服务器成功处理了请求，但没有返回任何内容。 206 （部分内容） 服务器成功处理了部分 GET 请求

3xx （重定向）---

301 （永久移动） 请求的网页已永久移动到新位置。服务器返回此响应（对 GET 或 HEAD 请求的响应）时，会自动将请求者转到新位置。 302 （临时移动） 服务器目前从不同位置的网页响应请求，但请求者应继续使用原有位置来进行以后的请求。 303 （查看其他位置） 请求者应当对不同的位置使用单独的 GET 请求来检索响应时，服务器返回此代码。 304 （未修改） 自从上次请求后，请求的网页未修改过。服务器返回此响应时，不会返回网页内容。 305 （使用代理） 请求者只能使用代理访问请求的网页。如果服务器返回此响应，还表示请求者应使用代理。 307 （临时重定向） 服务器目前从不同位置的网页响应请求，但请求者应继续使用原有位置来进行以后的请求

400 （错误请求） 服务器不理解请求的语法。--

401 （未授权） 请求要求身份验证。 对于需要登录的网页，服务器可能返回此响应。 403 （禁止） 服务器拒绝请求。 404 （未找到） 服务器找不到请求的网页。 405 （方法禁用） 禁用请求中指定的方法。 406 （不接受） 无法使用请求的内容特性响应请求的网页。 407 （需要代理授权） 此状态代码与 401（未授权）类似，但指定请求者应当授权使用代理。 408 （请求超时） 服务器等候请求时发生超时。 409 （冲突） 服务器在完成请求时发生冲突。服务器必须在响应中包含有关冲突的信息。 410 （已删除） 如果请求的资源已永久删除，服务器就会返回此响应。 411 （需要有效长度） 服务器不接受不含有效内容长度标头字段的请求。 412 （未满足前提条件） 服务器未满足请求者在请求中设置的其中一个前提条件。 413 （请求实体过大） 服务器无法处理请求，因为请求实体过大，超出服务器的处理能力。 414 （请求的 URI 过长） 请求的 URI（通常为网址）过长，服务器无法处理。 415 （不支持的媒体类型） 请求的格式不受请求页面的支持。 416 （请求范围不符合要求） 如果页面无法提供请求的范围，则服务器会返回此状态代码。 417 （未满足期望值） 服务器未满足”期望”请求标头字段的要求。

5xx（服务器错误） 这些状态代码表示服务器在尝试处理请求时发生内部错误。 这些错误可能是服务器本身的错误，而不是请求出错。

500 （服务器内部错误） 服务器遇到错误，无法完成请求。 501 （尚未实施） 服务器不具备完成请求的功能。例如，服务器无法识别请求方法时可能会返回此代码。 502 （错误网关） 服务器作为网关或代理，从上游服务器收到无效响应。 503 （服务不可用） 服务器目前无法使用（由于超载或停机维护）。通常，这只是暂时状态。 504 （网关超时） 服务器作为网关或代理，但是没有及时从上游服务器收到请求。 505 （HTTP 版本不受支持） 服务器不支持请求中所用的 HTTP 协议版本。

### 网站渲染流程

6-1 浏览器与服务器交互

(4)响应永久重定向网址览器

(2)查找域名的IP网址览器

Web服务器

(1)输入网址网址览器

(3)发送HTTP请求 网址览器

浏览器

(6)返回对(5)/(3)响应，返回一个HTML响应个HTML

XIANGYING

(5)跟踪重定向地址,(3)网址览器

7) 浏览器开始显示HTML

8) 浏览器发送请求，以获取嵌入在HTML中的对象。

6-2 html、js、css解析

HTML

,SVG,

XHTM

Content Tree/

DOM Tree

P

DOM API

JavaScript

JS Tree

CSS

CSS rule tree

Rendering Tree/

Frame Tree

(incl. style)

Construct

（incl. Selector

matching）

graphics

API

Parse

Reflow/

layout览器

CSS API

\*\*解析HTML以构建DOM树 ——构建render树 —— 布局render树 ——绘制render树

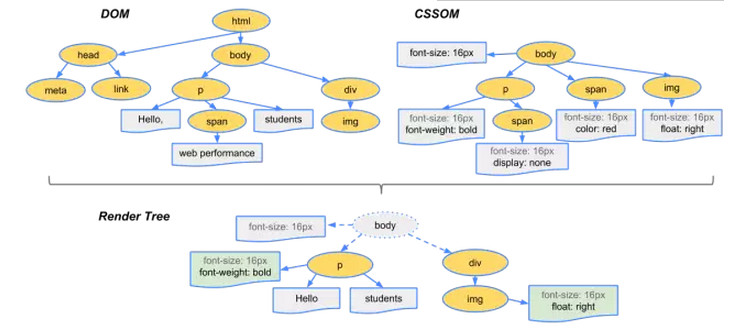
6-3通过html自上而下遍历来render tree

\* Rendering Tree 不等同于 DOM Tree，像 Header 或 display:none 就没放在渲染树中。

\* CSS 的Rule Tree 主要是为了完成匹配并把CSS Rule附加上 Rendering Tree 上的每个 Element(即 DOM 节点）

\* Render Tree，浏览器已知网页中有哪些节点、各个节点的CSS定义以及它们的从属关系。

\* 然后，计算每个 Frame（也就是每个Element）的位置，这又叫 layout 和 reflow 过程。



浏览器渲染的过程为了用户体验是逐步进行的(并不是整个html都解析完)，即解析完一部分就开始构建和渲染展示在屏幕。

(同时dom解析和网络下载资源)

6-5优化

CSS 匹配HTML 元素是一个相当复杂和有性能问题的事情。（DOM 树要小，CSS 尽量用id 和 class，千万不要过渡层叠下去）

\* dom深度尽量浅。

\* 减少inline javascript、css的数量。

\* 使用现代合法的css属性。

\* 不要为id选择器指定类名或是标签，因为id可以唯一确定一个元素。

\* 避免后代选择符，尽量使用子选择符。原因：子元素匹配符的概率要大于后代元素匹配符。后代选择符;#tp p{} 子选择符：#tp>p{}

\* 避免使用通配符，举一个例子，.mod .hd \*{font-size:14px;} 根据匹配顺序,将首先匹配通配符,也就是说先匹配出通配符,然后匹配.hd（就是要对dom树上的所有节点进行遍历他的父级元素）,然后匹配.mod,这样的性能耗费可想而知.

6-6 Repaint Reflow

\* Repaint（重绘） —— 屏幕一部分要重画，比如某个元素的背景颜色、文字颜色发生改变等，但是元素的几何尺寸没有变，并不影响元素周围和内容布局属性，将只会引起浏览器的 repaint，重画某一部分。  
\* Reflow（回流） —— 浏览器要花时间去渲染，当它发现了某个部分发生的变化影响了布局，即意味着元件的几何尺寸变了，我们需要重新验证并计算 Render Tree 进行重新渲染。Render Tree的一部分或全部发生了变化，这就是Reflow，或者说是Layout （HTML 使用的是 flow based layout，也就是流式布局，所以如果某元件的几何尺寸发生了变化，需要重新布局，也就是 Reflow）。reflow 会从 <html> 这个root frame 开始递归往下，依此计算所有节点的几何尺寸和位置。在 reflow 的过程中，可能会增加一些 frame，比如一个文本字符串必须被包装起来。