题目一：

方法一（浮动）

```
<!DOCTYPE html>

<html lang="en">



<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

    <style>

        * {

            margin: 0;

            padding: 0;

        }



        nav {
```

```
    width: 100%;

    height: 50px;

    line-height: 50px;

    background-color: red;

    text-align: center;

}



.left {

    height: 50px;

    width: 200px;

    float: left;

    background-color: #666;

}



.right {

    height: 50px;

    width: 150px;
```

```
            float: right;

            background-color: #666;

        }

    </style>

</head>



<body>

    <nav>

        <div class="left">欢迎你</div>

        <div class="right">退出登录</div>

    </nav>

</body>



</html>
```

方法二（flex 布局）

```
<!DOCTYPE html>

<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        nav {
            width: 100%;
            display: flex;
            justify-content: space-between;
            line-height: 50px;
```

```
            text-align: center;

            background-color: red;

        }



        .left {

            width: 200px;

            height: 50px;

            background-color: #666;

        }



        .right {

            width: 150px;

            height: 50px;

            background-color: #666;

        }

    </style>

</head>
```

```
<body>

    <nav>

        <div class="left">欢迎你</div>

        <div class="right">退出登录</div>

    </nav>

</body>



</html>
```

问题二：

```
<!DOCTYPE html>

<html lang="en">



<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>
```

```
</head>

<body>

</body>

<script>

    const list = [

        { lang: 'nodejs' },

        { lang: 'golang' },

        { lang: 'php' },

        { lang: 'nodejs' },

        { lang: 'php' },

        { lang: 'php' },

    ];

    let map = new Map();

    let count;
```

```
//遍历 list，统计 lang 出现的次数

for (let i = 0; i < list.length; i++) {

    if (map.get(list[i].lang)) {

        map.set(list[i].lang, map.get(list[i].lang) + 1);

    } else {

        map.set(list[i].lang, 1);

    }

}

//将 map 升序排序

let ans = [...map].sort((a, b) => {

    a[1] - b[1];

})



//将数组转化为对象

ans = Object.fromEntries(ans);



//输出 ans
```

```
    console.log(ans);



</script>



</html>



//方法二：ForEech

<!DOCTYPE html>

<html lang="en">



<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>
```

```
<body>



</body>

<script>

    const list = [

        { lang: 'nodejs' },

        { lang: 'golang' },

        { lang: 'php' },

        { lang: 'nodejs' },

        { lang: 'php' },

        { lang: 'php' },

    ];

    let ans = {}

    list.forEach(function (value, index) {

        if (ans[value.lang]) {

            ans[value.lang]++
```

```
        } else {

            ans[value.lang] = 1;

        }

    })

    console.log(ans)

</script>



</html>
```

问题三：

报错原因：localhost:5050 向 www.qq.com/news 发送请求违反了浏览器的同源策略。 如果两个 url 拥有相同的协议、域名和端口则为同源，否则，就是不同源。 解决方案：

1.JSONP

原理：JSONP 利用了 script 标签没有跨域限制的特性来完成； 用 vue.js 实现如下：

```
this.$http.jsonp('http//www.qq.com/news/recommended?page=1&per_page=5',{

            params:{},

            jsonp:'fn'
```

```
}).then((res) => {

    console.log(res)

})
```

2.反向代理

```
原理：利用服务端到服务端之间没有同源策略

vue 框架中 webpack.config.js 做如下配置

module.exports ={

    devServer:{

        proxy:[{

            target:'http//www.qq.com/', //代理跨域目标接口

            changeOrigin:true,

        }]

    }

}
```

问题四：

fiddler 如何抓 https 包的环境搭建：

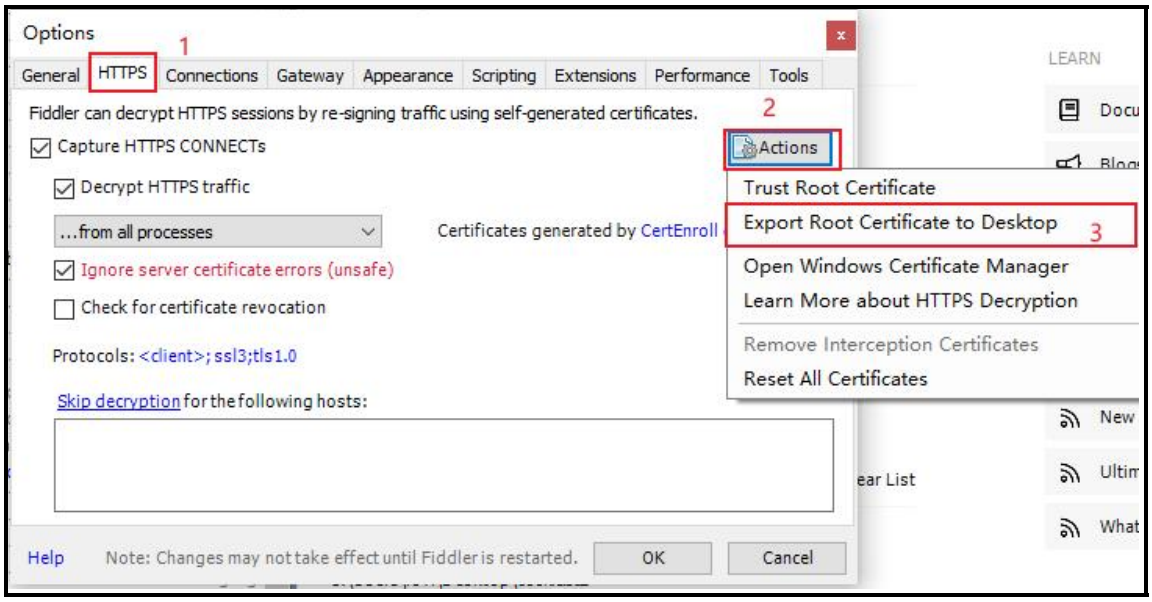
1.下载最新版 fiddler，推荐在官网下载：

https://www.telerik.com/download/fiddler

2.安装 fiddler，傻瓜式安装。

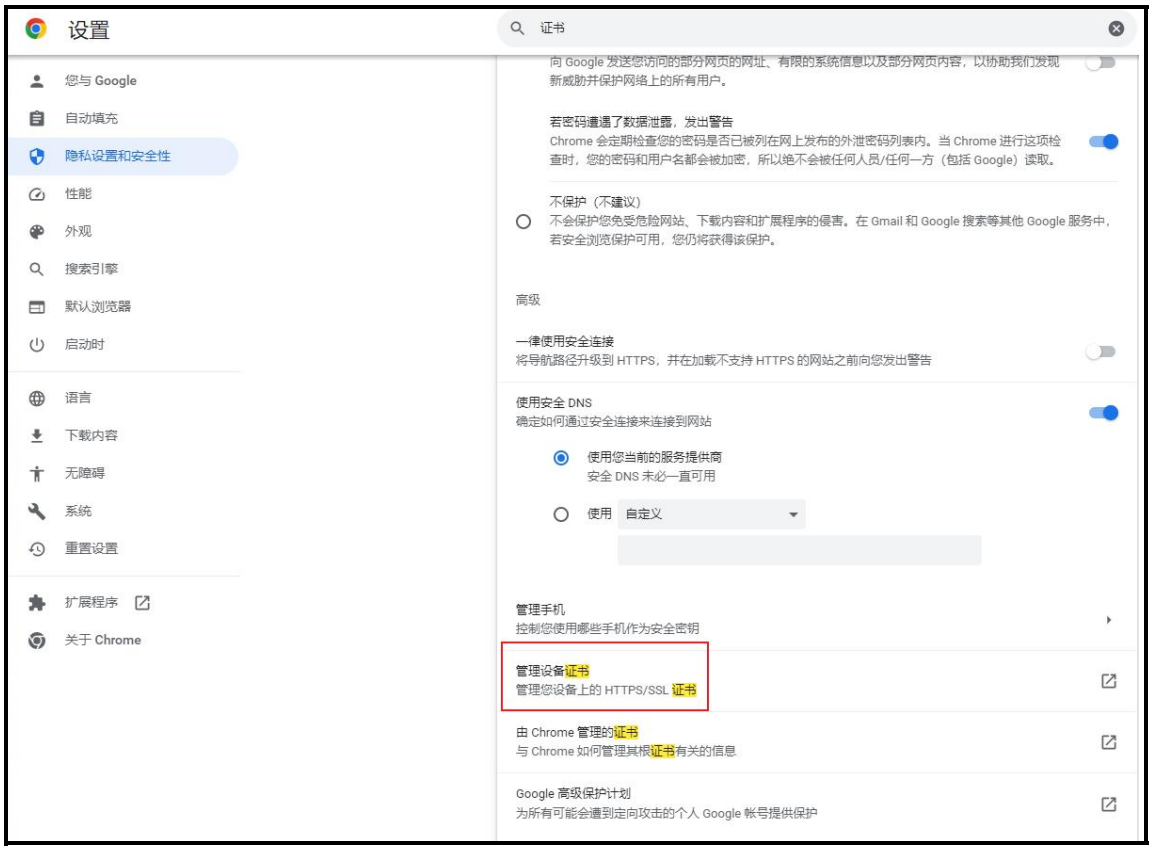
3.打开 fiddler，并进行如下步骤：点击菜单中的 Tools -> Options；出现如下窗口。

如图按次序点击，将证书文件下载导出至桌面。并将窗口选项勾选至如下图。

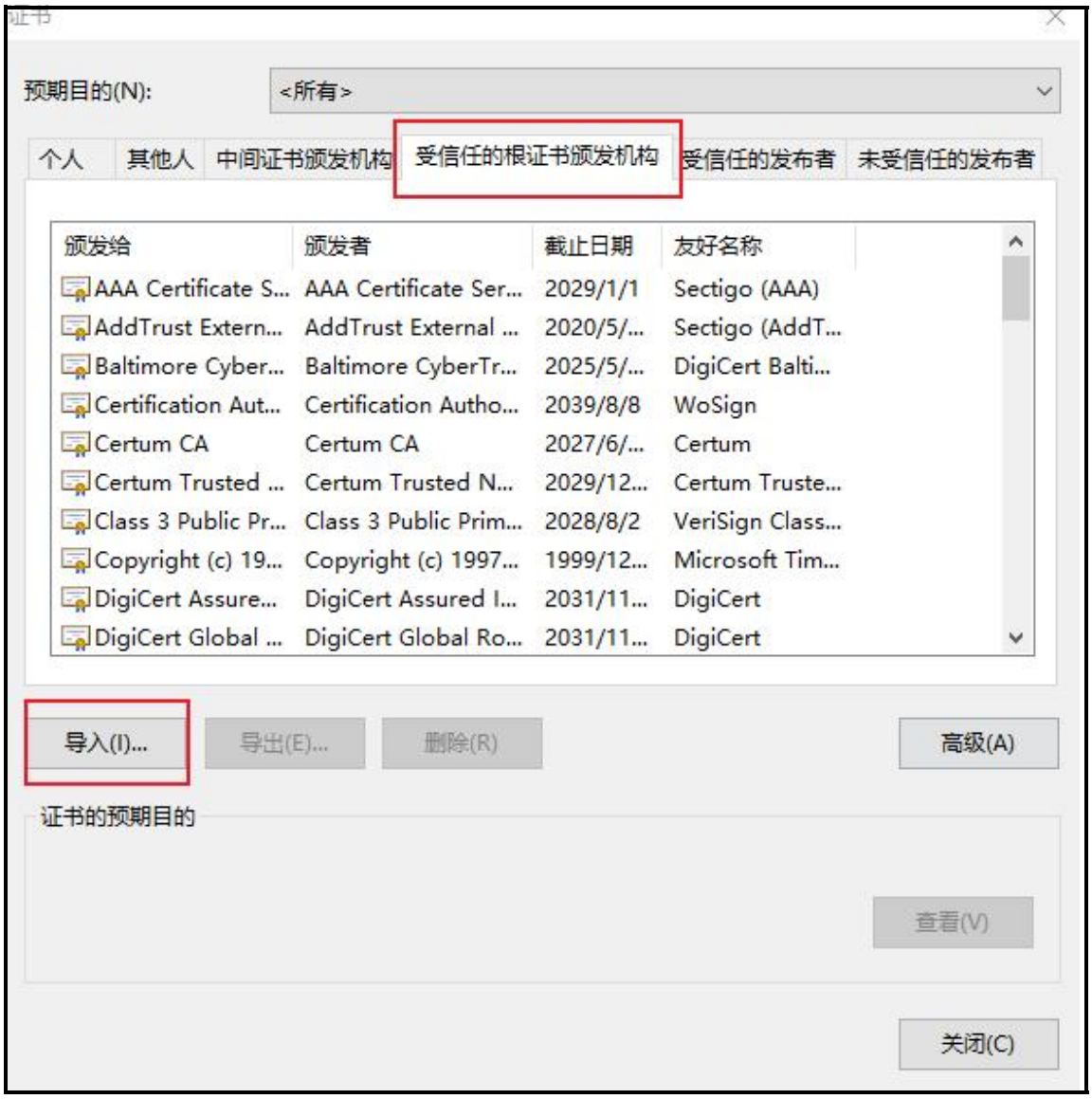


4.在浏览器中导入证书 FiddlerRoot.cer。以谷歌浏览器为例，在流浪器输入：chrome://settings/，

搜索到管理证书。如图：



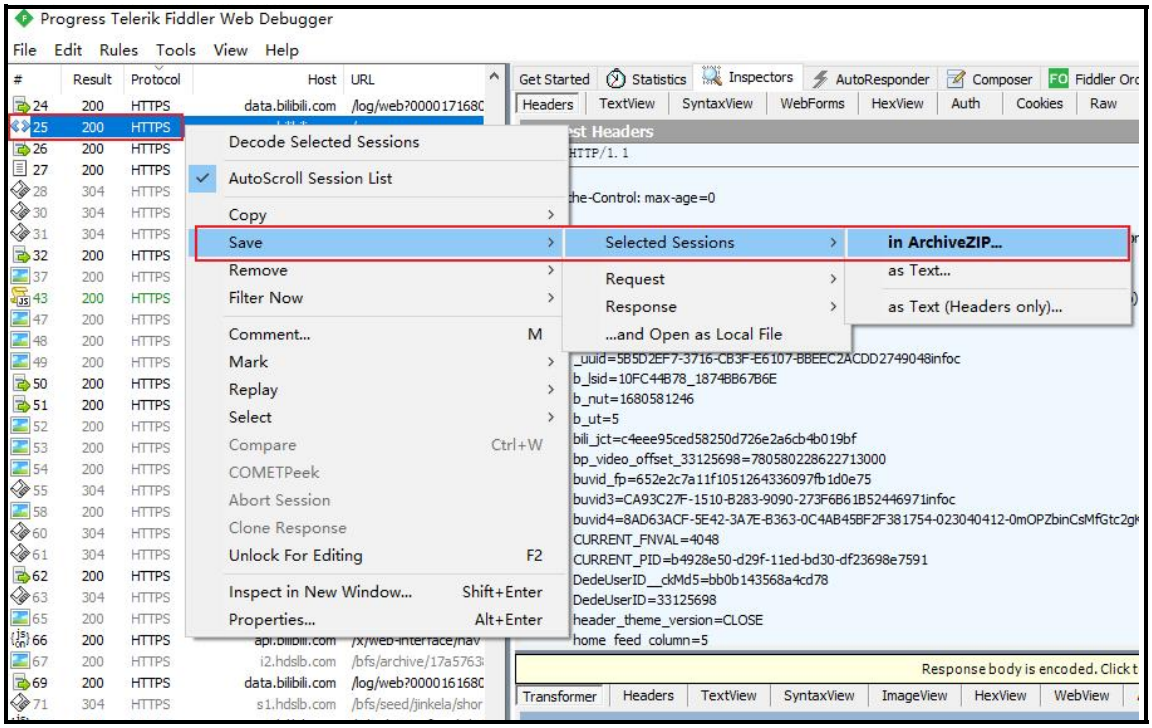
点击进入管理证书页面，在菜单中选择受信任的根证书颁发机构导入证书 FiddlerRoot.cer。位置如图：

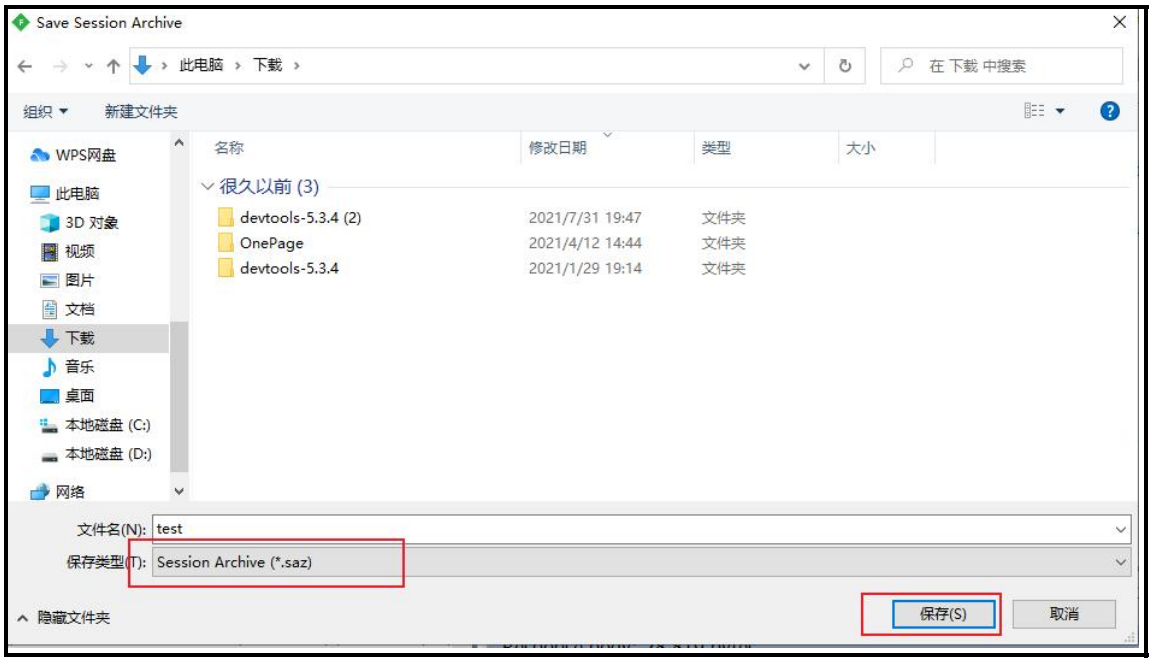


5.重新启动 fiddler 就可以进行 https 的抓包了

如何将请求保存为 Saz 格式：

1.打开 fiddler，并打开你想抓包的网页。在软件右方列表选择你要保存抓包的请求，进行如下图操作：





这样就将请求保存为 Saz 格式了。

如何修改/删除 Saz 格式文件中的请求头：

基本原理：百度百科

SAZ文件是一种特殊格式的 .ZIP文件。可以将.SAZ后缀修改为.ZIP, 然后使用标准的ZIP压缩工具打开。

在SAZ文件内，可以找到：

1. _index.htm： 一个可选文件，便于用户直接阅读。这个文件在SAZ文件被加载时不会被处理，仅仅用于手工检查。

2.raw文件夹：包含所有的记录了网络Sesison信息的文件。

在raw文件夹内，对于每一个网络Sesison，会有三个文件：

sessid#_c.txt：包含客户端请求

sessid#_s.txt： 包含服务器端请求

sessid#_m.txt：包含一些元数据，比如：Session标志位，socket重用信息，等等。

操作步骤：

1.将 saz 文件后缀改为 zip 格式。

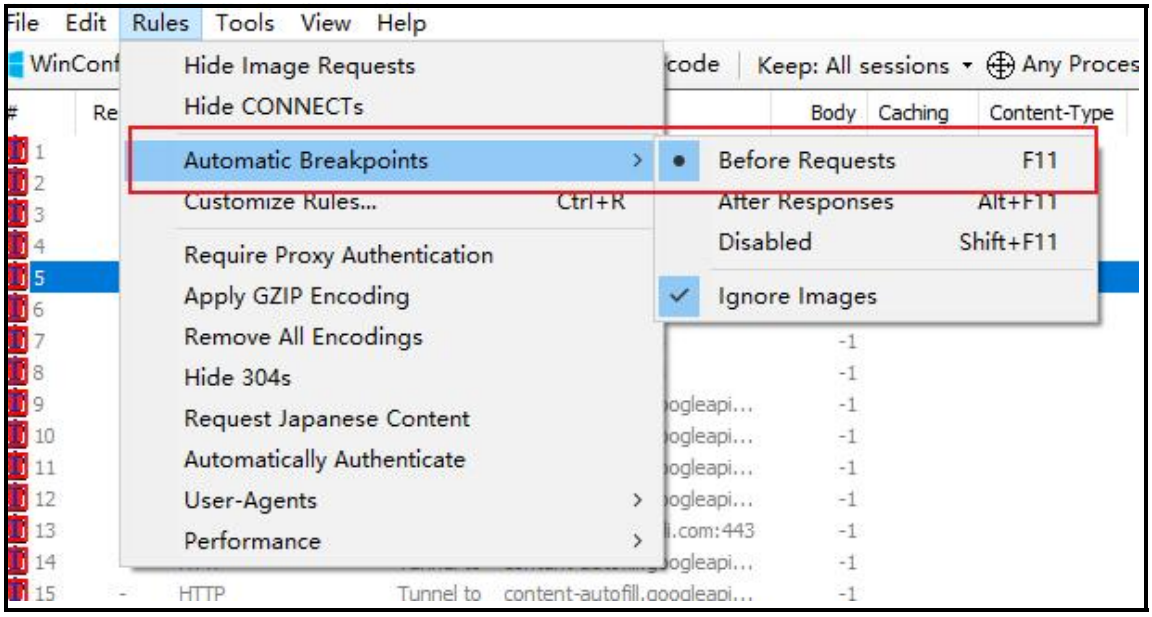
2.解压并打开 raw/*_c.txt；找到 Cookie 列，将想要删除的隐私 cookie 删除。保存。

3.将修改后的文件夹重新压缩，并将后缀名改回 saz 就完成了对 saz 格式文件的修改。

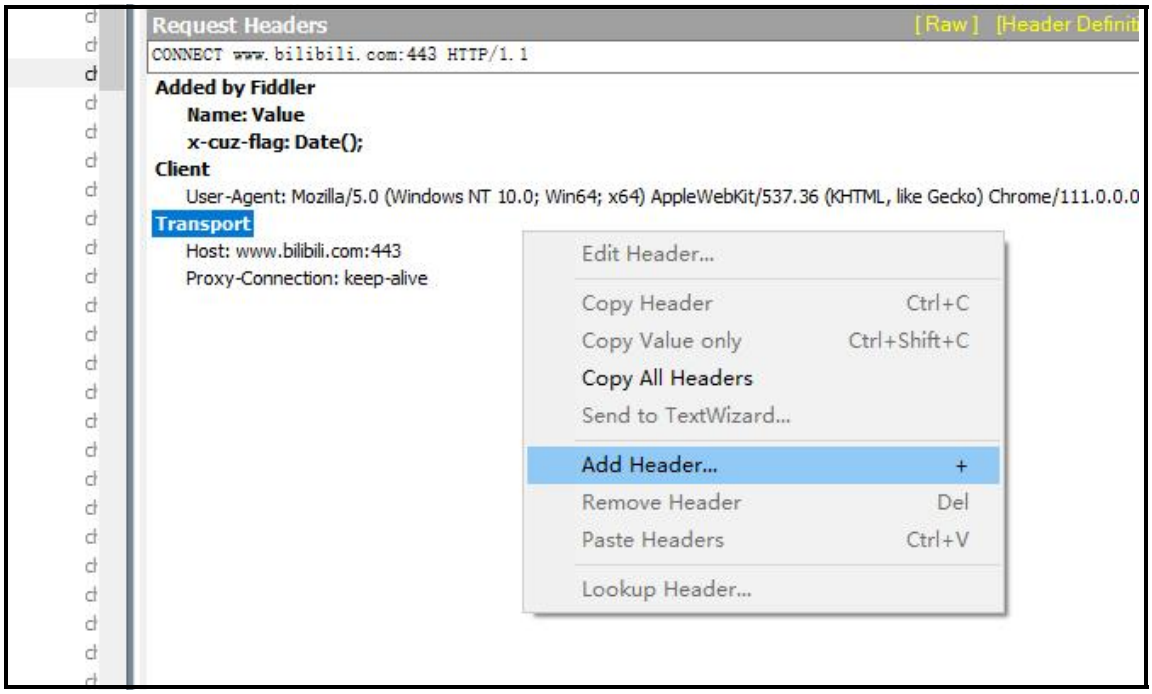
如何在请求头中添加请求行：

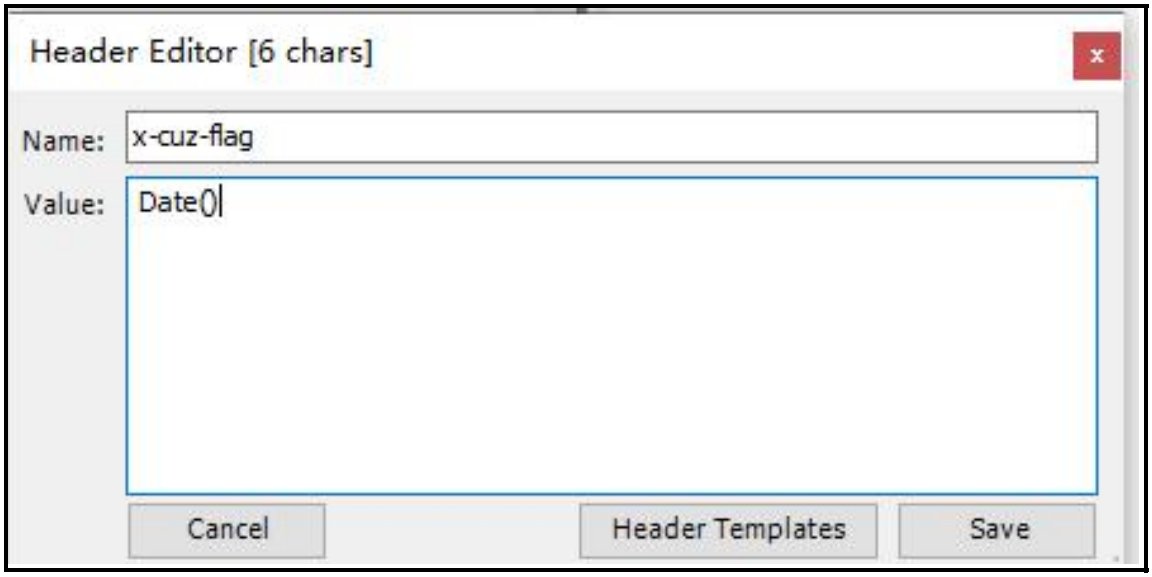
为了演示方便，这里采取了全局断点的模式来演示。

1.设置断点，将断点在请求还未发出前。



2.选择你想要为其添加请求行的请求，添加请求行。





如上图操作，请求行就添加好了。

问题五：

https://stackblitz.com/edit/vue3-typescript-vue-cli-starter-pxxz6w?file=src/components/List.vue