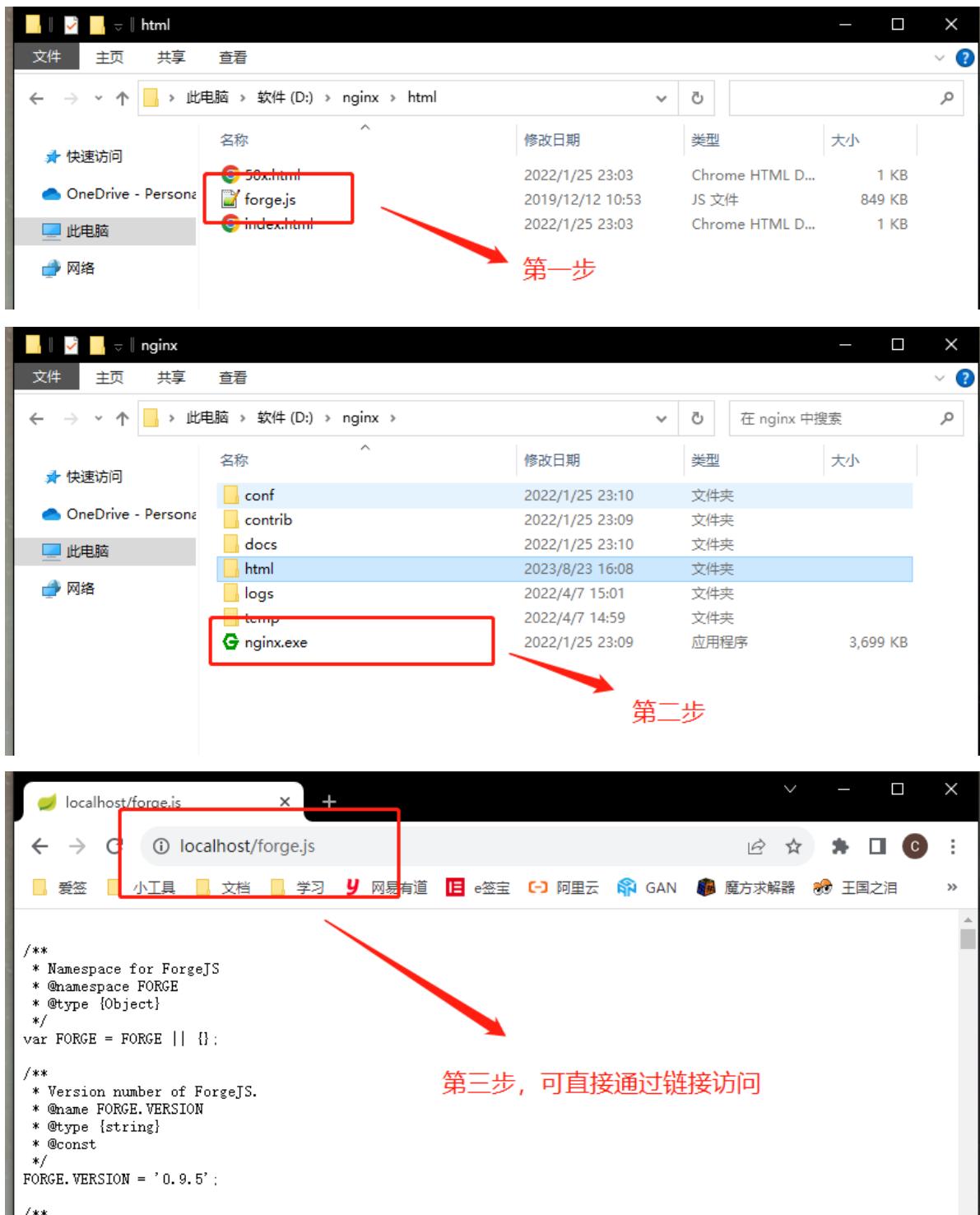
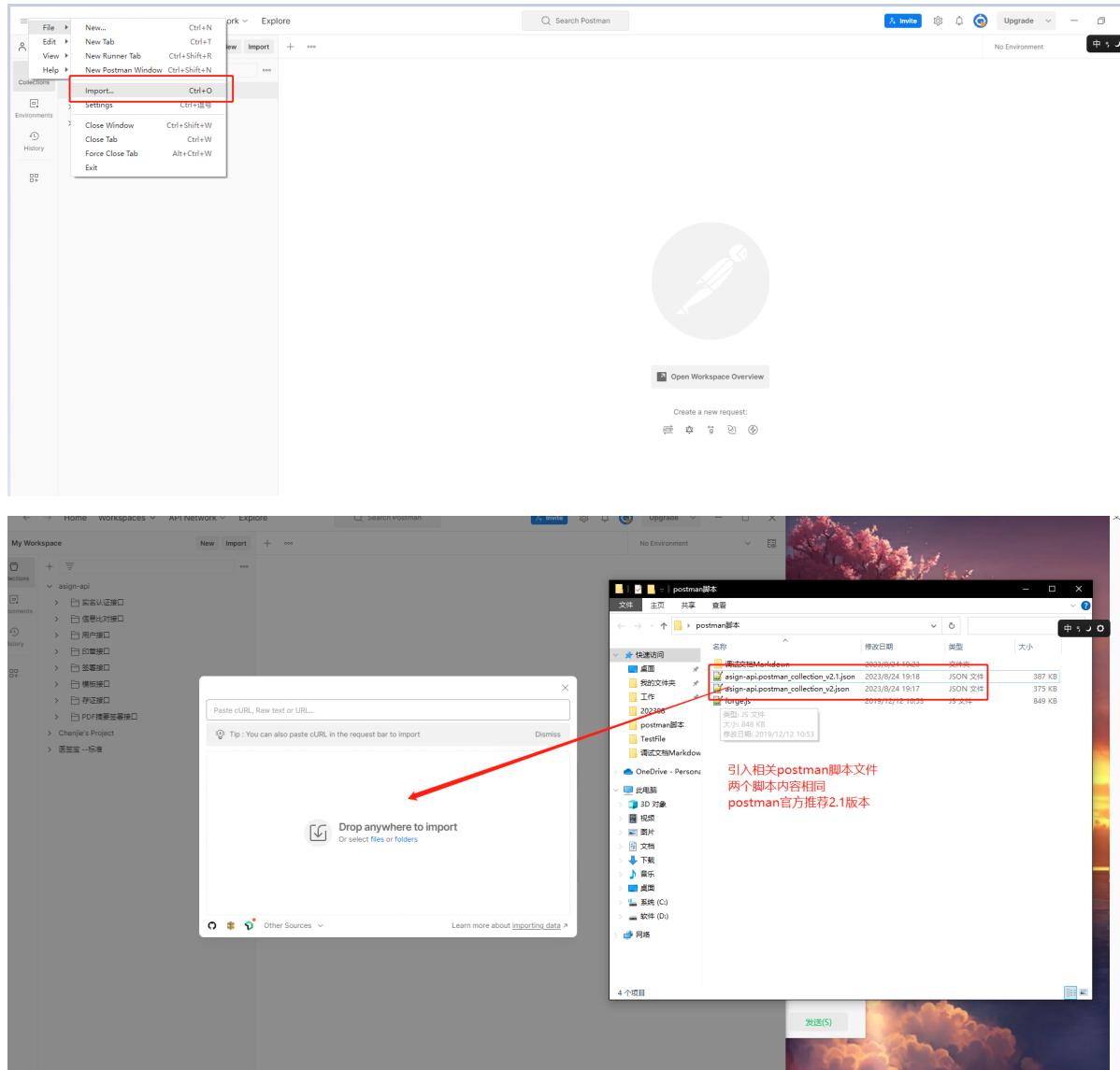


1. 将forge.js配置成可访问资源

1. 因目前 Postman 自带的 cryptoJS 不支持爱签当前签名加密方式，故需要额外引入 forge.js 来实现签名加密。
2. 因 Postman 前置脚本不支持直接引入 js 文件，所以需要将 forge.js 配置成可访问资源，由 Postman 前置脚本读取引入。
3. 将 forge.js 配置成可访问资源有多种方式，下面以 Nginx 为例，直接将 forge.js 放入 Nginx 目录下的 html 文件夹中，并启动 Nginx



2.Postman引入脚本



3.脚本变量

The screenshot shows the Postman interface with the 'asign-api' collection selected. The 'Variables' tab is highlighted with a red box and an arrow pointing to it from the top right. A second red box highlights the table below, which lists the following variables:

Variable	Initial value	Current value
appId	{appId}	{appId}
url	{url}	{url}
privateKey	{privateKey}	{privateKey}
timestamp		
sign		
bizData		
forgeUrl	{forgeUrl}	{forgeUrl}

- appId (需填充) : 开放平台分配的应用 appId
- url (需填充) : api接口地址, 测试: <https://prev.asign.cn>, 生产: <https://api.asign.cn>, <https://oapi.asign.cn>
- privateKey (需填充) : 秘钥前后缀必须

```
-----BEGIN PRIVATE KEY-----  
{秘钥}  
-----END PRIVATE KEY-----
```

- forgeUrl (需填充) : 第一步配置的 forge.js 地址
- timestamp (不填充) : 时间戳
- sign (不填充) : 签名加密字符串
- bizData (不填充) : 业务请求数据

4.发起请求

1. 选择相应接口, 进入前置脚本选项 (Pre-request Script), 找到 getBizData() 方法, 对接口文档中设置的相关请求数据进行填充
2. 接口文档中 `MultipartFile` 和 `List<MultipartFile>` 类型参数, 在 `body -> form-data` 单独添加
3. 下面以 OCR身份证识别 接口为例

```

1 // 生成请求参数
2 function getBizData() {
3     const param = {};
4     param.side = 'front'; // 身份证正反面
5     return param;
6 }
7 // 读取文件并转换为base64
8 const bizData = getBizData();
9 console.log('请求参数：' + JSON.stringify(bizData));
10 console.log('文件内容：' + JSON.stringify(getFilepath()));
11 // 生成签名
12 const jsonString = JSON.stringify(JSON.parse(bizData));
13 pm.variables.set('bizData', jsonString);
14 pm.variables.set('signString', jsonString);
15 const timestamp = Date.now() + 3600 * 60 * 10;
16 console.log('时间戳：' + timestamp);
17 const signString = timestamp + timestamp;
18 // 生成签名的请求头
19 const sign = signature(jsonString, timestamp);
20 console.log('签名：' + sign);
21 pm.variables.set('sign', sign);
22 console.log("Postman设置脚本执行结束, 开始生成请求");
23
24 // 签名
25 function signature(jsonString, timestamp) {
26     const signString = jsonString +

```

Key	Value	Description
appId	(appId)	
timestamp	(timestamp)	
bizData	(bizData)	
image	2.png	

5. 请求实例

Key	Value	Description
serialNo	(serialNo)	
data	null	

Response body:

```

1 // 读取文件并转换为base64
2 function getBizData() {
3     const param = {};
4     param.serialNo = "PA15020230024102404411159"; // 认证码
5     return param;
6 }
7 // 读取文件并转换为base64
8 const bizData = getBizData();
9 console.log('请求参数：' + JSON.stringify(bizData));
10 console.log('文件内容：' + JSON.stringify(getFilepath()));
11 // 生成签名
12 const jsonString = JSON.stringify(JSON.parse(bizData));
13 pm.variables.set('bizData', jsonString);
14 pm.variables.set('signString', jsonString);
15 const timestamp = Date.now() + 3600 * 60 * 10;
16 console.log('时间戳：' + timestamp);
17 const signString = timestamp + timestamp;
18 // 生成签名的请求头
19 const sign = signature(jsonString, timestamp);
20 console.log('签名：' + sign);
21 pm.variables.set('sign', sign);
22 console.log("Postman设置脚本执行结束, 开始生成请求");
23
24 // 签名
25 function signature(jsonString, timestamp) {
26     const signString = jsonString +

```