V8 调试环境配置指南

> 这份指南仅涉及 Chrome V8 各版本调试环境的配置，**不会包含任何题目解法相关的提示信息**，V8 本地编译对于本题的解题**不是必要的**，题目附件中已经提供了 patch 后的可执行程序。

如果你对 V8 感兴趣的话，可以选择：

- 直接在本地配置
    - 需要保证网络畅通，否则会出现各种各样奇怪的错误
- 使用打包好的镜像

由于前一种方案网上已有大量详细的教程，在此就不赘述了，本文写给希望配置 V8 调试环境却被~~网络问题~~（千奇百怪的问题）所折磨的读者，提供了 docker 上配置环境的替代方案：

在以下镜像中（点击前往下载，sha256：4643480cf0cf39f45f09343b7d1c8779a73a9fc5117243d846d6bb18341e53b5）已经准备好了编译不同版本 V8 所需要的环境（文件较大，配置较花时间和空间，请自主判断是否下载），你只需要在终端中进行如下操作（默认已有 docker 环境）：

```
#  导入镜像
## 先来到下载文件的目录下，执行（最后一项为自定义的镜像标签）：
## 由于镜像较大（8.6G），需要等待一段时间
docker import v8debug.tar.xz v8debug

## 进入 bash
docker exec -it $(docker run -dit v8debug bash) bash
```

现在已经进入了配置好的 docker 环境中，可以切换进 v8 目录下，运行：

```
$ ./out.gn/x64.debug/d8
V8 version 9.0.257.19
d8>
```

发现提前编译好的 d8 运行成功。

以下是做题时的 patch 步骤（先退出容器回到主机）：

```
# 先找到之前的容器：
docker ps -a

# 根据上一步的容器 id，将需要 patch 的文件复制进去：
docker cp ./evilCallback.diff $(上一步找到的容器 id):/root/
docker exec -it $(上一步找到的容器 id) bash

# 进入容器后：
cd root/v8
git apply < ../evilCallback.diff

# 编译（这一步比较花时间，可以直接用目标目录下已经编译好的文件进行调试，服务器上的题目以题目附件为准）
## debug 版本
tools/dev/v8gen.py x64.debug
ninja -C out.gn/x64.debug d8
## release 版本
tools/dev/v8gen.py x64.release
ninja -C out.gn/x64.release d8
```

注意：以题目附件为准，也建议将编译好的 d8 及其快照文件 snapshot_blob.bin 拷贝到 docker 外运行，以确保运行过程中内存表现一致。