# Git webhook以及流水线的实现

因为在开发的过程中，需要去查看实际的开发效果，就需要去打包部署这些服务。每次手动打包的过程比较耗时而且非常繁琐，所以设计了一个流水线来对应用进行打包部署。原理就是基于git webhook对git 的提交进行监听，如果进行了 git push 这一步，那么仓库会响应相应的事件。具体实现如下所示。

|  |
| --- |
| const { execSync } = require('child\_process');  const http = require('http');  const readline = require('readline');  const { Buffer } = require('buffer');  const hostname = '127.0.0.1';  const port = 3000;  const fs = require("fs");  const desPath = '/usr/local/exam-fe';  const execBash = 'git clone git@gitee.com:simpleling/exam-fe.git && cd exam-fe && npm i && npm run build';  function deleteFolder(path) {  let files = [];  if( fs.existsSync(path) ) {  files = fs.readdirSync(path);  files.forEach(function(file,index){  let curPath = path + "/" + file;  if(fs.statSync(curPath).isDirectory()) {  deleteFolder(curPath);  } else {  fs.unlinkSync(curPath);  }  });  fs.rmdirSync(path);  }  }  const server = http.createServer(async (req, res) => {  if (req.url === '/') {  const start = new Date().getTime();  // deleteFolder(desPath);  try {  // await execSync(execBash);  const end = new Date().getTime();  const data = new Uint8Array(Buffer.from(`${start}-${end}\n`));  fs.appendFile('log.txt', data, err => {  console.log(err)  });  res.statusCode = 200;  res.setHeader('Access-Control-Allow-Origin', '\*');  res.end('success\n');  } catch (error) {  res.statusCode = 500;  res.setHeader('Access-Control-Allow-Origin', '\*');  res.end('Interal Error\n');  }  } else if (req.url === '/getList') {  var readStream = fs.createReadStream('log.txt');  const rl = readline.createInterface({  input: readStream  });  const arr = [];  rl.on('line', line => {  arr.push(line);  });  rl.on('close', () => {  res.statusCode = 200;  res.setHeader('Access-Control-Allow-Origin', '\*');  res.end(JSON.stringify(arr));  });  }  });  server.listen(port, hostname, () => {  console.log(`Server running at http://${hostname}:${port}/`)  }); |

