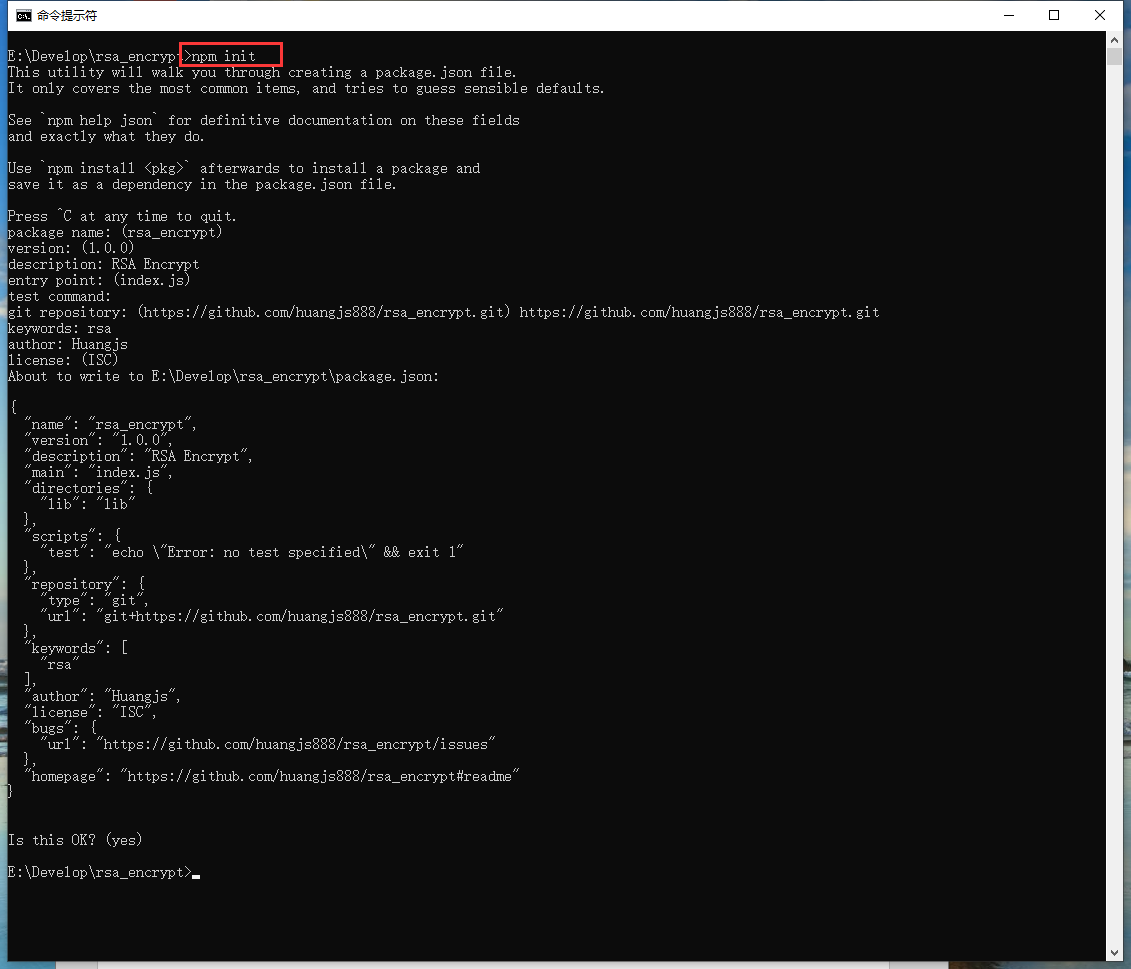
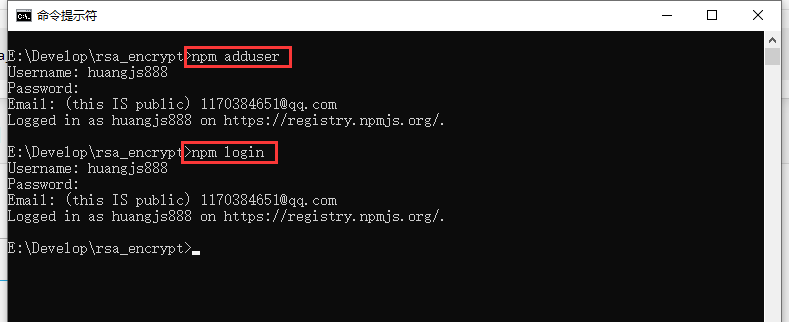
前提是已经装好node和npm

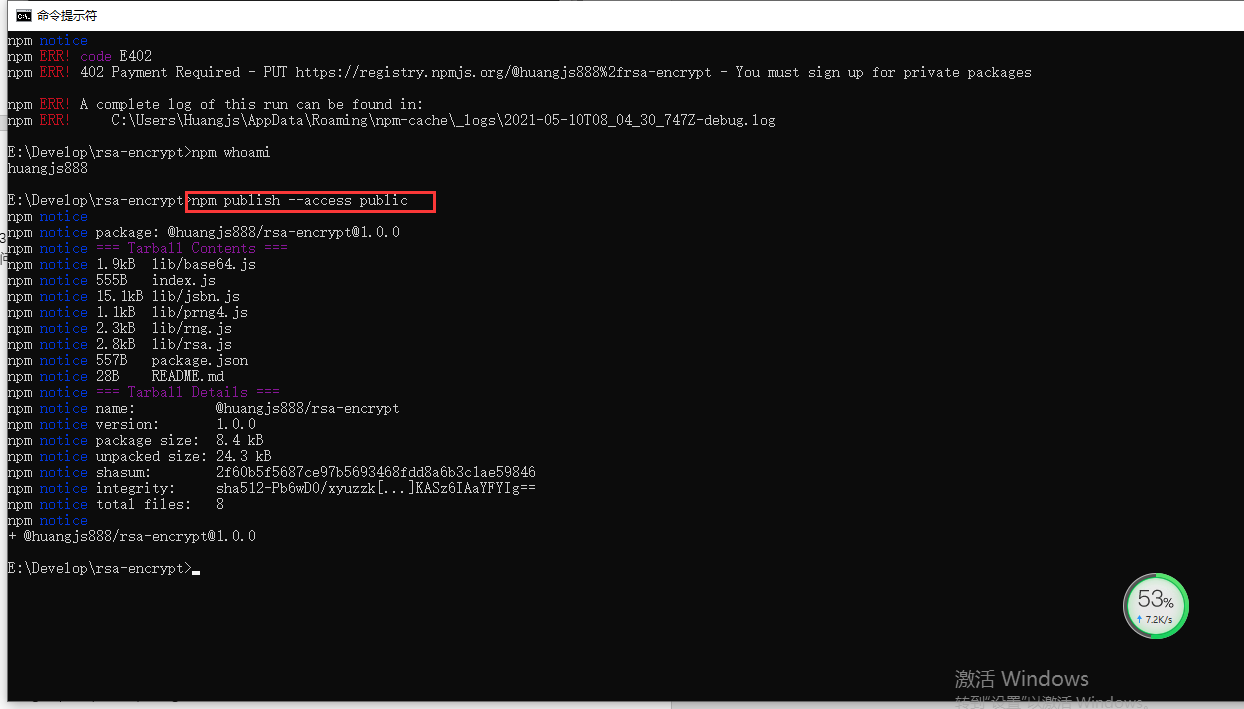
1. 命令行进入自己要发布的项目，如果该项目没有package.json，则需要npm init输入相关信息创建该文件。



1. 需要注册一个npm的帐号，可以进入官网<https://www.npmjs.com/>进行注册，也可以使用命令npm adduser，依次填写好用户名，密码以及邮箱，成功后会发邮件到邮箱验证，进入邮箱点击链接验证，然后npm login依次写入信息登录（因为我这里已经注册过了，所以没有注册成功标志）。已经有账号的，直接登录即可。这里注册或登录的时候可能出现错误，比如400，500之类的，记得将npm配置的相关镜像先删除，具体是在操作系统/user/{account}/.npmrc文件，也可以命令行修改，npm config set registry https://www.npmjs.com



1. 最后使用命令 npm publish –access public进行发布。



问题集：

如果出现403 Forbidden（You do not have permission to publish或Package name too similar to existing packages等） 可能是你的项目名在npm上已经被占用，此时如果去npm搜索该模块名称，可以发现已有该模块。所以发布前可以先搜索一下，如果搜索不到再使用，如果存在，则修改名称，直接修改package.json里的name即可。

403 Forbidden

按照文档中的步骤，注册登录后，准备好了库，然后执行 npm publish，报了下面的错误：

Error: 403 Forbidden - PUT https://registry.npmjs.org/xx - You do not have permission to publish “xx”. Are you logged in as the correct user?

查询资料后，发现了这个 issue，原来是因为公开的库里面，已经有了同名的库。由于是免费的账号，所以想在 npm 上发包，要么换个名称，要么花钱创建私有包。还有一种方式就是在内网搭建自己的服务。

Error: 402 Payment Required

换了个名字，想起了自己见过的库，就模仿加了个 @ 前缀的包，再次发布的时候，报了下面的错误：

Error: 402 Payment Required - PUT https://registry.npmjs.org/@xx/xx - You must sign up for private packages

查看文档，发现 @npm/package-name 这种形式的包名，是有作用域的包名形式，执行 npm publish 的时候默认是发布私有的包。因此，第一种方式是花钱买私有包的服务，另外一种方式就是指定参数，表示公开：

npm publish --access public

1

需要注意的是这种形式的包名跟 npm 账户有对应关系，不能随便填写。

npm init --scope=@my-org

1

这种形式表示是一个组织，my-org 对应是 npm 中的组织名。

npm init --scope=@my-username

1

这种形式表示是个人，my-username 对应是 npm 中的用户名。

对旧包的处理

在尝试的过程中，发布了一些包，想着只是测试的包，想要移除掉，但 npm 官方很不推荐这种做法，推荐用 npm deprecate 指令，告知安装者相关信息，例如“我不再维护这个库了，请不要使用”。

npm deprecate package-name "This is test package, do not use it!"

1

其中 package-name 是指发布的包的名称，可能跟库的名称不一样。

发布成功了，但在 npm 上找不到包

在一次发布中，看到发布成功的提示，但到 npm 账户发现没有相关的包。

反复尝试了几次，发现原来是因为自己本地 registry 指向了内网的一个地址，并没有指向 npm 官方的地址 https://registry.npmjs.org/ 。

所以发包的时候，先确定一下发布的指向：

npm config get registry

或

npm config list

1

2

3

如果发现指向不对，有两种修改的方式：

方式 1：使用 npm config set registry 改成对应的指向，但这个是全局修改，之后如果需要发布其它指向的包，又要修改回来。

方式 2：执行指令时添加 --registry 参数

在 package.json 中添加参数：

"publishConfig": { "registry": "https://npm.pkg.github.com/" }

1

登录的时候，添加对应的参数：

npm login --registry=https://registry.npmjs.org/

1

在使用 npm deprecate 指令的时候，如果本地 registry 指向不对，也要添加 --registry 参数才会有效。

如何判断 npm 账户是否已登录

由于发布的包有不同的 registry 指向，在发布包的时候，登录要指向不同的源，所以发布之前想要看看是否已登录。

查看了文档，没有找到查看登录状态的特定命令，但可以使用能触发登录检查的命令，从侧面看是否已登录。

很容易想到的就是 npm login 命令，但登录过一次后，再执行时还是提示输入账号密码。

后来发现了一个简单的指令 npm whoami ，这个指令只能在登录状态下执行，没有登录就会提示：

npm ERR! code ENEEDAUTH

npm ERR! need auth This command requires you to be logged in.

npm ERR! need auth You need to authorize this machine using npm adduser

需要注意的是，这指令不带 registry 参数时，会指向 npm 默认的 registry ，如果这个不匹配，也是没有效果的。

当有多个 registry 时，一定要注意，不少的指令都需要一致的 registry 才有效。

更新发布的包

先查看下已经发布的包是什么版本：

$ npm view dylan-browser-info versions

[ '1.0.0' ]

提交修改后的版本

使用命令：npm version <update\_type>

update\_type 有三个参数

第一个是patch, 第二个是minor,第三个是 major

patch：这个是补丁的意思，就是小小的改一下

minor：这个是小修小改

major：这个是大改

使用：

如果说要改为1.0.1，最后一位增加了1，命令：npm version patch

如果说要改为1.1.1，中间一位增加了1，命令：npm version minor

如果说要改为2.1.1，第一位增加了1，命令：npm version major

这样操作只是在本地修改了版本，远端的npm仓库还没有修改还要使用下面的命令更新远端的：

npm publish

也可以手动的修改版本号，直接在package.json中修改version，然后npm publish