1、特点：事件驱动、非阻塞IO模型（异步），轻量和高效

2、模块查找规则-当模块拥有路径但没有后缀时：

require('./find.js') ;

require('./ find') ;

1） require方法根据模块路径查找模块，如果是完整路径，直接引入模块。

2）如果模块后缀省略, 先找同名JS文件再找同名文件夹。

3）如果找到了同名文件夹，找文件夹中的index.js。

4）如果文件夹中没有index.js就会去当前文件夹中的package.json文件中查找main选项中的入口文件。

5）如果找指定的入口文件不存在或者没有指定入口文件就会报错,模块没有被找到。

3、关于package.json文件：

当把项目给别人传递过去之后，只要把package.json也传过去，再用npm

关于依赖的说明：

1）在项目的开发阶段和线上运营阶段,都需要依赖的第三方包，称为项目依赖，

使用npm install包名命令下载的文件会默认被添加到package.j

2）在项目的开发阶段需要依赖，线上运营阶段不需要依赖的第三方包,称为开发依赖，

使用npm install包名--save-dev命令将包添加到packag

3）部署在服务器上只下载项目依赖，在开发时用的开发依赖是不需要下载的。

‘npm install --production’安装项目依赖

‘npm install’安装所有依赖

4）package.json里"script"属性可以把一些命令存储起来，这样

关于package-lock.json文件：

锁定包的版本，确保再次下载时不会因为包版本不同而产生问题；

加快下载速度，因为该文件中已经记录了项目所依赖第三方包的树状结构和包的下载地址

重新安装时只需下载即可，不需要做额外的工作。

部分第三方模块：

nodemon：

实时监听js文件并自动重新执行

nrm:

切换npm源,’nrm ls‘列出源列表，’nrm use “源名称”‘更改源

3、模块查找规则-当模块没有路径且没有后缀时

require('find') ;

1）Node.js会假设它是系统模块；

2）Node.js会 去node modules文件夹中；

3）首先看是否有该名字的JS文件；

4）再看是否有该名字的文件夹；

5）如果是文件夹看里面是否有index.js；

6）如果没有index.js查看该文件夹中的package.json中的main选项确定模块入口文件；

7）否则找不到报错。

4、异步函数

async关键字

1.普通函数定义前加async关键字普通函数变成异步函数

2.步函数默认返回promise对象

3.在异步函数内部使用return关键字进行结果返回结果会被包裹的promise对象中return关键字代替 了resolve方法

4.在异步函数内部使用throw关键字抛出程序异常

5.调用异步函数再链式调用then方法获取异步函数执行结果

6.调用异步函数再链式调用catch方法获取异步函数执行的错误信息

await关键字

1. await关键字只能出现在异步函数中

2. await promise await后面只能写promise对象写其他类型的API是不不可以的

3. await关键字可是暂停异步函数向下执行直到promise返回结果