#### 1、NodeJS中比较常用的日志模块

(1)Winston

它是一个专门用于管理记录node的日志库，在githu中是fork最多的。

Winston被设计为与多种传输支持一个简单而通用的日志库。winston记录器的每个实例都可以有不同程度的配置了多个位置。

(2)Bunyan (一个用于node.js服务的JSON日志库)

(3)log4js

node有log4js模块，我们可以直接安装

log4js可以与express框架完美结合起来

log4js可以将日志分为不同层级的异常或者错误，我们可以灵魂掌握

log4js还有挺多好玩的内容，比如smtp。这是个很有用的功能，比如当项目发生某个错误时，你希望程序能发邮件通知你，该功能就能派出用场了。

Winston和Bunyan都是很成熟的两个框架，Winston有一个强大的社区支持，而Bunyan使得日志的进一步系统分析处理非常方便。

**日志级别**

log4js共有6种日志级别，分别为：trace（追踪）、debug、info、warn、error、fatal。权值从小到大

如果输出的级别是info，则不会打印低于info级别的日志（trace跟踪和debug调试）

// 配置规则

/\*\*

 \* 第一种：

 \* configure方法为配置log4js对象，内部有levels、appenders、categories三个属性

 \* levels:

 \*         配置日志的输出级别,共ALL<TRACE<DEBUG<INFO<WARN<ERROR<FATAL<MARK<OFF八个级别,default level is OFF

 \*         只有大于等于日志配置级别的信息才能输出出来，可以通过category来有效的控制日志输出级别

 \* appenders:

 \*         配置文件的输出源，一般日志输出type共有console、file、dateFile三种

 \*         console:普通的控制台输出

 \*         file:输出到文件内，以文件名-文件大小-备份文件个数的形式rolling生成文件

 \*         dateFile:输出到文件内，以pattern属性的时间格式，以时间的生成文件

 \* replaceConsole:

 \*         是否替换控制台输出，当代码出现console.log，表示以日志type=console的形式输出

 \*

 \*/

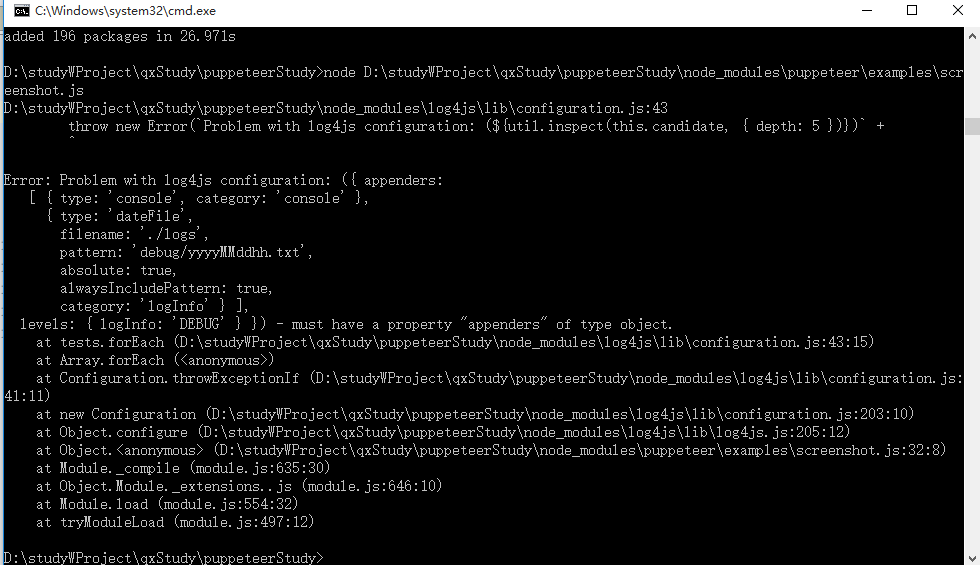
log4js.json文件：

1. {
2. "appenders": [
3. // 下面一行应该是用于跟express配合输出web请求url日志的
4. {"type": "console", "category": "console"},
5. // 定义一个日志记录器
6. {
7. "type": "dateFile",                 // 日志文件类型，可以使用日期作为文件名
8. 的占位符
9. "filename": "e:/weblogs/logs/",     // 日志文件名，可以设置相对路径或绝对路径
10. "pattern": "debug/yyyyMMddhh.txt",  // 占位符，紧跟在filename后面
11. "absolute": **true**,                   // filename是否绝对路径
12. "alwaysIncludePattern": **true**,       // 文件名是否始终包含占位符
13. "category": "logInfo"               // 记录器名
14. } ],
15. "levels":{ "logInfo": "DEBUG"}        // 设置记录器的默认显示级别，低于这个级别的日志，不会输出
16. }

**注意：json文件不支持任何形式的注释// 或 /\*\*/，因此上面的配置代码里的注释要全部移除，不然会编译出错的**

下载：npm install log4js

运行报错：



v1.x与v2.x配置规则发生变化，appenders及categories是一个对象，应改成如下：

log4js.configure('./config/log4js.json');

log4js.json

{

  "appenders": {

    "access": {

      "type": "dateFile",

      "filename": "log/access.log",

      "pattern": "-yyyy-MM-dd"

    },

    "rule-console": {

      "type": "console"

    },

    "rule-file": {

      "type": "dateFile",

      "filename": "log/server-",

      "encoding": "utf-8",

      "maxLogSize": 10000000,

      "numBackups": 3,

      "pattern": "yyyy-MM-dd.log",

      "alwaysIncludePattern": true

    },

    "rule-error": {

      "type": "dateFile",

      "filename": "log/error-",

      "encoding": "utf-8",

      "maxLogSize": 1000000,

      "numBackups": 3,

      "pattern": "yyyy-MM-dd.log",

      "alwaysIncludePattern": true

    }

  },

  "categories": {

    "default": {

      "appenders": [

        "rule-console",

        "rule-file",

        "rule-error"

      ],

      "level": "debug"

    },

    "http": {

      "appenders": [

        "access"

      ],

      "level": "info"

    }

  }

}

// 第二种配置方式

log4js.configure({

    appenders: {

        ruleConsole: {type: 'console'},

        ruleFile: {

            type: 'dateFile',

            filename: 'logs/server-',

            pattern: 'yyyy-MM-dd.log',

            maxLogSize: 10 \* 1000 \* 1000,

            numBackups: 3,// 要分割日志文件的最大的文件数

            alwaysIncludePattern: true

        }

    },

    categories: {

        default: {appenders: ['ruleConsole', 'ruleFile'], level: 'info'}

    }

});

默认情况下，log4js-node的每条log输出格式有如下几个字段：日志产生时间，level，日志分类，日志内容。如果用户没有配置，日志分类字段为“default”。