Node.js 第1天

上午总结

- Node.js 是什么
  - JavaScript 运行时
  - 既不是语言，也不是框架，它是一个平台
- Node.js 中的 JavaScript
  - 没有 BOM、DOM
  - EcmaScript 基本的 JavaScript 语言部分
  - 在 Node 中为 JavaScript 提供了一些服务器级别的 API
    - 文件操作的能力
    - http 服务的能力

总结

- Node 中的 JavaScript
  - EcmaScript
    - 变量
    - 方法
    - 数据类型
    - 内置对象
    - Array
    - Object
    - Date
    - Math
  - 模块系统
    - 在 Node 中没有全局作用域的概念
    - 在 Node 中，只能通过 require 方法来加载执行多个 JavaScript 脚本文件
    - require 加载只能是执行其中的代码，文件与文件之间由于是模块作用域，所以不会有污染的问题
      - 模块完全是封闭的
      - 外部无法访问内部
      - 内部也无法访问外部
    - 模块作用域固然带来了一些好处，可以加载执行多个文件，可以完全避免变量命名冲突污染的问题
    - 但是某些情况下，模块与模块是需要进行通信的
    - 在每个模块中，都提供了一个对象：`exports`
    - 该对象默认是一个空对象
    - 你要做的就是把需要被外部访问使用的成员手动的挂载到 `exports` 接口对象中
    - 然后谁来 `require` 这个模块，谁就可以得到模块内部的 `exports` 接口对象
    - 还有其它的一些规则，具体后面讲，以及如何在项目中去使用这种编程方式，会通过后面的案例来处理
  - 核心模块
    - 核心模块是由 Node 提供的一个个的具名的模块，它们都有自己特殊的名称标识，例如

        - fs 文件操作模块
        - http 网络服务构建模块
        - os 操作系统信息模块
        - path 路径处理模块
        - 。。。。
    - 所有核心模块在使用的时候都必须手动的先使用 `require` 方法来加载，然后才可以使用，例如：
        - `var fs = require('fs')`

- http
    - require
    - 端口号
        - ip 地址定位计算机
        - 端口号定位具体的应用程序
    - Content-Type
        - 服务器最好把每次响应的数据是什么内容类型都告诉客户端，而且要正确的告诉
        - 不同的资源对应的 Content-Type 是不一样，具体参照：http://tool.oschina.net/commons
        - 对于文本类型的数据，最好都加上编码，目的是为了防止中文解析乱码问题
    - 通过网络发送文件
        - 发送的并不是文件，本质上来讲发送是文件的内容
        - 当浏览器收到服务器响应内容之后，就会根据你的 Content-Type 进行对应的解析处理

- 模块系统

- Node 中的其它的核心模块

- 做一个小管理系统：
    - CRUD

- Express Web 开发框架
    - `npm install express`