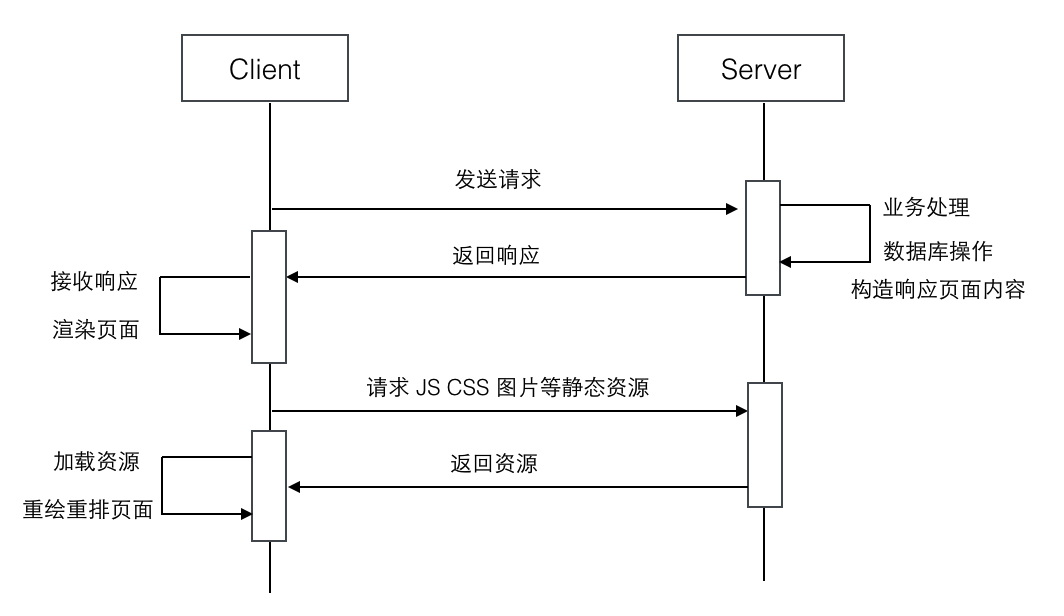
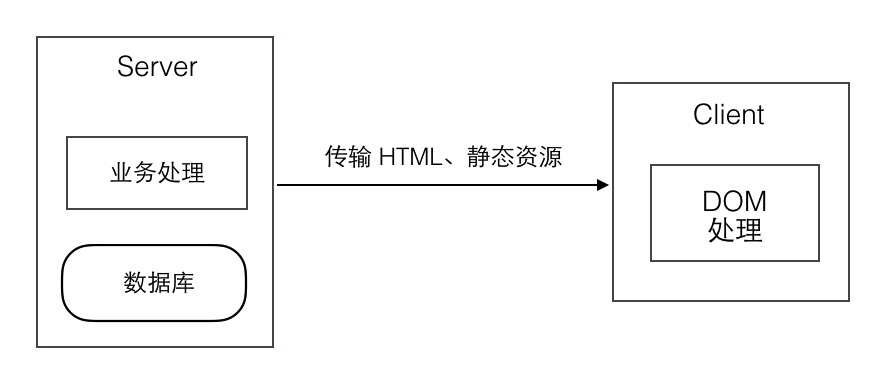
1 　Meteor 的工作原理

1.1 　工作流程

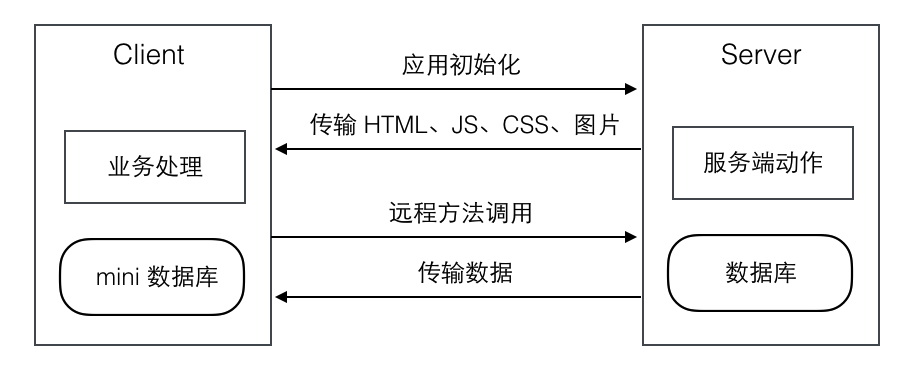
Meteor 在工作方式上进行了较大创新，和传统Web 应用区别较大。下面先回顾一下传统应用的工作流程，如图所示。



客户端（Client）负责向服务器请求所需的数据、资源，然后渲染显示；服务器端（Server）负责业务处理、数据库操作、构造响应内容、资源管理，服务器端的责任大、任务重。其各自职责关系如图所示。



Meteor的工作方式更像是手机APP。客户端首次访问 Meteor应用时，会从服务器把需要用到的资源都加载到客户端，如 JS、CSS、字体、图片，并创建一个mini数据库。然后和服务器端建立好数据通信的通道。之后，用户操作应用过程中涉及业务操作时，也是在客户端进行处理；进行数据库操作时，也是操作客户端的mini 数据库。服务器端只负责向客户端传输数据、数据的安全写入，以及执行一些只能在服务器端进行的操作，例如发送email，如图所示。



Meteor 应用的客户端包含了应用所需的静态资源、业务处理代码、一个简化的数据库。如手机APP 一样，很多操作直接在本地完成，需要执行特定动作和需要数据时才请求服务器端。

所以相比较于传统Web 应用，Meteor 选择了重客户端、轻服务器端的模式，充分利用现代客户端强大的运算能力，减轻服务器端的压力。

1.2 　核心技术

Meteor 的工作方式必然需要一些特定的技术来支持，让我们来了解一下Meteor 的几个核心技术。

1. mini 数据库（mini-database）

Meteor 的底层技术中首先吸引我的就是客户端的 mini 数据库。Meteor 目前支持的数据库是 MongoDB，所以客户端的mini 数据库就是 miniMongo。

对于开发人员来讲，miniMongo 就像是一个真实 MongoDB 数据库，可以进行各种增删改查的操作，和MongoDB 的 API 完全一致。

miniMongo 的主要作用是缓存数据，相当于服务器端数据库的局部镜像，它不会缓存全部数据，只是缓存当前客户端用到的数据。

使用 miniMongo 的效果就是应用运行非常快，而且提供了更好的用户体验。例如用户保存了一条数据，Meteor会先保存到 miniMongo，保存成功后立即反馈给用户，体验极其顺畅；同时 Meteor会把数据同步到服务器端的真实数据库中，这个过程对于用户和开发者都是透明的。

那么如果网络出现问题，或者后台数据库操作时出现问题时，数据没有同步成功怎么办？

当客户端发现没有同步成功后，会通知用户出现了问题，页面执行相应的错误处理逻辑。例如用户保存了一条数据，数据先被写入 miniMongo，然后反馈用户操作成功，同时后台进行数据库同步。万一服务器端操作失败，会通知客户端，客户端会告知用户之前的操作有问题，并执行相应的错误处理流程。

2. Tracker

Tracker提供了响应式应用的基础功能。下面先简单了解一下什么是响应式。以之前创建的项目为例，页面中有一个按钮，单击按钮后，页面中显示的那一个数字自动加1。通过查看代码，代码的逻辑如图所示。

{{ counter }} 通过函数关联了 val 变量，按钮单击事件的处理函数中修改了变量 val 的值，并没有更新页面中的内容，但{{ counter }} 自动更新了，这就是响应式。

响应式的背后技术基础就是 Tracker。Tracker会跟踪目标数据，当其有任何变化后，都会重新计算使用到目标数据的地方。

在上面的示例中，变量 val 是一个响应式变量，会被 Tracker 跟踪，{{ counter }} 是变量 val 的消费者，当 val 被修改后，Tracker 便通知它的消费者进行更新。

3. DDP

DDP是一个数据传输协议。Web应用通常会使用HTTP，为什么还要使用 DDP呢？因为 HTTP 适合传输document，而 Meteor 中主要是传输数据，HTTP 在这方面就不太适合了，所以需要使用专门用于传输数据的 DDP。

DDP 基于 websockets，实现了全双工的数据传输，这一点也优于HTTP。如果使用 HTTP，则只能是客户端请求服务器获取数据，服务器端无法主动向客户端发送数据，而 DDP 的双向机制使数据传输更加主动、灵活。

DDP 使用 JSON 格式封装数据。因为 MongoDB 存储的文档结构是 JSON，客户端的JS 对JSON 的处理也是非常方便的，所以 DDP 协议使客户端和服务器端的数据沟通变得极其自然。

DDP 协议也是响应式功能的基础。因为通过 DDP，服务器端可以主动向客户端发送数据，所以当数据库中有任何变化时，都可以立即通知客户端，客户端便可以进行更新操作，以快速响应。

2 　优势与不足

对于任何一项技术，都有其擅长的领域，也有其不擅长的地方。下面就看一下Meteor 的优势和劣势。

2.1 　优势

Meteor 作为一站式的全栈开发平台，使用一种开发语言就可以贯穿前后端的开发，具有方便的数据交换协议、繁荣的生态等特质，使Meteor 自然地具备了很多优势，如下所示。

易学

使用Meteor 可以快速看到效果，这对学习者来说是一个很大的激励。

Meteor 提供了一套通用JavaScript API，开发者无须深入研究某个特别的前端库，或者某个后端框架，了解基础的JavaScript 就足以起步了。

偏向客户端

现在的应用都非常注重用户端的体验，为了提升客户端的智能效果，就需要客户端与服务器能够双向沟通，需要服务器可以推送数据给客户端。

Meteor 使用的DDP 协议就可以自动实现全双工通信，开发者无须为此费心。

响应式

在目前很多应用的开发中，处理事件（用户单击了某些元素后触发某动作， 如更新数据库，或者更新当前视图）的代码是一个重要部分。

在响应式编程中，这类事件处理函数的工作就减少了。

响应式是Meteor 的主要特征，所以Meteor 非常适合如实时聊天或者在线游戏类的应用。

代码高度重用

与Java 一样：写一次，到处运行。

基于Meteor 的同构特性，相同的代码可以运行于客户端，也可以运行在服务器端，运行在手机移动端也没问题。

强大的 CLI 命令行工具

Meteor 提供了一个命令控制台工具，用来辅助整个开发过程（具体功能上面有描述）。

健康的生态系统

Meteor还是一个生态系统，拥有大量的扩展包，提供了非常丰富的功能。这减少了开发者的很多工作量，而且众多开发者还在不断分享更多的扩展包。

2.2 　弱势

虽然使用 Meteor可以开发很多类型的应用，但在有些情况下，还是建议选择其他的开发平台。毕竟 Meteor不是全能的，有其自身的弱项，在以下一些方面存在不足。

运算密集型应用

Meteor是基于Node.js的，Node.js本质上是单线程处理模式，不能很好地利用多处理器，所以 Meteor不能提供很强的计算能力。

成熟度

Meteor毕竟还很年轻，在大型应用方面还没有成熟的案例，Meteor在大型部署和处理高请求压力方面还需证明自己。

在社区方面，尽管Node.js的社区已经非常成熟，对大家帮助很大，但它还是没法和老牌语言的社区相比，如PHP、Java。

在主机环境方面，支持Meteor的主机仍大大少于支持PHP、Python等语言的主机。

约束少

在Meteor中，对于项目的结构方面没有严格的规定。其好处是很自由，但同时也是缺点。在一个人开发时，没有约束意味着开发速度快；但是在团队中，还是有清晰、固定的结构比较好，便于协作开发。

SQL

如果你的项目一定要使用SQL数据库，那么目前Meteor还无法满足此需求。

现在Meteor官方支持的数据库只有MongoDB。虽然已经计划支持SQL数据库，还有社区成员修改为支持SQL的成功案例，但毕竟尚未成熟支持。

静态化内容

类似新闻类型的网站，很多内容都已经生成为静态化的文件。客户端发送请求给服务器，服务器返回静态化HTML内容，这个场景更适合使用传统Web 平台—可以充分利用服务器的静态内容缓存—用户请求一个新闻页面，服务器端从缓存获取静态化文件，直接返回给用户，速度非常快。

而使用 Meteor 则利用不到 Meteor 的任何优势。因为Meteor 的优势是响应式和强大的交互通信协议，静态类型的网站自然不需要这些特质。

初次加载时间

如果对于加载时间有较高要求，就不适合使用Meteor。因为Meteor 初次加载慢、后期访问快，初始访问时会相对耗时，需要加载很多静态资源。