УДК 004.4

ОСОБЕННОСТИ WEB-РАЗРАБОТКИ НА ПРИМЕРЕ СЕРВЕРНОЙ ПЛАТФОРМЫ NODEJS

Жуйков Н.А
Научный руководитель – Белова С.В.

Всегда считалось, что JavaScript обычное дополнение к браузеру, которое необходимо только для управления формами и манипулирования DOM-деревом веб-страницы.Такое мнение складывалось из-за того, что JavaScript достаточно простой язык программирования.Но он одновременно гибкий, что позволяет писать код в разных парадигмах: от процедурного до объектно-ориентированного в смеси с функциональным стилем. Главное преимущество этого языка — его асинхронность, которая позволяет выполнять блок кодане последовательно, а именно в тот момент, когда для него будут готовы все данные. Такое поведение достигается наличием функции, которая называется callback, или обработчик событий. И хотя писать действительно сложный код в таком стиле немного неудобно, для этого уже придумали свои фреймворки. Например, NodeJS.

Рисунок 1. Веб-приложение, реализованное на языке JavaScript

NodeJS – серверный фрэймворк для языка JavaScript, который выполняет код на стороне сервера, так называемом backend-е, а не на стороне браузера, как привычный JavaScript. Все механизмы обработки запросов и прочих операций ввода/вывода (I/O) построены на событиях, что означает, что в NodeJS нет практически никакого способа, чтобы

заблокировать работающий в данный момент поток. В отличие от других популярных фрэймворков, NodeJS не используют многопоточную модель, здесь есть только один рабочий поток, который обслуживает все запросы пользователей и отвечающие ресурсы (на рисунке 2 приведена схема). И существует стек асинхронных потоков NodeJS, для операций ввода-вывода.

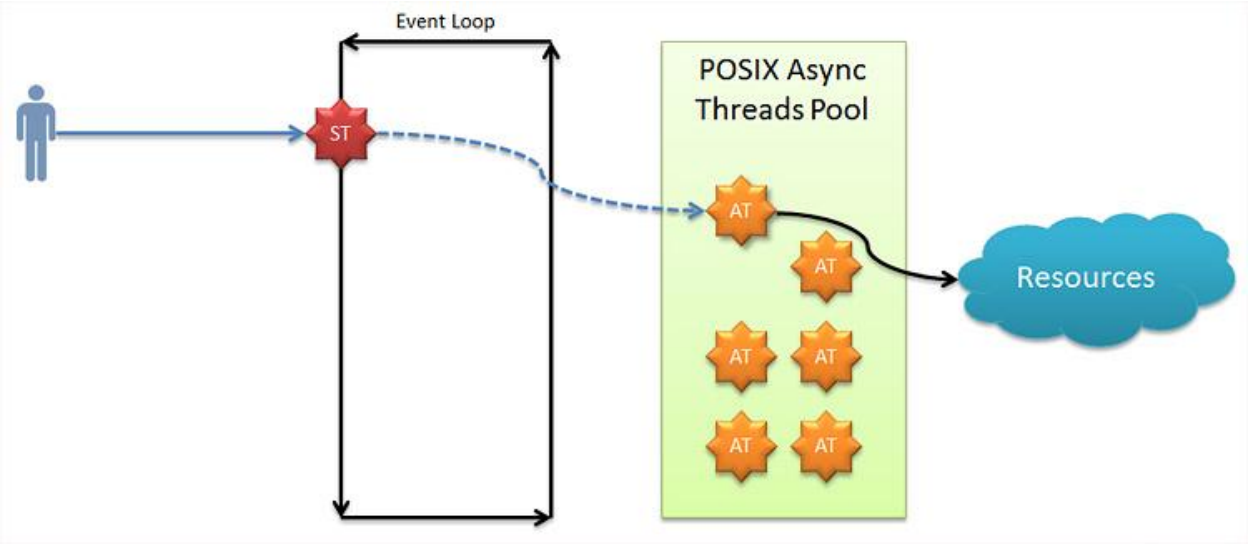

Рисунок 2. Асинхронная модель NodeJS

Производительность такой системы гораздо выше, чем многопоточной модели. На рисунке 3 представлены показатели производительности NodeJS и Apache.

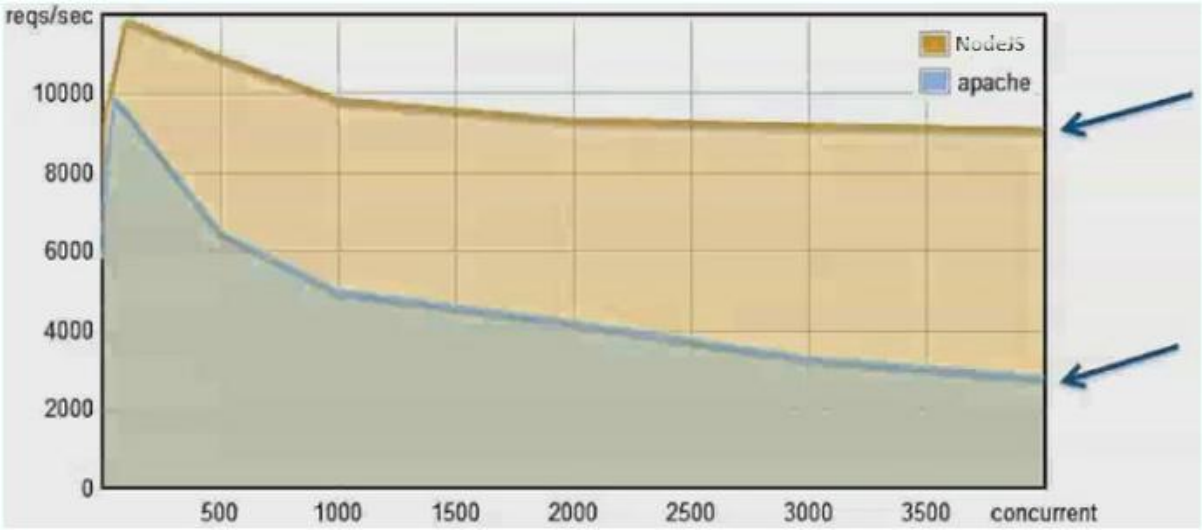

Рисунок 3. Показатели производительности NodeJS и Apache

При работе с асинхронной моделью, предоставляемой фрэймворком NodeJS, необходимо учитывать некоторые ее особенности:

1. Не использовать блокирующие операции, такие как считывание файла в переменную в время её объявления, поскольку как правило подобные действия полностью разрушают структуру асинхронной модели.
2. Не использовать больших циклов для обработки данных. Циклы, особенно те, которые требуют много времени на выполнение, будут останавливать поток программы. Если все же возникает необходимость использования цикла, то нужно использовать события.
3. Не создавать больших методов, занимающих много процессорного времени. Если подобная необходимость есть, то метод должен разбиваться на более мелкие, которые будут вызываться последовательно.

```
schema.statics.allUsersFiles = function(username, callback) {
    var File = this;
    async.waterfall([
        function(callback) {
            File.find({username: username}, callback);
        },
        function(recordsByUser, callback) {
            if (recordsByUser) {
                callback(recordsByUser);
            } else {
                callback("No Text");
            }
        }
    ], callback);
};
```

Рисунок 4. Пример кода функции с применением callback-ов

Таким образом фрэймворк NodeJSпозвоялет разрабатывать достаточно быстрыеи функциональные приложения, а с учетом его применения в веб-разработке, мы получаем возможность писать все приложение целиком(frontи backend) на одном языке – JavaScript. Ниже приведен список крупных проектов, реализованных на NodeJS:

1. Яндекс.Почта
2. GMail
3. Почта mail.ru
4. XMPP-сервер Вконтакте
5. Облачный хостинг Joyent
6. Облачный хостинг Heroku
7. Корпоративная социальная сеть Yammer