NODEJS

Leonardo Soares Paulino

Analise e Desenvolvimento de Sistemas – Escolas e Faculdades QI
Gravataí – RS – Brasil

leonardosoarespaulino@outlook.com.br

Abstract. *This article describes the structure for creating APIs called Nodejs.*

Resumo. *Este artigo descreve sobre o framework para criação de API chamado Nodejs.*

1. NodeJs

Node.js é um interpretador de código JavaScript que funciona do lado do servidor. Seu objetivo é ajudar programadores na criação de aplicações de alta escalabilidade (como um servidor web), com códigos capazes de manipular dezenas de milhares de conexões simultâneas, numa única máquina física. O Node.js é baseado no interpretador V8 JavaScript Engine (interpretador de JavaScript open source implementado pelo Google em C++ e utilizado pelo Chrome). Foi criado por Ryan Dahl em 2009, e seu desenvolvimento é mantido pela empresa Joyent, onde Dahl trabalha.

2. Que Problemas o NodeJS resolve

Node estabeleceu o objetivo número um que é “fornecer uma maneira fácil para construir programas de rede escaláveis”. Qual é o problema com os programas servidores atuais? Vamos fazer os cálculos. Em linguagens como Java™ e PHP, cada conexão cria uma nova thread que potencialmente tem anexado 2 MB de memória com ela. Em um sistema que tenha 8 GB de RAM, isso põe o número máximo teórico de conexões concorrentes a cerca de 4.000 usuários. E quando o número de usuários aumenta, se você quer que sua aplicação web suporte mais usuários, você tem que adicionar mais e mais servidores. Somado a estes custos também podem haver possíveis problemas técnicos: um usuário pode usar diferentes servidores para cada requisição, então cada recurso compartilhado deve ser compartilhado para todos os servidores. Por todas estas rações, o gargalho em toda a arquitetura de aplicações web (incluindo velocidade de tráfego, velocidade do processador e velocidade da memória) é o número de conexões concorrentes que o servidor pode manipular.

Node resolve esta questão trocando a maneira como a conexão é tratada no servidor. Ao invés de criar uma nova OS thread a cada conexão (e alocar a memória anexa a ela), cada conexão dispara um evento executado dentro da engine de processos do Node. Node afirma que nunca vai dar deadlock, já que não há bloqueios permitidos, e ele não bloqueia diretamente para chamadas

de I/O. Node também alega que um servidor rodando ele pode suportar dezenas de milhares de conexões simultâneas.

Então, agora que você tem um programa que pode manipular dezenas de milhares de conexões simultâneas, o que você pode realmente fazer com o Node? Seria incrível se você tivesse uma aplicação web que necessitasse desta quantidade de conexões. Este é um daqueles tipos de problema: “se você tem um problema, não é mais um problema”.

3. Nodejs definitivamente não é?

Sim, Node é um servidor de programas. Entretanto o produto base do Node definitivamente não é como o Apache ou o Tomcat. Estes servidores são basicamente servidores ready-to-install e estão prontos para instalar aplicativos instantâneamente. Você pode subir e rodar um servidor em um minuto com estes produtos.

Node definitivamente não é isso. Parecido com como o Apache pode adicionar um módulo PHP para permitir desenvolvedor criarem páginas da web dinâmicas, e um módulo SSL para conexões seguras, Node tem o conceito de módulos que podem ser adicionados no núcleo do Node.

Há literalmente centenas de módulos para rodarem com o Node, e a comunidade é bastante ativa em produzir, publicar e atualizar dezenas de módulos por dia.

4. Como o NodeJs funciona

O Node roda em uma Java Script V8 VM. Mas espere Java Script no servidor? Isso, você leu certo. Java Script no lado do servidor pode ser um conceito novo para todos que trabalharam exclusivamente com o Java Script no lado do cliente, mas a ideia em si não é tão absurda – porque não usar a mesma linguagem de programação no cliente que você usa no servidor? O que é V8? O motor Java Script V8 é o motor que a Google usa com seu navegador Chrome. Poucas pessoas pensam sobre o que realmente acontece com o JavaScript no lado do cliente. Bem, a engine JavaScript realmente interpreta o código e o executa. Com o V8 a Google criou um ultra-rápido interpretador escrito em C++, com um outro aspecto único: você pode baixar a engine e incorporá-la em qualquer aplicação desejada. Isso não está restrito em rodar em um navegador. Então Node atualmente usa o motor JavaScript V8 escrito pela Google e propõe que seja usado no servidor. Perfeito! Para que criar uma nova linguagem quando há uma boa solução já disponível?

5. Programação orientada a evento

Muitos programadores foram ensinados a acreditar que a programação orientada a objetos é um modelo de programação perfeito e a não usarem nada mais. Node utiliza o que é chamado modelo de programação orientada a evento.

Programação orientada a evento no lado do cliente com jQuery:

// jQuery code on the client-side showing how Event-Driven programming works
// When a button is pressed, an Event occurs - deal with it

// directly right here in an anonymous function, where all the

// necessary variables are present and can be referenced directly
$("#myButton").click(function(){ if ($("#myTextField").val() != $(this).val())
alert("Field must match button text"); });

O lado do servidor na verdade não é diferente do lado do cliente. Claro que não há botões sendo pressionados e não há campos de texto sendo escritos, mas em um nível mais alto, os eventos estão ocorrendo. Uma conexão é feita – evento!

Dado é recebido através da conexão – evento! Data parou de chegar através da conexão – evento! Por que é que este tipo de configuração é ideal para o Node? JavaScript é uma excelente linguagem para programação orientada a evento, porque ela permite funções anônimas e encerramentos, e o mais importante, a sintaxe é familiar para quase todos que já programaram na vida. As funções de callback que são chamadas quando um evento ocorre podem ser escritas no mesmo lugar onde você captura o evento.

Fácil para desenvolver, fácil para manter. Sem frameworks complicados de Orientação a Objeto, sem interfaces, nenhum potencial para o excesso de arquitetura de qualquer coisa. Basta escutar um evento, escrever uma função de callback, e o Node toma conta de tudo.

6.Considerações Finais

Considerando todas as informações buscadas na internet, considero que Nodejs é um excelente framework recém lançado com muito potencial futuro.

7. References

O que é NodejS, http://nodebr.com/o-que-e-node-js//, outubro.

.