*Node.JS Parte I - A*

**. Servidor x Cliente**

Apenas o cliente chama o servidor, que devolve ao cliente determinada informação, ou seja, o cliente faz requisições para o servidor que devolve com as respostas.

**Servidor:** Somente responde requisições, armazena dados, tem grande poder de processamento, está sob a governança da empresa, atende a vários clientes e tem grande impacto de segurança

**Cliente:** Envia requisições, normalmente não armazena dados, tem baixo poder de processamento, está sob a guarda do cliente, atende a somente um cliente, menor impacto de segurança

**. Server-side x Client-side**

**Server-side:** Tudo que roda no servidor como: Banco de dados, Programas em Node.js, Fotos e Emails

**Client-side:** Tudo que roda no dispositivo do cliente como: Navegadores, Aplicativos, Programas instalados

**. Aplicações server-side:** Muito comuns nos anos 90-2000, hoje em dia não são tão comuns**.** São aplicações em que todas as regras de negócio, lógica de programação e tratamento de dados estão do lado do servidor. O servidor também tem papel de converter o resultado do processamento em hipertexto (famoso HTML) e a aplicação cliente tem somente o papel de exibir o resultado

**. Evolução do client-side:** Os fatores que contribuíram para essa evolução foram o surgimento de computadores pessoais mais potentes, smartphones e a própria evolução dos navegadores. O cliente-side começou a dividir as responsabilidades com o servidor, o mesmo agora é: Responsável pela geração do HTML, pela lógica de usabilidade, pela lógica de programação mais “leve” e com os frameworks modernos tornou-se mais generalista e reaproveitável.

**OBS:** JSON é uma forma de codificar os dados para que o cliente consiga entender, é o “idioma” que o cliente conversa com o servidor

**. Ambientes cliente-side:** Hoje em dia, existem três ambientes mais comuns no mercado, são eles: Navegador (linguagem JavaScript), Android (linguagem Kotlin) e iOS (linguagem Swift)

**. Arquitetura client-server:** Cliente -> Notebook ou Smarthpone (JavaScript) -> Servidor (API e Banco de Dados)

**. Backend x Frontend**

**. Frontend (client-side):** É a parte do notebook ou smarthphone que tem lá como a linguagem o JavaScript

**. Backend (server-side):** É a parte do servidor que tem a API (node) e o Banco de dados

**. Por que o Node:** O Node tornou possível a utilização de uma única linguagem (JavaScript) para desenvolvimento do backend e do frontend

**. Node:** Não é uma linguagem, é um framework.É um ambiente em tempo de execução open-source (código aberto) e multiplataforma que permite aos desenvolvedores criarem todo tipo de aplicativos e ferramentas do lado servidor (backend) em JavaScript. Node é usado fora do contexto de um navegador (ou seja, executado diretamente no computador ou no servidor). Como tal, o ambiente omite APIs JavaScript específicas do navegador e adiciona suporte para APIs de sistema operacional mais tradicionais, incluindo bibliotecas de sistemas HTTP e arquivos.

**. História do Node:** Surgiu em 2009 pelo Ryan Dahl devido a insatisfação com o Apache. É escrito em C, C++ e JavaScript e suporta linguagens como JavaScript e TypeScript

**. Documentações de apoio:** nodejs.org; devdocs.io/node; stackoverflow.com

**. Para que serve um pacote:** Permite compartilhar funcionalidades implementadas entre diferentes projetos, pessoas e empresas

**. NPM:** É o gerenciador de pacotes do Node permitindo a busca, instalação, gerenciamento, criação e publicação de pacotes

**. História do NPM:** Surgiu em 2010 pelo Isaac Z. Schlueter a fim de gerenciar pacotes no NodeJS que antes era terrível. É escrito em JavaScript.

**. Componentes do NPM**

**Website:** Busca e análise de pacotes, onde fica toda a documentação de apoio dos pacotes (www.npmjs.com)

**CLI:** Comandos de terminal, mais utilizado. Instalar pacotes, desinstalar pacotes, atualizar pacotes (https://docs.npmjs.com/cli/v7/commands)

**Registry:** Local onde ficam armazenados os pacotes. Armazenar todos os pacotes disponíveis no NPM, acesso através do NPM CLI (npm install) (registry.npmjs.org)

**. Documentações de apoio:** npmjs.com; devdocs.io/npm; stackoverflow.com

**. JavaScript:** É uma linguagem de programação que permite o desenvolvimento de programas frontend, ela é carregada quando uma página web é acessada e o navegador é o responsável pela sua execução. Seu objetivo inicial era dar “vida” as páginas web, tornar o frontend mais “inteligente” e diminuir as interações com o backend/servidor

**. História do JavaScript:** Criado em 1995 pelo Netscape, no começo chamava LiveScript. Biblioteca jQuery foi uma das mais utilizados com o JavaScript

**. Documentações de apoio:** developer.mozilla.org/javascript; [www.w3schools.com/nodejs](http://www.w3schools.com/nodejs); stackoverflow.com

**. TypeScript:** É uma linguagem de programação moderna para desenvolvimento JavaScript. É uma linguagem compilada para JavaScript, assim o programa construído pode rodar em qualquer lugar onde o JavaScript roda. Ele oferece funcionalidades como tipagem estática, classes e interfaces. É um superset do JavaScript, precisa ser compilado e é mais recomendado para grandes projetos.

**. História do TypeScript:** Foi lançado em 2013 pela Microsoft onde o seu diferencial era trazer elementos de linguagens mais estruturadas para o JavaScript

**. Documentações de apoio:** [www.typescriptlang.org](http://www.typescriptlang.org); stackoverflow.com