

Javascript x Typescript.

Introdução a APIs RESTful

Javascript ou Typescript?

Podemos resumir JS a uma linguagem que foi criada para usos rápidos e então cresceu para uma ferramenta com implementação de servidores, feito para escrever aplicações com milhões de linhas.

coerção dos argumentos

```
if ( '' == 0 ) {  
    // É! mas porque??  
}  
if ( 1 < x < 3 ) {  
    // Verdadeiro para qualquer valor de x!  
}
```

acessar propriedades que não existem...

```
const obj = { width: 10, height: 15 };  
// Porque isso é NaN? Ortografia é difícil!  
const area = obj.width * obj.heighth;
```

Javascript ou Typescript?

“TypeScript mantém uma relação incomum com o [JavaScript](#), ou seja, oferece todos os recursos do JavaScript e uma camada adicional sobre eles: o sistema de tipos TypeScript.”



```
function aumento(salario, aumento){  
  return salario + aumento  
}  
aumento(10, "30")
```

```
const produto = {  
  id: 1,  
  nome: "Copo",  
};
```

```
function aumento( salario: number, aumento: number){  
  return salario + aumento  
}  
aumento(100, "30")
```

Argument of type 'string' is not assignable to parameter of type 'number'. ts(2345)

[View Problem \(Alt+F8\)](#) No quick fixes available

```
interface Produto {  
  id: number;  
  name: string;  
}
```

Feedback mais eficiente de erros

Processo de refatoração mais ágil

autocomplete de código

Qual a principal diferença entre Javascript e Typescript?

Javascript:

1. Tipagem dinâmica
2. Interpretação e execução imediata
3. Suporte amplo
4. Ecossistema rico

Typescript:

1. Tipagem estática
2. Compilação
3. Suporte a funcionalidades modernas de Javascript
4. Ferramentas melhoradas
5. Classes e interfaces

Typescript: um verificador de tipos estáticos

Detecção de erros sem execução do código é chamado de verificação estática.

Verificação estática de tipos:

```
const obj = { width: 10, height: 15 };  
const area = obj.width * obj.heighth;
```

Property 'heighth' does not exist on type '{ width: number; height: number; }'.
Did you mean 'height'?

Como Typescript se relaciona com Javascript então?

- Sintaxe

```
let a = (4
```

)' expected.

- Tipos

```
console.log(4 / []);
```

js

```
console.log(4 / []);
```

The right-hand side of an arithmetic operation must be of type 'any', 'number', 'bigint' or an enum type.

ts

Javascript ou Typescript?

- Comportamento em tempo de execução

Como um princípio, o TypeScript **nunca** muda o comportamento de tempo de execução de código JavaScript. Isso significa que você mover código do JavaScript do TypeScript, é **garantido** que vai rodar da mesma forma mesmo que o TypeScript pensa que o código tem erros de tipos.

- Tipos apagados

Isso significa que o TypeScript nunca muda o *comportamento* do seu programa baseado nos tipos que infere. O fim disso é que enquanto você vê erros de tipo durante a compilação, o sistema de tipos em si não tem influência sobre como seu programa funciona quando roda. Finalmente, TypeScript não fornece nenhuma biblioteca de tempo de execução adicional. Seus programas vão usar as mesmas bibliotecas padrão (ou externas) que os programas JavaScript, então não há ferramentas específicas do TypeScript adicionais para se aprender.

Eu deveria aprender Javascript ou Typescript?

Não tem como aprender Typescript sem aprender Javascript! TS compartilha sintaxe e comportamento de tempo de execução com Javascript.

Todas as questões de JS se aplicam a TS.

Se você se encontra procurando por algo como “como organizar uma lista em TypeScript”, lembre-se: **TypeScript é o ambiente de execução do JavaScript com verificador de tipo em tempo de compilação**. A forma com que você organiza uma lista em TypeScript é a mesma no JavaScript.

Introdução à APIs RESTful

- **Application Programming Interface (API)** - conjunto de regras e protocolos que permite que um software interaja com outro.
- **Interface** pode ser pensada como um contrato de serviço entre duas aplicações através de solicitações e respostas.
- **Documentação** contém informações sobre como os desenvolvedores devem estruturar suas solicitações e respostas.

Como as APIs funcionam?

Arquitetura cliente-servidor

APIs SOAP (Simple Object Access Protocol) - XML

APIs RPC (Remote Procedure Calls) - conclui uma função no servidor e obtém a resposta.

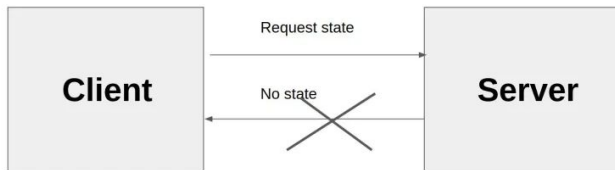
APIs WebSocket - oferece suporte à comunicação bidirecional entre aplicativos cliente e o servidor.

APIs REST - cliente envia solicitações ao servidor como dados. O servidor inicia funções internas e retorna os dados de saída ao cliente.

O que são APIs REST?

- REST - Transferência Representacional de Estado: a implementação do cliente e do servidor pode ser feita de forma independente, sem que cada um conheça o outro. Isso significa que o código do lado do cliente pode ser alterado a qualquer momento, sem afetar a operação do servidor, e o contrário também é válido.
- REST define um conjunto de funções como GET, POST, PUT, DELETE que trocam dados através de requisições HTTP.
- Stateless: cada requisição do cliente ao servidor é independente e contém todas as informações necessárias para que o servidor processe a informação, isso significa que **o servidor não precisa saber nada sobre o estado em que o cliente se encontra e vice-versa;**

What is Stateless?



O que são APIs REST?

- Cacheable: As restrições de cache requerem que as informações contidas em uma resposta a uma solicitação sejam definidas como cacheáveis ou não cacheáveis. Caso uma resposta for armazenável em cache, então, é dado ao cliente o direito de utilizar novamente esses dados para atividades similares no futuro.
- Uniform interface: O foco é uma interface uniforme entre os componentes. O princípio de generalidade, característica da engenharia de software, quando aplicado à interface de componentes, simplifica a arquitetura do sistema e torna as interações mais visíveis.

identificação de recursos

gerenciamento de recursos por meio de representações

mensagens auto-descritivas

hipermídia

- Sistema em camadas: **Acréscimo de restrições de sistema por camadas.** Assim, a arquitetura é constituída por camadas que seguem uma ordem hierárquica. Isso restringe o comportamento dos elementos, de modo que cada um deles só possa “enxergar” a camada com que estão interagindo no momento.

Métodos HTTP e seus usos dentro de uma API

São utilizados para que consigamos criar o **CRUD** do nosso sistema, é preciso criar, ler, atualizar e deletar uma entidade do nosso sistema.

GET: solicita a representação de recurso do nosso sistema que foi modelado no nosso banco de dados.

POST: é usado para enviar dados para o servidor para criar um novo recurso no nosso bd.

PUT: substitui todas as representações atuais do recurso de destino pelo payload da requisição.

PATCH: é usado para aplicar modificações parciais a um recurso do nosso bd.

DELETE: é usado para remover um recurso do nosso bd

Métodos HTTP e seus usos dentro de uma API

Query strings

Parte de uma URL que contém dados que são enviados para o servidor como parte de uma requisição

`https://api.example.com/users?id=12345&name=John%20Doe`

- Filtrar dados
- Ordenar
- Fornecer informações adicionais para a req

Quais são os benefícios das APIs REST?

Integração: APIs são usadas para integrar novas aplicações com sistemas de software existentes.

Inovação: Setores inteiros podem mudar com a chegada de uma nova aplicação. As empresas precisam responder rapidamente e oferecer suporte à rápida implantação de serviços inovadores. Elas podem fazer isso fazendo alterações no nível da API sem precisar reescrever todo o código.

Expansão: Qualquer empresa pode fornecer acesso semelhante aos seus respectivos bancos de dados internos usando APIs gratuitas ou pagas.

Facilidade de manutenção: A API atua como um gateway entre dois sistemas. Cada sistema é obrigado a fazer alterações internas para que a API não seja afetada. Dessa forma, qualquer alteração futura de código feita por uma parte não afetará a outra parte.

O que é um endpoint de API e porque é importante?

É uma URL específica que responde a uma solicitação, representando um ponto de acesso onde um recurso é explorado através do método HTTP (GET, POST, PUT, PATCH ou DELETE).

Elementos

- URL
- Métodos HTTP
- Parâmetros
- Resposta

Importância

- Organização
- Interoperabilidade
- Escalabilidade
- **Segurança**
- Documentação

GET <https://api.example.com/users/12345>

Como proteger uma API REST?

- Tokens de autenticação

Usados para autorizar os usuários a fazer a chamada de API. Os tokens de autenticação verificam se os usuários são quem afirmam ser e se têm direitos de acesso para aquela chamada de API específica. Por exemplo, quando você faz login em seu servidor de e-mail, seu cliente de e-mail usa tokens de autenticação para acesso seguro.

```
{  
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",  
  "token_type": "Bearer",  
  "expires_in": 3600  
}
```

JSON Web Token (JWT) é um padrão aberto (RFC 7519) que define uma maneira compacta e autocontida de transmitir informações entre partes como um objeto JSON. Essas informações podem ser verificadas e confiáveis porque são assinadas digitalmente

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTU2MjM0MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36P0k6yJV_adQssw5c
```

Boas práticas em APIs REST

- Utilizar nomes coerentes e claros para os recursos nas URLs
- Retornar os códigos de status HTTP corretos
 - ex: 200, 404, 500 e etc
- Fazer validações completas dos dados recebidos
- Autenticar e autorizar o acesso aos recursos corretamente;
- Permitir a paginação dos resultados com limit e offset;
- Documentar corretamente os recursos e respostas;
- Versionar a API para evitar problemas com atualizações;
- Não retornar dados sensíveis desnecessariamente;

Referências

- <https://www.typescriptlang.org/pt/docs/handbook/typescript-from-scratch.html>
- Gao, Zheng, Christian Bird, and Earl T. Barr. "To type or not to type: quantifying detectable bugs in JavaScript." *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*. IEEE, 2017.
- https://link.springer.com/chapter/10.1007/978-3-662-44202-9_11
- <https://www.escoladnc.com.br/blog/o-que-e-protocolo-http-e-metodos-de-requisicao-em-apis/>
- <https://aws.amazon.com/pt/what-is/api/>
- https://en.wikipedia.org/wiki/Query_string
- Jansen, Remo H., Vilic Vane, and Ivo Gabe De Wolff. *TypeScript: Modern JavaScript Development*. Packt Publishing Ltd, 2016.
- <https://rockcontent.com/br/blog/rest/>