06

Método que devolve uma promise

Vejamos nossa classe `NegociacaoDao`. O método `adiciona` que devolve uma *promise*:

```
class NegociacaoDao {

    constructor(connection) {

        this._connection = connection;
        this._store = 'negociacoes';
    }

    adiciona(negociacao) {

        return new Promise((resolve, reject) => {

            let request = this
                ._connection
                .transaction([this._store],"readwrite")
                .objectStore(this._store)
                .add(negociacao);

        });
    }

   listaTodos() {
        // ainda não implementado
  }

}
```

Veja que o método está incompleto, porque em nenhum momento chama a função `resolve` ou `reject`, fundamentais para que a *promise* retorne seu valor ou uma exceção.

Qual das opções abaixo completa o método adiciona?

Selecione uma alternativa

A

```
adiciona(negociacao) {

    return new Promise((resolve, reject) => {

        let request = this
            ._connection
            .transaction([this._store],"readwrite")
            .objectStore(this._store)
            .add(negociacao)
            .onsuccess = e => resolve();
            .onerror = e => reject(e.target.error.name);

    });
}
```

B

```
adiciona(negociacao) {

    return new Promise((resolve, reject) => {

        let request = this
            ._connection
            .transaction([this._store],"readwrite")
            .objectStore(this._store)
            .add(negociacao);

        request.onsuccess = e => resolve();
        request.onerror = e => reject(e.target.error.name);

    });
}
```

C

```
adiciona(negociacao) {

    return new Promise((resolve, reject) => {

        let request = this
            ._connection
            .transaction([this._store],"readwrite")
            .objectStore(this._store)
            .add(negociacao);

        request.onsuccess = e => reject();
        request.onerror = e => resolve(e.target.error.name);

    });
}
```