Entrando no território dos mocks:

Antes de falar específicamente sobre **mocks**, vamos **melhorar nosso sistema**, agora precisamos adicionar uma camada de **persistência do nosso método de locação**(no caso, onde ele será salvo, não importa se vai ser banco, arquivo ou qualquer outro lugar). É aqui que devemos adicionar algum serviço para persistir a locação:

```
        //Entrega no dia seguinte
        Date dataEntrega = new Date();
        dataEntrega = adicionarDias(dataEntrega, 1);
        if(DataUtils.verificarDiaSemana(dataEntrega, Calendar.SATURDAY)) {
            dataEntrega = adicionarDias(dataEntrega, 1);
        }
        locacao.setDataRetorno(dataEntrega);

        //Salvando a locacao...
        //TODO adicionar método para salvar

        return locacao;
    }
```

Normalmente é aplicado o padrao **DAO(Data access object)** nesse ponto, então a forma de persistência desses dados ficam disponíveis para a camada de serviço.

DAO: DAO é um padrão de acesso a dados com base em objetos. Basicamente ele abstrai como os dados serão persistidos e recuperados. *(explicação chula, deverá ser revisado assim que possível).*

Por isso vamos **usar uma interface aqui**, não importa como a aplicação persistirá os dados de locação, a responsabilidade da implementação é dela, a gente **só assina o contrato que prove o método para nós**.

Então criamos uma interface DAO da locação em um package de dao, e nela só temos um método abstrato de salvar que recebe uma locação:

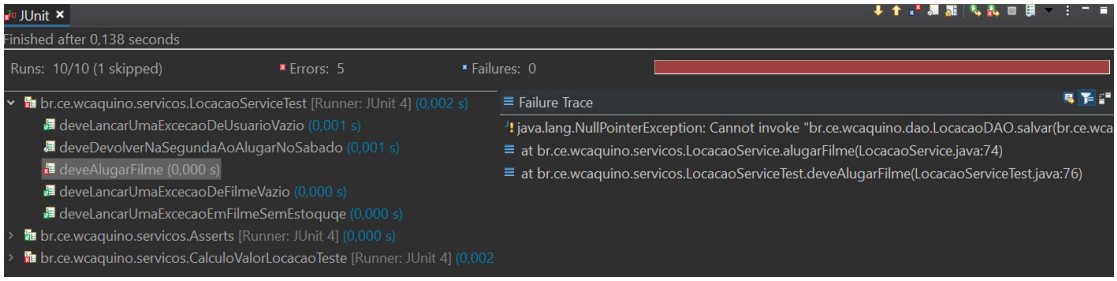
```
package br.ce.wcaquino.dao;

import br.ce.wcaquino.entidades.Locacao;

public interface LocacaoDAO {

    public void salvar(Locacao locacao);

}
```

Agora a gente declara a **LocacaoDAO como variável de instância da LocacaoService** e a utiliza no método de alugar chamando salvar e passando a locação:

```
public class LocacaoService {

    private LocacaoDAO dao;

        //Salvando a locacao...
        dao.salvar(locacao);

        return locacao;
```

Executando os testes:

Agora se rodarmos os testes depois dessa mudança, sem novidade, alguns deles falham. Isso ocorre porque agora passamor a ter a dependência do DAO no método:



Então temos que implementar o DAO para passar nos teses? **NÃO!!!**

Se fizessimos isso, eles seriam **agora testes de integração** e não mais testes **unitários**, esse problema será resolvido com Mock, o que será discutido nas próximas versões do projeto.