Inversão de Controle

IOC (Inversão de Controle)

Ronaldo Lanhellas

IOC

- Normalmente, quando precisamos acessar um recurso presente em outra classe, criamos uma nova instância desta classe e, assim, podemos acessar seus parâmetros e métodos de forma segura, já que aquela instância foi criada exclusivamente para o uso no nosso escopo atual.

IOC

```
class MyPresenter {
    private UserRepository repository;

    MyPresenter() {
        this.repository = new UserRepository();
    }
}
```

IOC

- Quando usamos o repository = new UserRepository() estamos gerando um acoplamento entre a nossa classe MyPresenter e a classe UserRepository pois a primeira é quem fica encarregada de inicializar a segunda e isso gera uma dependência entre ambas.

IOC

- Inicialmente, isso não parece ser um problema, já que fomos ensinados a fazer dessa forma. O problema começa a surgir quando precisamos isolar as camadas da nossa aplicação

IOC

- Para evitar este problema fazemos com que uma classe nunca crie instâncias de outras classes pertencentes a outras camadas de sua aplicação. Para conseguirmos atingir este objetivo utilizamos a Injeção de Dependências.

Injeção de Depêndencia (DI)

- A Injeção de Dependências é a forma que usamos para atingir a inversão de controle. Existem diversas estratégias de implementação mas nos atentaremos às 3 mais conhecidas e utilizadas.

Injeção por Construtor

```
class MyPresenter {
    private UserRepository repository;

    MyPresenter(UserRepository repository) {
        this.repository = repository;
    }
}
```

Injeção por Setter

```
class MyActivity {

    private MyPresenter presenter;

    public void onCreate() {
        ...
        presenter = new MyPresenter();
        presenter.setRepository(new UserRepository());
    }

}

class MyPresenter {
    private UserRepository repository;

    public void setRepository(UserRepository repository) {
        this.repository = repository;
    }
}
```

Injeção por Interface

```
interface RepositorySetter {
    public void setRepository();
}

class MyPresenter implements RepositorySetter {
    private UserRepository repository;

    @Override
    public void setRepository(UserRepository repository) {
        this.repository = repository;
    }
}
```