

## Para saber mais - Criação do Database via Singleton

A abordagem de centralizar a instância do Database é muito comum na comunidade de devs Android. Inclusive, a [própria documentação faz uma nota](https://developer.android.com/training/data-storage/room/) (no final da página) sugerindo o uso do [padrão Singleton](https://en.wikipedia.org/wiki/Singleton_pattern) em situações que o nosso App mantém apenas um único processo (a situação comum no geral).

Considerando o mesmo código, utilizando o Singleton teríamos o seguinte resultado:

```
public static AgendaDatabase getInstance(Context context) {  
    if (instance == null) {  
        instance = Room.databaseBuilder(context, AgendaDatabase.class, NOME_BANCO_DE_DADOS)  
            .allowMainThreadQueries()  
            .build();  
    }  
    return instance;  
}
```

Note que o objetivo é fornecer apenas uma única instância. A princípio, essa abordagem é sugerida pelo fato de que apenas uma única instância é necessária e a criação do banco de dados pode ser custosa.

É importante notar também que existe outros cuidados para que a solução mantenha a proposta de única inicialização, como é o caso do acesso síncrono por meio do [synchronized](https://docs.oracle.com/javase/tutorial/essential/concurrency/syncmeth.html) (<https://docs.oracle.com/javase/tutorial/essential/concurrency/syncmeth.html>) que evita o problema de concorrência em situações que duas threads acessam ao mesmo tempo e gera instâncias diferentes.