

# Projeto Final de Programação Modular

## ÉFIFO TaskManager

Bruno Vinicius, Diego Aniceto, Édipo Fernandes, Lucas Barsand  
{bruno.avila, dhca, edipofvo, lucas.barsand}@dcc.ufmg.br

### Resumo

Este Relatório descreve o processo de desenvolvimento da aplicação **ÉFIFO TaskManager** gerenciador de tarefas. Para que a implementação deste fosse possível foram utilizados diversos conhecimentos de orientação a objeto, Interface Gráfica, Banco de Dados, entre outros.

### Introdução

Atualmente, as pessoas sentem muito a necessidade de organizar as suas tarefas, e isto é muito bem exemplificado com a alta popularidade de aplicativos de tarefas para smartphones e em e-mails, como o Google Tasks localizado dentro do GMail e Tarefas do calendário do Outlook.com.

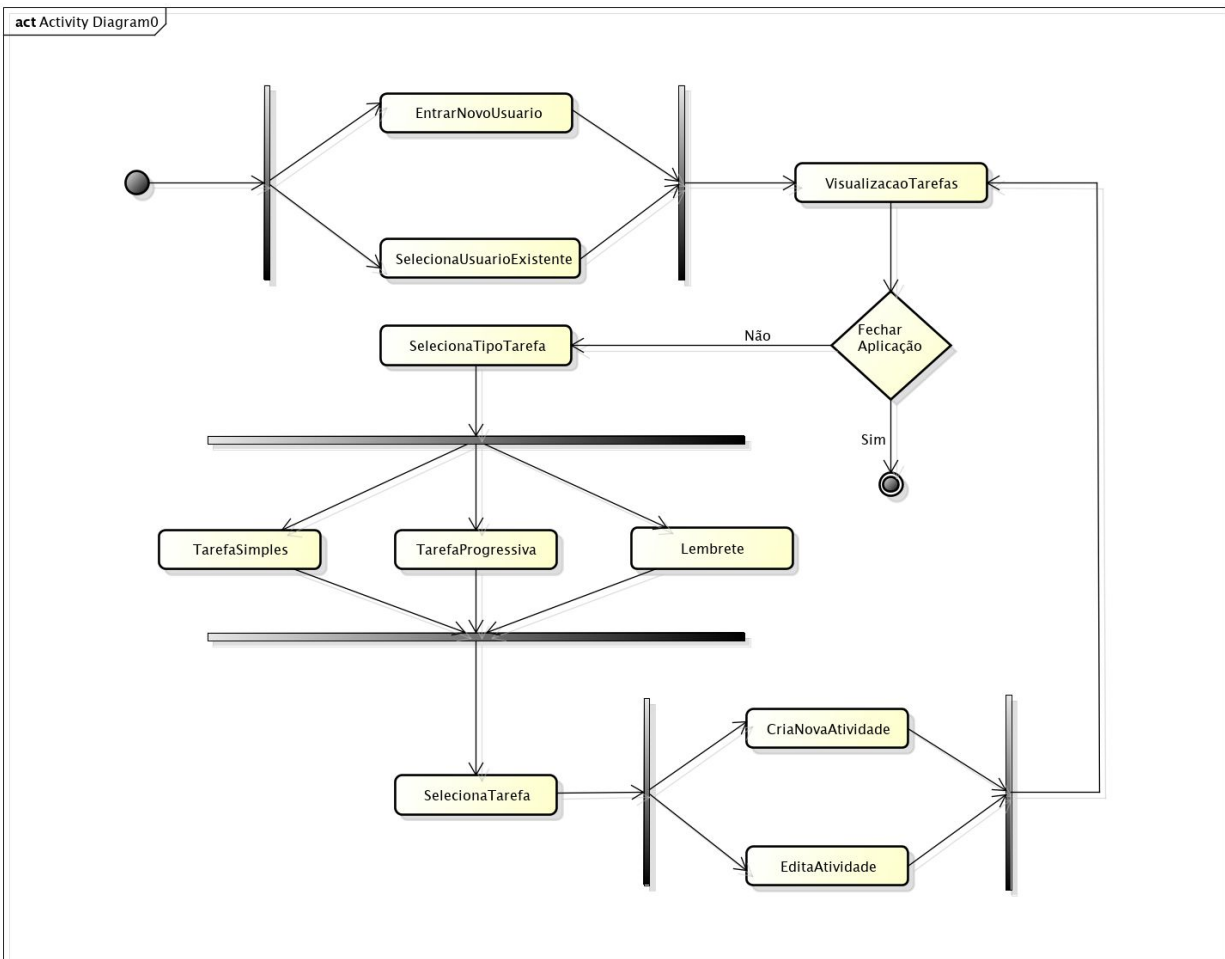
Pensamos em criar um aplicativo mais completo, pensando nas categorias de tarefas mais utilizadas. O nosso aplicativo foi desenvolvido para ser simples de usar e eficiente, contemplando as principais necessidades dos usuários. Incluímos então 3 Tipos de tarefas:

1. Tarefa simples  
Possui apenas Título e descrição
2. Tarefa progressiva  
Possui Título, Descrição, Data e Progresso. O progresso é um numero entre 0 e 100, e indica a porcentagem do progresso que o usuário esteja alcançando em cada momento da realização de sua tarefa.
3. Tarefa com Lembrete  
Possui Título, Descrição, Data e Hora. Por esta tarefa possuir Data e Hora, desenvolvemos um alarme para que quando o ÉFIFO TaskManager estiver aberto, este possa avisar ao usuário quando o tempo determinado da tarefa se esgotar.

### Implementação

O **ÉFIFO TaskManager** é uma aplicação de gerenciamento de tarefas bem completa, que permite que o usuário possa cadastrar tipos de tarefas distintas e acompanhar o seu andamento de forma simplificada e centralizada. Apesar de ser uma aplicação mais completa, seu fluxo de funcionamento é bem simples tornando a aplicação fácil de ser utilizada e com recursos importantes para o usuário.

Abaixo segue o **Diagrama de Atividades** que descreve o fluxo do programa com as opções que o usuário pode ter a cada ação realizada por ele:




Durante o processo de desenvolvimento da aplicação foi verificado a necessidade de utilização de diversas tecnologias, estas serão descritas a seguir nesta mesma secção.

## Interface Gráfica

A partir do nosso objetivo de oferecer um software simples e de fácil utilização, desenvolvemos uma interface amigável que simplifica a utilização e tornando a aplicação simples de ser utilizada. Para a implementação da interface gráfica foram utilizadas as próprias bibliotecas fornecidas pelo JAVA são elas:

- **java.swing.\*:** essa biblioteca foi escrita em “java puro”, foi usada em grande parte das interfaces implementadas pela sua portabilidade. Todos os componentes utilizados na aplicação são pertencentes a essa biblioteca, botões, caixa de texto, tabelas, entre outros recursos.
- **java.awt.\*:** essa biblioteca foi escrita utilizando recursos do sistema para sua utilização o que pode trazer alguns problemas, por isso ela foi utilizada basicamente para definir o layout da aplicação

Abaixo segue um exemplo de umas das janelas desenvolvidas com essas bibliotecas:



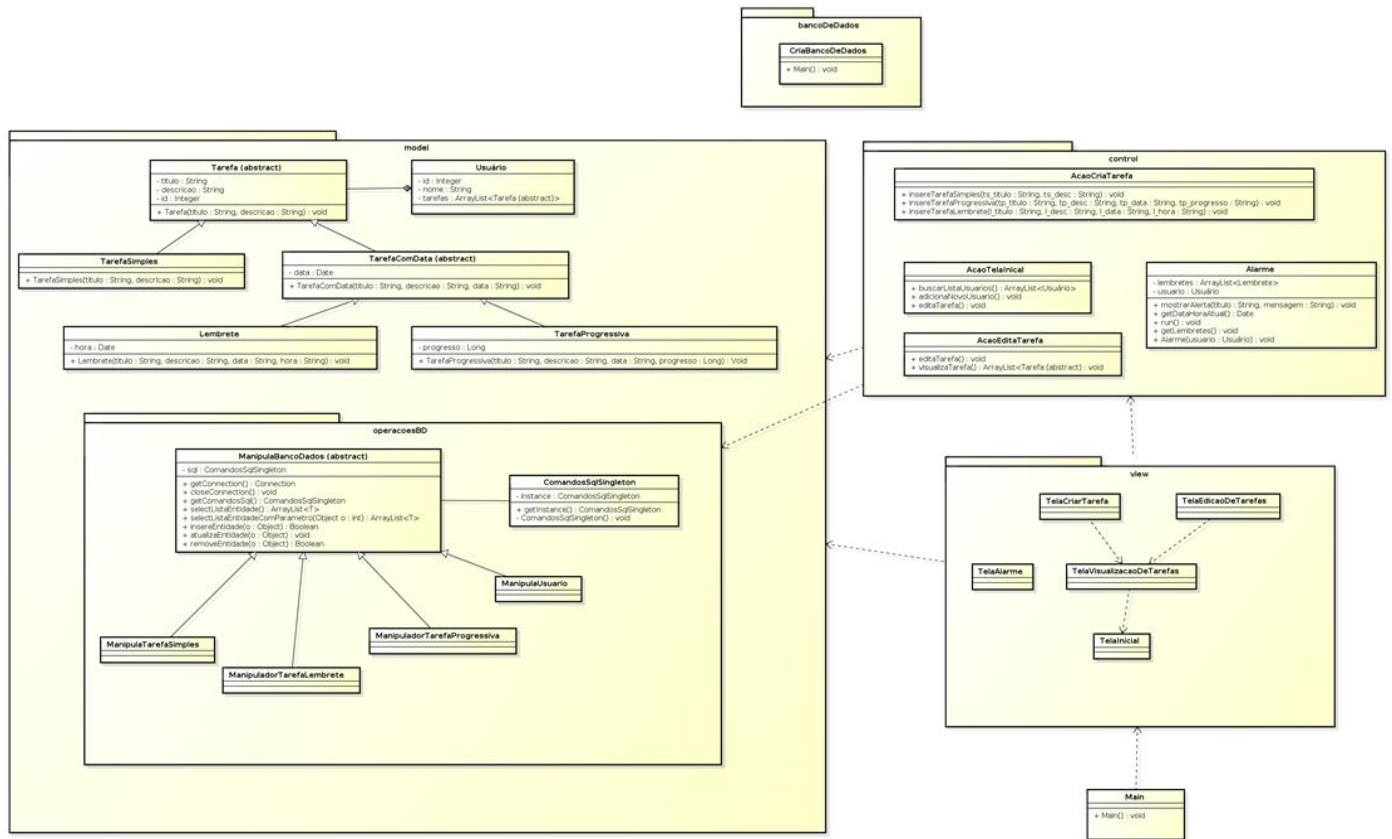
### Padrão de Projetos e Arquitetural:

Devido ao fato de o **ÉFIFO TaskManager** possuir interface gráfica, decidimos implementar todo o código fonte seguindo o padrão arquitetural e de projetos MVC que separa a representação da informação da interação do usuário com ela.

No pacote *model* foi inserido todas as classes que representam as tarefas de diferentes tipos e o usuário do sistema, além das operação de CRUD no banco de dados. no pacote *view* foram implementados todas as interfaces de interação com o usuário. E finalmente no pacote *control* foi desenvolvido a comunicação entre o usuário e as classes de entidade e persistencia.

Além das classes pertencentes aos pacotes MVC, foram necessario a criação de duas classes fora destes pacotes: a classe ***CriaBancoDeDados.java*** que é responsável por gerar um banco de dados caso seja necessário e a classe ***Main.java*** que executa a aplicação

Todas as classes desenvolvidas podem ser visualizadas no **Diagrama de Classes** a seguir, bem como os atributos e operações que atuam sobre elas:



Implementamos o padrão de projeto Singleton com o objetivo de centralizar todos os acessos ao banco de dados para fazer as operações de CRUD. Este padrão faz com que seja criada apenas uma conexão, onde todas as classes que utilizam esta mesma para realizar as operações necessárias

## Banco de Dados

Optamos durante o desenvolvimento do trabalho por persistir os dados em um banco de dados com o objetivo de facilitar o manuseio dos dados nas operações de inserção, remoção e atualização das tarefas. Como solução, utilizamos SQLite como banco de dados devido à sua simplicidade e leveza, e por não exigir a instalação de um grande banco de dados, nem de um SGBD no computador do Cliente.

## Thread

O alarme foi implementado como uma Thread que fica consultando as datas e horários dos lembretes cadastrados no banco a cada um minuto, mostrando uma tela de aviso com os dados da tarefa na hora especificada.

## Conclusão

Este trabalho foi bastante útil para implementar diversos novos conceitos que foram vistos em sala de aula nesta disciplina, como padrões de projeto, interface gráfica com AWT e Swing, melhores práticas de programação, orientação a objetos, UML, e vários outros conceitos da linguagem JAVA, além de Arquitetura de Software visto em Engenharia de Software.

A prática destes conceitos se mostrou altamente eficiente em nosso aprendizado, fazendo com que o ensino ministrado fosse realmente guardado em nossas cabeças, nos tornando desenvolvedores muito melhores.

## Referências

1. <http://www.devmedia.com.br/java-jtabbedpane-componente-de-formularios-em-java/1512> [Acessado em 11/06/2015 às 18:20]
2. <http://www.caelum.com.br/apostila-java-testes-xml-design-patterns/interfaces-graficas-com-swing/> [Acessado do dia 11/06/2015 ao 15/06/2015 ]
3. Slides do professor sobre interface gráfica e manipulação de eventos