Marcus Becker - contato: **meu**mundo**java**@gmail.com

Criando uma agenda simples com NetBeans 6.5

(Swing application framework e Beansbinding)

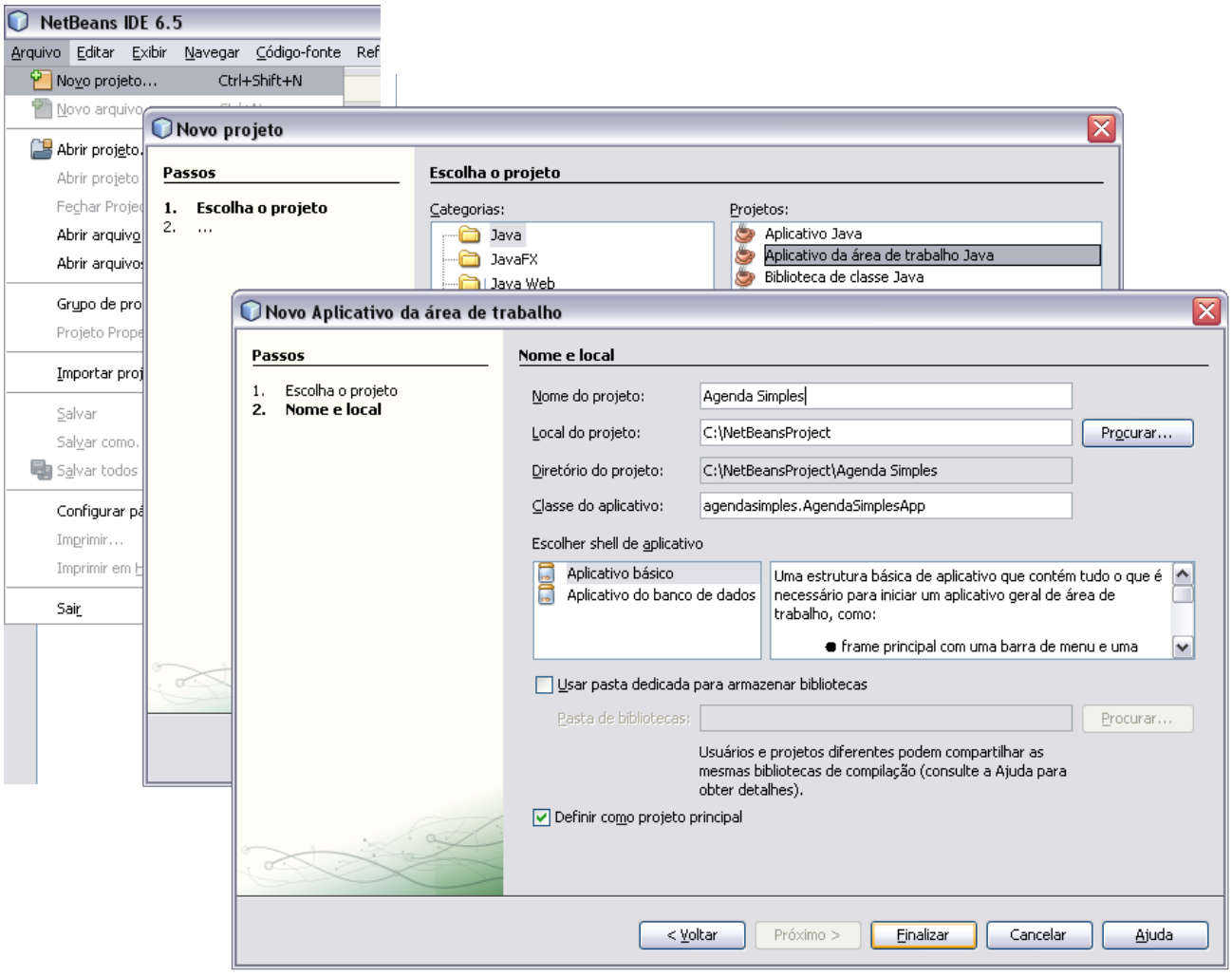
Já faz algum tempo que escrevi uma agenda simples usando o Eclipse com o Visual Class Editor.
Demorei em torno de uma dia para criar a agenda.
Hoje com as funções da nova versão do NetBeans, consigo criar essa mesma agenda em alguns minutos, dependendo, em alguns cliques.
O objetivo deste tutorial é mostrar isso, mas mantendo o conceito da primeira agenda simples que não usa banco de dados.

Legenda das cores:

Texo	Quando se trata de alguma propriedade, campo ou botão do NetBeans
Itálico	Nome de classe ou método

Iniciando:

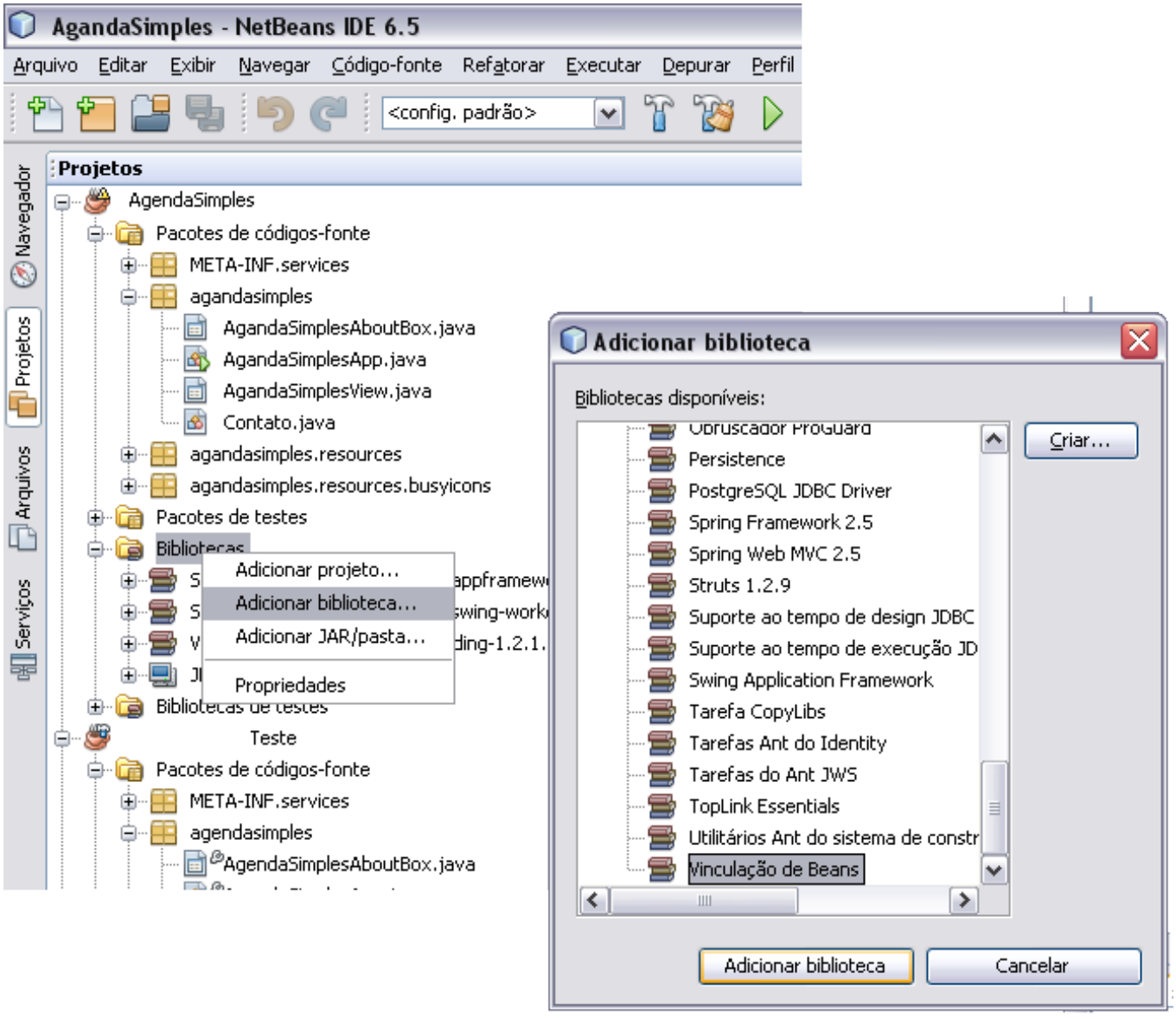
1. Crie um novo projeto no NetBeans.
2. Na Categoria Java escolha o projeto Aplicativo de área de trabalho Java
3. Clique em Próximo
4. Altere o Nome do projeto para Agenda Simples
5. Em Escolher shell de aplicativo marque a opção aplicativo básico
6. Clique em Finalizar



O NetBeans irá criar uma aplicação Desktop básica já funcional e três classes criadas *AgandaSimplesAboutBox*, *AgandaSimplesApp*, *AgandaSimplesView*.
Além de criar mais uma classe que será a representação do nosso contato, precisamos importar para o nosso projeto mais uma biblioteca. A biblioteca que será responsável pela vinculação dos componentes.

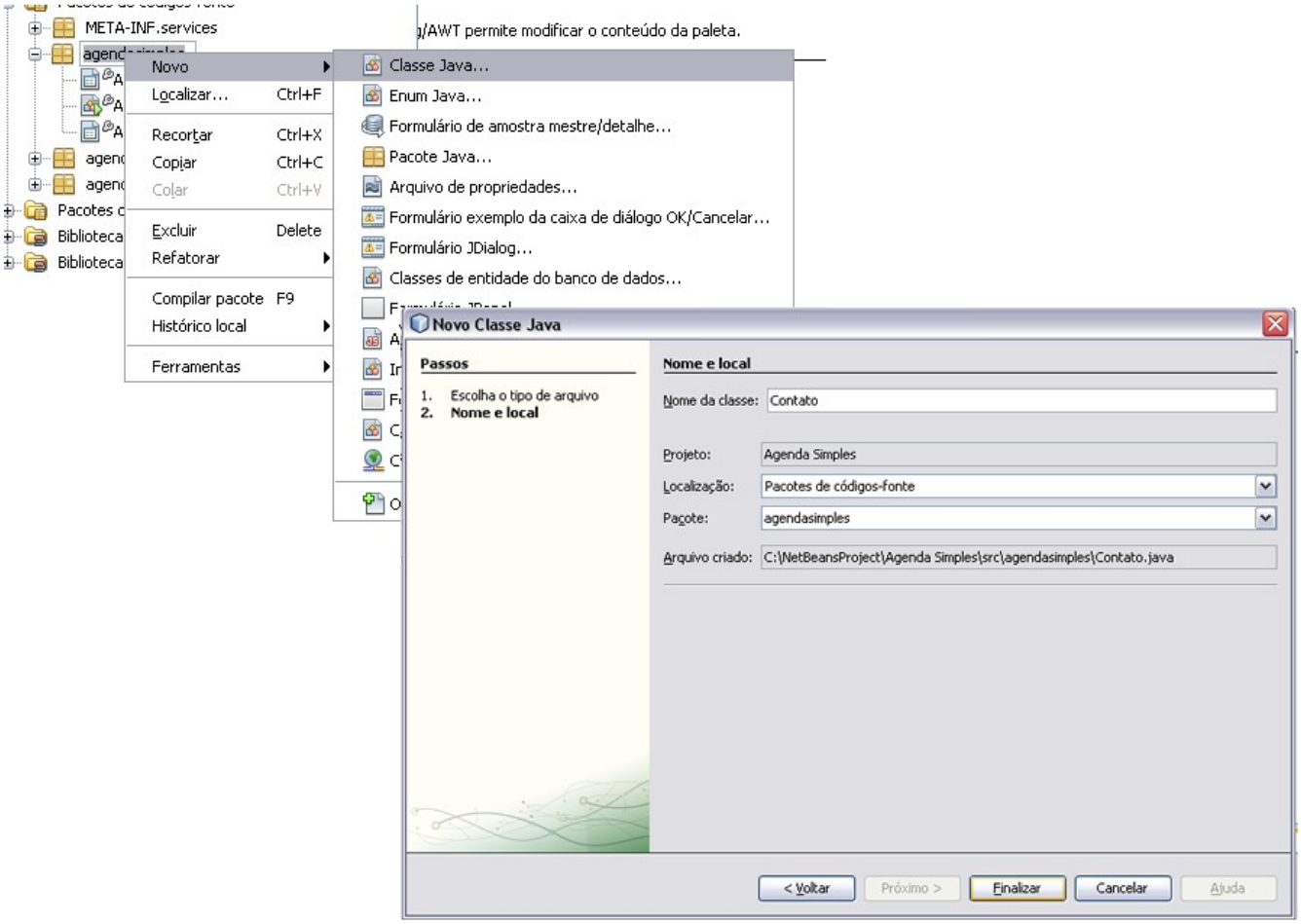
Importando biblioteca de vinculação:

1. Abra a janela Projetos e selecione a pasta Bibliotecas do seu projeto.
2. Clique com o botão direito em Biblioteca, no menu Adicionar biblioteca...
3. Selecione Vinculação de Beans (no final da lista) e clique em Adicionar Biblioteca



Criando a classe contato:

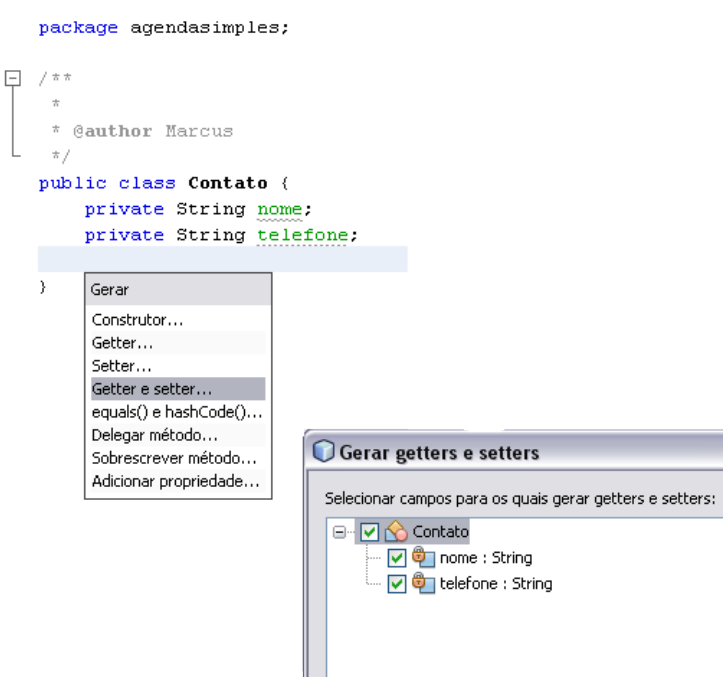
1. Clique com o botão direito no pacote agendasimples
2. Escolha Novo, depois Classe java
3. Altere o Nome da classe para Contato
4. Clique em Finalizar



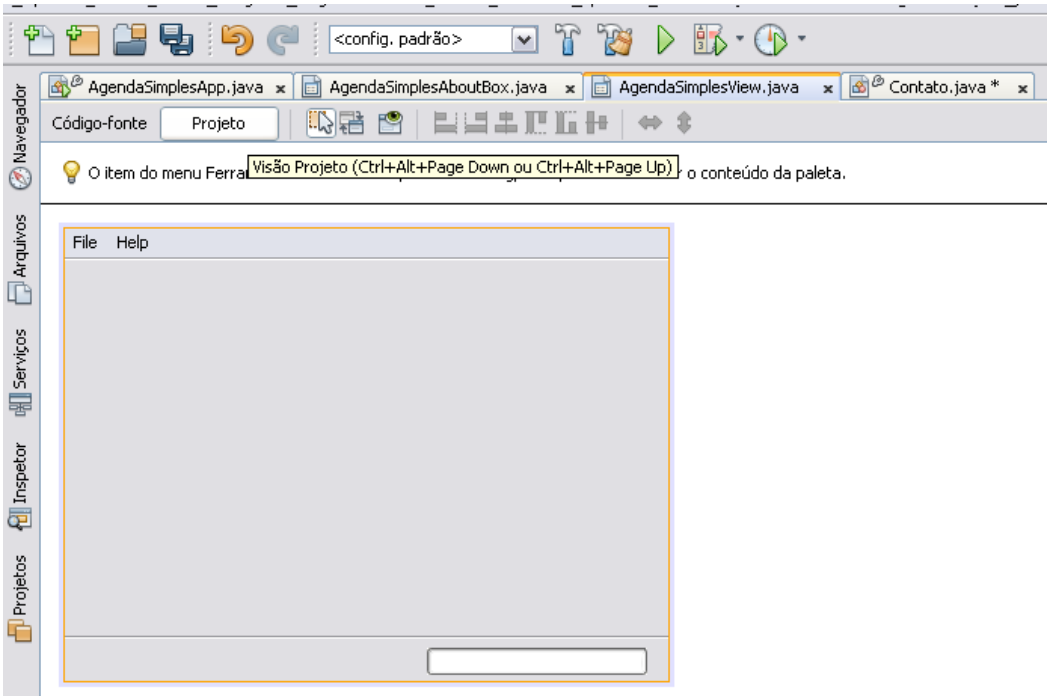
Nossa classe *Contato* precisá ser um POJO e seus campos serão *nome* e *telefone*.

Criando e encapulsando os campos:

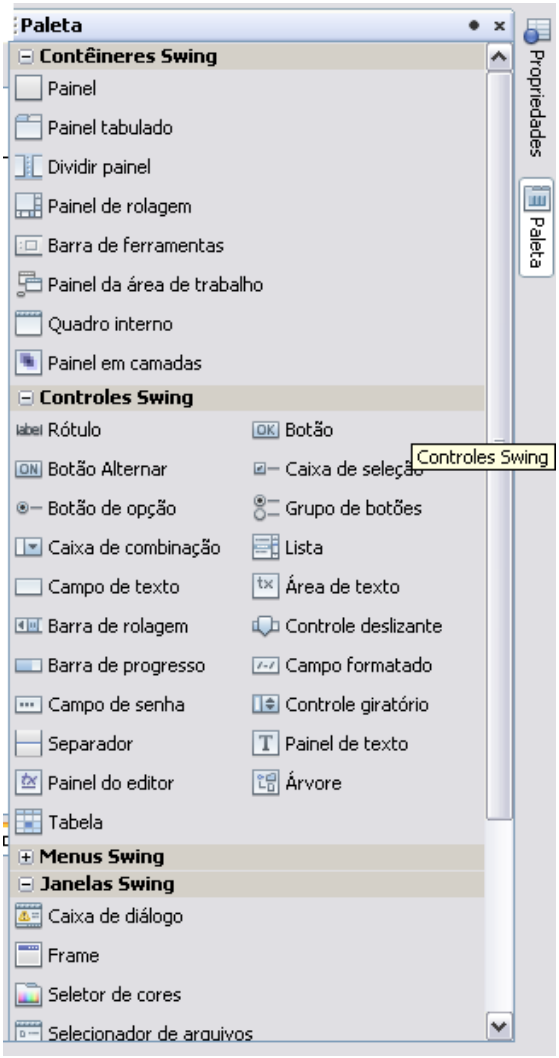
1. Crie dois campos privados do tipo String (*private String nome* e *private String telefone*)
2. Com o ponteiro do mouse entre os colchetes da classe, pressiona ALT + INSERT
3. No menu que apareceu, selecione Getter e Setter
4. Na janela Gerar getter e setter, selecione Contato. Isso selecionará todos os campos
5. Clique em Gerar e salve as alterações



Agora vamos trabalhar a camada visual,
abra a classe *AgandaSimplesView*.
Certifique-se de estar visualizando a janela Projeto
e não a janela Código-fonte.



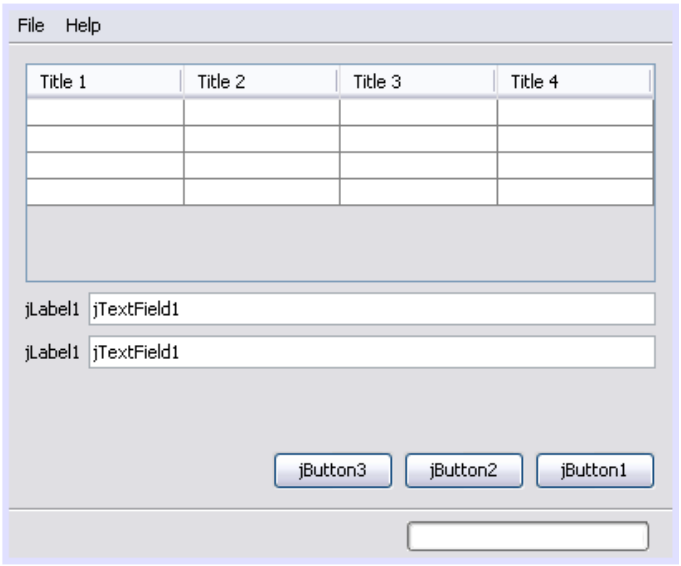
Arraste da Paletas os seguintes Controles Swing para o projeto:



Inserindo componentes e quantidades:

- Tabela - 1
- Rótulo - 2
- Campo de texto - 2
- Botão - 3

Alinhe como preferir ou como mostra a imagem



Nos itens Tabela e Campo de texto, vamos alterar o nome da variável para facilitar seu uso.
Repita os três passos seguindo a tabela.

1. Botão direito sobre o componente
2. Escolha a opção Alterar nome da variável...
3. Na janela Renomear / Novo nome
   coloque o nome correspondente

Componente	Novo nome
Tabela	tabela
1º Campo de texto	tfNome
2º Campo de texto	tfTelefone

Obs. Os botões não precisam ser renomeados.

Até aqui não vimos muita diferença entre o NetBeans e o Eclipse. Mas as semelhanças param por aqui.
Vamos fazer a listagem dos nossos contatos. Altere o modo de visualização para Código-fonte.
Crie um método que retorne um ArrayList contendo nome e telefone de nossos contatos, representado pela classe *Contato* que criamos anteriormente.

Criando método getContato():

1. Digite as linhas de código abaixo:

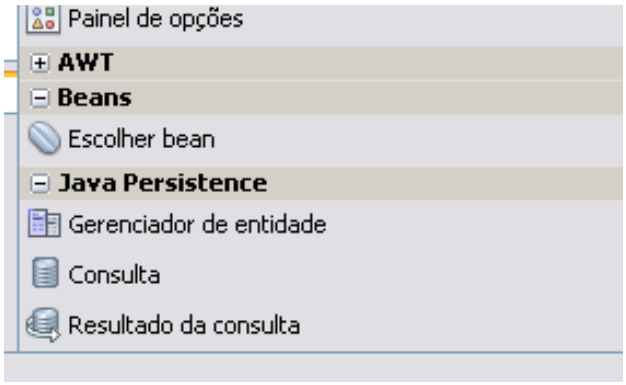
```
private List<Contato> getContato() {
    List<Contato> lista = new ArrayList<Contato>();
    // Criando um contato para teste:
    Contato c = new Contato();
    c.setNome("Marcus Becker - meumundojava@gmail.com");
    c.setTelefone("11 ****-*****");
    lista.add(c);

    return lista;
}
```

2. Atualize os imports com CTRL + SHIFT + I

Para exibir os dados em nossa tabela, precisamos vincular a tabela em um objeto List.
Depois usaremos alguns métodos que alterarão nosso objeto List (como o método getContato()) e essas alterações refletirão em nossa tabela.

Podemos criar o objeto List de duas formas:

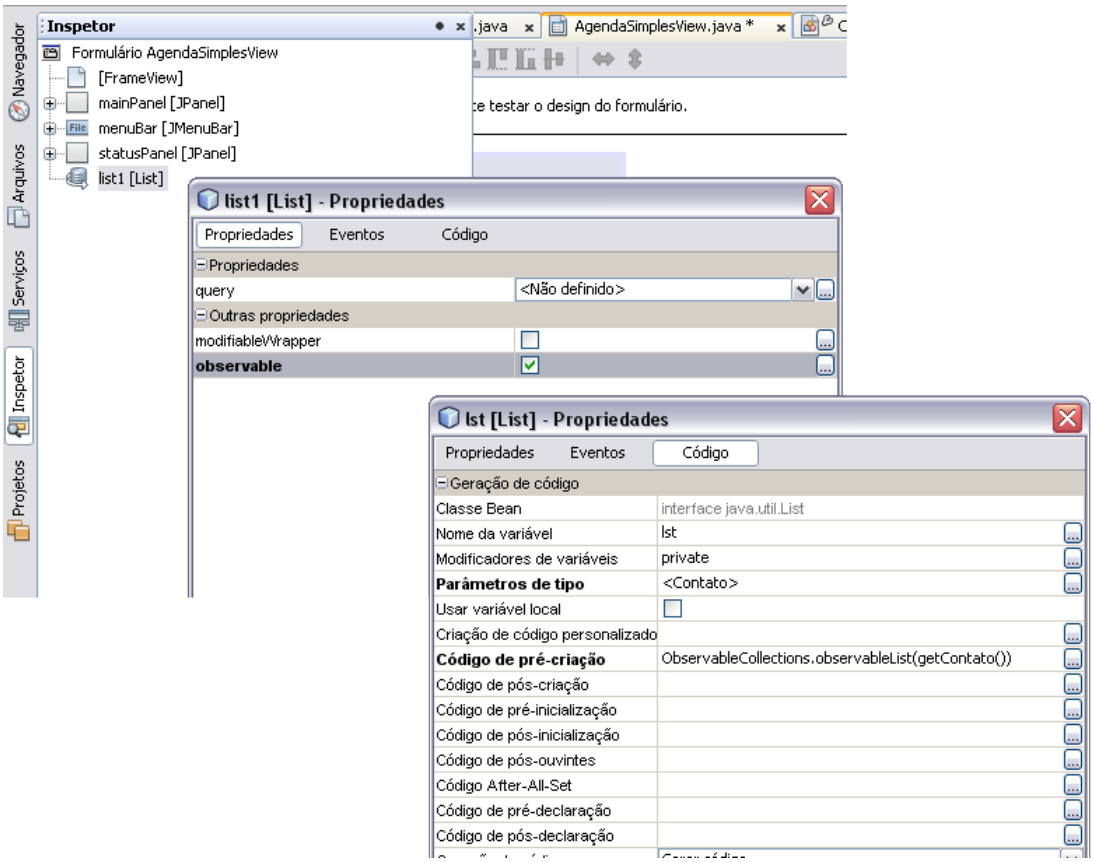
- Selecionar na Paleta Java Persistence o componente Resultado da Consulta.
- Selecionar na Paleta Beans o componente Escolher Bean e no campo Nome da classe digitar *java.util.List*



Depois de adicionar o componente usando alguma dessas duas formas, veremos um erro em nosso código. Isso ocorre porque o tipo de criação padrão do objeto List espera um objeto Query.
Vamos configurar nosso List. No Modo visual selecione-o na paleta Inspector

Configurando objeto List:

1. Selecione o componente list1, com o botão direito clique em Propriedades
2. Na aba Propriedades marque a caixa Observable
3. Na aba Código altere o Nome da variável para lst
4. Altere Parâmetros de tipo para <*Contato*>
   (Contato é o nome da classe que representa nossos dados. Não esqueça de usar < e >
5. O mais importante é alterar o campo Criação de código personalizado para *ObservableCollections.observableList(getContato())*
6. Na aba Código-fonte adicione os imports necessários CTRL + SHIFT + I
   (*import org.jdesktop.observablecollections.ObservableCollections;*)

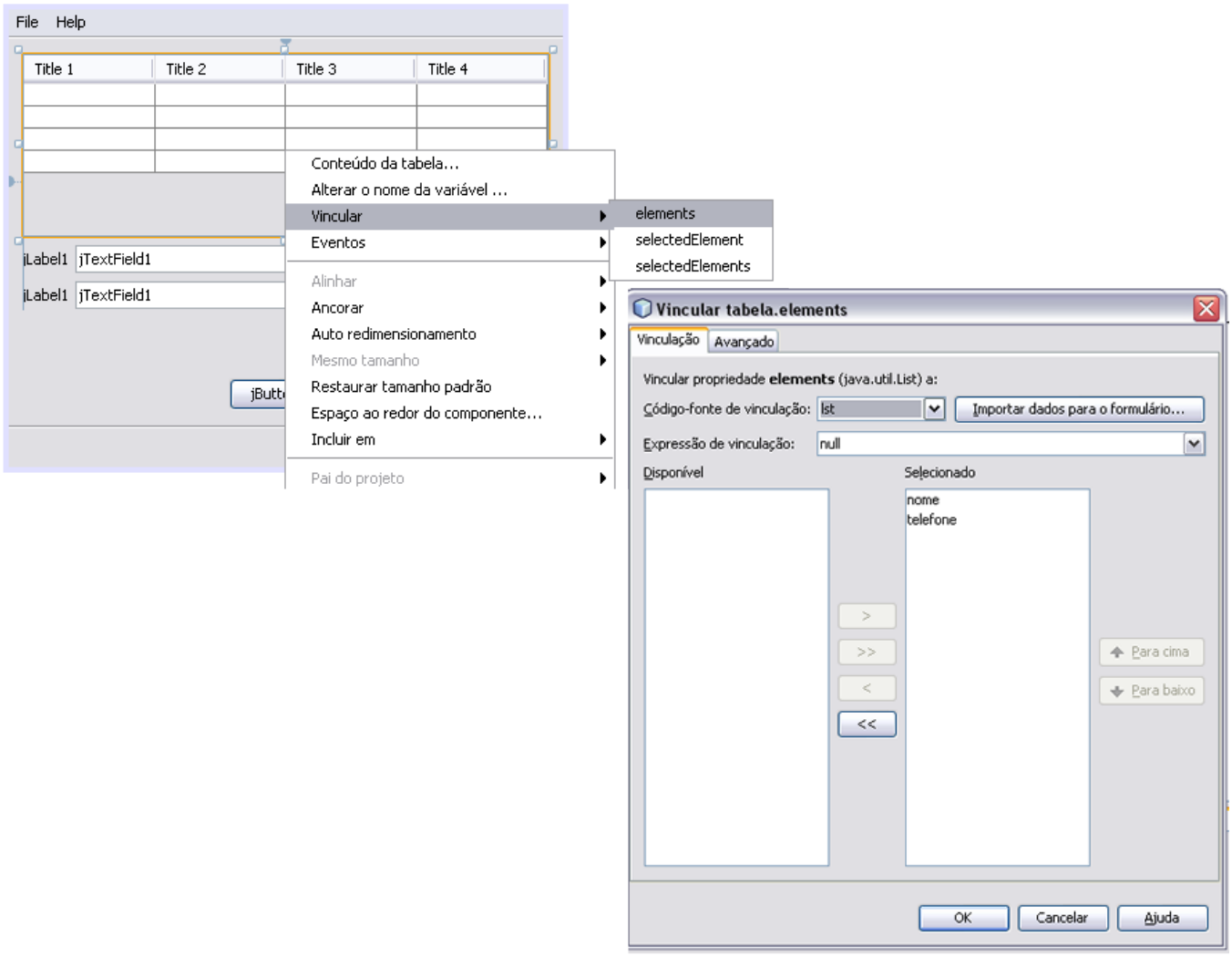


O próximo passo é associar nossa tabela ao objeto List.

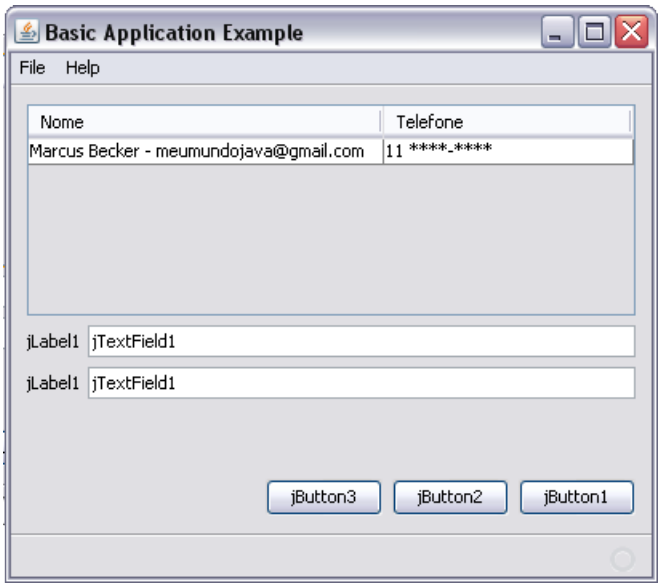
Com essa associação tornaremos o conteúdo da tabela o conteúdo da nossa lista.
E por usarmos uma *observableList,* todas as alterações na nossa List refletirão na tabela.

Vínculando Tabela ao objeto List:

1. Clique com o botão direito na Tabela escolhendo o item Víncular, após, Elements
2. Na opção Código-fonte de vinculação selecione lst (o nome do nosso recipiente *List<Contato>*)
3. Se tudo ocorreu bem, você verá os campos que declaramos na classe *Contato*. Clique em OK.

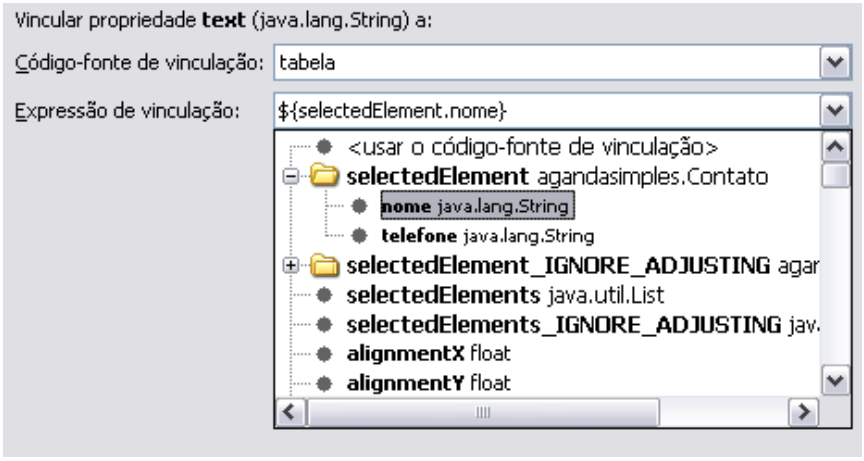


Execute o projeto, nossa tabela já estará exibindo os dados de teste que inserimos ao criar o método *getContato()*.
Agora será muito fácil inserir, remover e alterar os dados na agenda.Vamos ajustar os campos de texto para receber os valores da tabela.



Vinculando Texto à Tabela:

1. Clique com o botão direito no primeiro campo de texto escolhendo o item Víncular, após, Text
2. Na opção Código-fonte de vinculação selecione tabela (o nome da nossa Tabela)
3. Em Expressão de vinculação navegue até selectedElement e escolha o subitem nome
4. Repita esses passos para o segundo campo de texto, mas agora selecione o subitem telefone

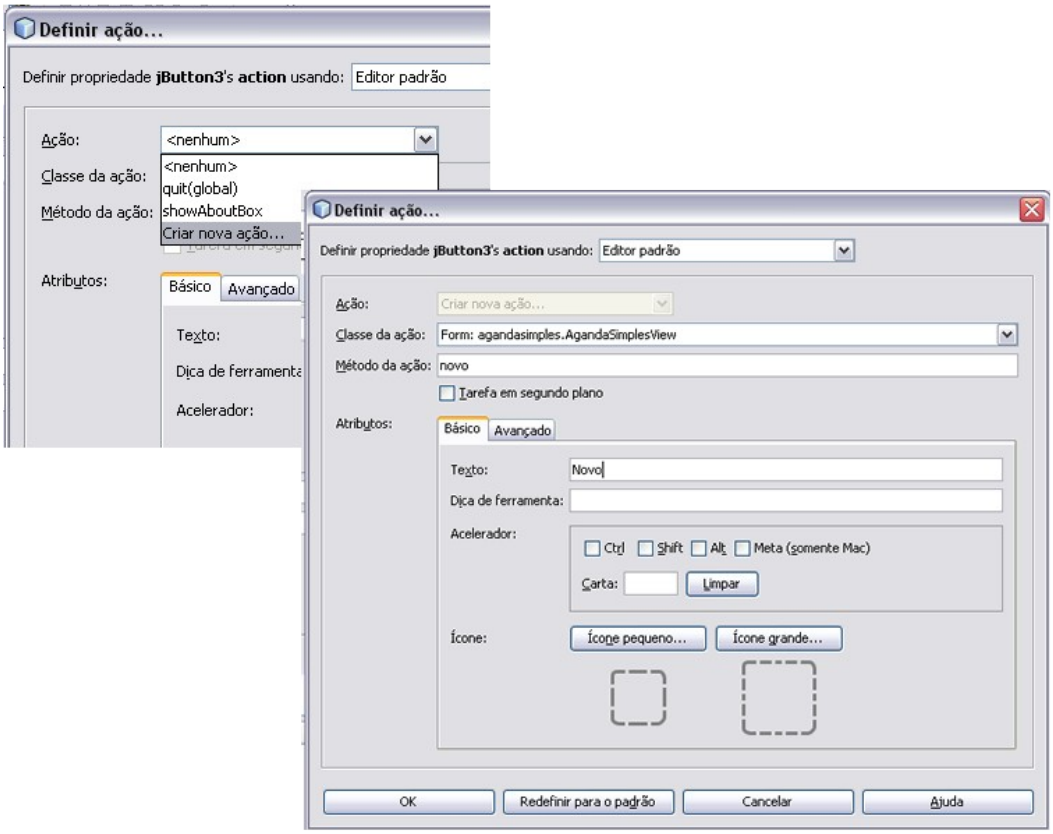


Execute o projeto e verifique que ao selecionar algum item da Tabela, os campos de texto são preenchidos automaticamente.

Vamos criar as funcionalidades dos botões. Temos três botões:
O primeiro será responsável por criar um novo registro.
O segundo salvará nossa List.
O terceiro será responsável por excluir algum contato.
Para os três botões, definiremos ações distintas. O processo será o mesmo, apenas alterando o nome da ação.

Criando ações:

1. Clique com o botão direito sobre o primeiro botão
2. Selecione o item Definir ação...
3. Na caixa de seleção Ação selecione Criar nova ação
4. No Método da ação digite *novo* (novo será o nome do método)
5. Em Atributos, aba Básico, item Texto digite Novo (Esse Novo será o texto do botão)
6. Quando clicar em OK, você irá para o novo método criado no Código-fonte. Volte para o modo Projeto
7. Repita os passos para os outros botões seguindo o mesmo padrão (*salvar*, Salvar e *excluir*, Excluir)



Não iremos focar no código dos botões, já que estamos mesmo interessados no NetBeans e sua facilidade de desenvolvimento. Mas junto com o tutorial segue o projeto que pode ser aberto pelo NetBeans e o aplicativo funcional (pasta dist).