

Commit

 This commit does not belong to any branch on this repository, and may belong to a fork outside of the repository.

constants - 5

Browse files

 KauanProg committed on Jun 19

1 parent [210b201](#) commit [b8e7203](#)

Showing 8 changed files with 16 additions and 18 deletions.

Whitespace







Ignore whitespace

Split

Unified

▼ BIN +47 Bytes (110%) [corretores.tmp](#) 

Binary file not shown.

▼  4     [src/Controler/ControleComprador.java](#) 

10	10	import java.io.FileOutputStream;
11	11	import java.io.ObjectInputStream;
12	12	import java.io.ObjectOutputStream;
13		- import java.io.Serializable;
14	13	import java.util.ArrayList;
15	14	import javax.swing.JOptionPane;
16	15	
20	19	*/
21	20	public class ControleComprador{
22	21	private ArrayList<Comprador> listaComprador = new ArrayList<Comprador>();
	22	+ final int MessageType = 1;
23	23	
24	24	public ControleComprador(){
25	25	desserializaComprador();
30	30	Comprador novoComprador = new Comprador(cpf, nome, email,fone,contatoPref);
31	31	listaComprador.add(novoComprador);
32	32	System.out.println("Cadastrado Comprador!");
33		- JOptionPane.showMessageDialog(null, "Comprador cadastrado!", "Sucesso", 1);
	33	+ JOptionPane.showMessageDialog(null, "Comprador cadastrado!", "Sucesso",
		MessageType);
34	34	}

```

35      35
36      36      public Comprador retornaCompradorPorCpf(String cpf) {

```

3 src/Controler/ControleCorretor.java

```

19      19      */
20      20      public class ControleCorretor {
21      21          private ArrayList<Corretor> listaCorretor = new ArrayList<Corretor>();
22      22      +      final int MessageType = 1;
23      23
24      24          public ControleCorretor(){
25      25              desserializaCorretor();
30      31              Corretor novoCorretor = new Corretor(cpf, nome,
                email,fone,creci,percCorretagem);
31      32              listaCorretor.add(novoCorretor);
32      33              System.out.println("Cadastrado Corretor!");
33      33      -      JOptionPane.showMessageDialog(null, "Corretor cadastrado!", "Sucesso", 1);
34      34      +      JOptionPane.showMessageDialog(null, "Corretor cadastrado!", "Sucesso",
                MessageType);
34      35          }
35      36
36      37      public Corretor retornaCorretorPorCpf(String cpf) {

```

4 src/Controler/ControleImovel.java

```

17      17      import java.io.ObjectOutputStream;
18      18      import java.util.ArrayList;
19      19      import java.util.Calendar;
20      20      - import java.util.Date;
21      21      import java.util.List;
22      22      import javax.swing.JOptionPane;
23      23
35      34          private ArrayList<Imovel> listaImovelProposta = new ArrayList<Imovel>();
36      35          private ArrayList<Imovel> listaImovelVendido = new ArrayList<Imovel>();
37      36          private List<String> tiposImovel = new ArrayList<String>();
37      37      +      final int MessageType = 1;
38      38
39      39          public ControleImovel(){
40      40              desserializaImovel();
51      51              Imovel novoImovel = new Imovel(codigo, tipo,
                endereco,descricao,foto,preco,comissao, Calendar.getInstance(),vendedorImovel);
52      52              listaImovel.add(novoImovel);
53      53              System.out.println("Cadastrado Imovel!");
54      54      -      JOptionPane.showMessageDialog(null, "Imóvel cadastrado!", "Sucesso", 1);
54      54      +      JOptionPane.showMessageDialog(null, "Imóvel cadastrado!", "Sucesso",
                MessageType);
55      55              if(tipo.equals(Util.LOTE)){
56      56                  listaImovelLote.add(novoImovel);
57      57              }else if(tipo.equals(Util.CASA)){

```

12 src/Controler/ControlePrincipal.java

```

21      21      import View.LimiteCadVendedor;
22      22      import View.LimiteConPropostas;
23      23      import View.LimitePrincipal;
24      - import java.awt.event.WindowEvent;
25      - import java.awt.event.WindowListener;
26      24      import java.util.ArrayList;
27      25      import java.util.Calendar;
28      26      import java.util.GregorianCalendar;
29      27      import java.util.List;
30      - import javax.swing.JFileChooser;
31      28      import javax.swing.JOptionPane;
32      29
33      30      /**
36      33      */
37      34      public class ControlePrincipal {
38      35
39      - //limite principal
40      - private LimitePrincipal limPrincipal;
41      -
42      36      //controladores
43      37      private ControleComprador ctrComprador;
44      38      private ControleVendedor ctrVendedor;
45      39      private ControleImovel ctrImovel;
46      40      private ControleCorretor ctrCorretor;
47      41      +
48      42      + final int MessageType = 0;
49      43
50      44      public ControlePrincipal(){
51      - limPrincipal = new LimitePrincipal(this);
52      + new LimitePrincipal(this);
53      46
54      47      ctrComprador = new ControleComprador();
55      48      ctrVendedor = new ControleVendedor();
141      137      Vendedor vendedorImovel =
142      138      ctrVendedor.retornaVendedorPorCpf(cpfVendedor);
143      139      ctrImovel.cadastraImovel(codigo, tipo, descricao, endereco, foto,
144      140      preco, comissao, vendedorImovel);
145      141      }else{
146      142      - JOptionPane.showMessageDialog(null, "Não existe o vendedor com o CPF
147      143      especificado!", "Erro", 0);
148      140      + JOptionPane.showMessageDialog(null, "Não existe o vendedor com o CPF
149      141      especificado!", "Erro", MessageType);
150      142      System.out.println("Não existe o vendedor com o CPF especificado.");
151      143      }
152      144      }

```

4 src/Controler/ControleVendedor.java

```

21      21      public class ControleVendedor implements Serializable{

```

```

22      22      private ArrayList<Vendedor> listaVendedor = new ArrayList<Vendedor>();
23      23
24      24      + final int MessageType = 1;
25      25      +
26      26      public ControleVendedor(){
27      27          desserializaVendedor();
28      28      }
29      29
30      32      Vendedor novoVendedor = new Vendedor(cpf, nome, email,fone,contatoPref);
31      33      listaVendedor.add(novoVendedor);
32      34      System.out.println("Cadastrado Vendedor!");
33      35      - JOptionPane.showMessageDialog(null, "Vendedor cadastrado!", "Sucesso", 1);
34      35      + JOptionPane.showMessageDialog(null, "Vendedor cadastrado!", "Sucesso",
35      36      MessageType);
36      36      }
37      37
38      38      public boolean existeVendedor(String cpf){

```

▼ ↕ 3 ■■■■■ src/Model/Comprador.java 📄

```

8      8      private String contatoPref;
9      9      private ArrayList<String> listaTipoImovelCompra;
10     10
11     11      - public Comprador(String cpf, String nome, String email, String fone,
12     12      - String contatoPref) {
13     11      + public Comprador(String cpf, String nome, String email, String fone, String
14     12      contatoPref) {
15     13          super(cpf, nome, email, fone);
16     14          this.contatoPref = contatoPref;
17     15      }

```

▼ ↕ 4 ■■■■■ src/TestaImpl.java 📄

```

6      6      public class TestaImpl {
7      7
8      8      public static void main(String[] args) {
9      9      - /*
10     9      +
11     10      ArrayList<Imovel> listaImoveis = new ArrayList();
12     11
13     12      // Criando vendedores, compradores e corretores
14     12      }
15     13
16     14      }
17     15      - /*
18     16      +
19     17      //cria o controlador principal
20     18      new ControlePrincipal();
21     19      }

```

0 comments on commit **b8e7203**

Please [sign in](#) to comment.