



Commit

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

refactor long method

Browse files

 Felipe-Gs committed on Jun 18

1 parent 282d7c5 commit 6b72278

Showing 1 changed file with 147 additions and 125 deletions.

Whitespace Ignore whitespace Split Unified

272

openodonto-web/src/main/java/br/ueg/openodonto/controle/busca/AbstractSearchable.java

```
16      16      import br.ueg.openodonto.servico.busca.SearchFilterBase;
17      17      import br.ueg.openodonto.servico.busca.Searchable;
18      18
19      - public abstract class AbstractSearchable<T> implements Searchable<T>,Serializable {
20      -
21      -     private static final long serialVersionUID = 8694703534298665402L;
22      -
23      -     private List<FieldFacade>          facade;
24      -     private Map<String,SearchFilter>    filtersMap;
25      -     private Map<String,InputMask>      masksMap;
26      -     private MessageDisplayer           displayer;
27      -     private ManageExample<T>           manageExample;
28      -
29      -     private List<SearchFilter>          filtersList;
30      -     private List<InputMask>            masksList;
31      -
32      -
33      -     public AbstractSearchable(Class<T> classe,MessageDisplayer displayer) {
34      -         this.displayer = displayer;
35      -         buildMask();
36      -         buildFilter();
37      -         buildFacade();
38      -         this.manageExample = new ManageExample<T>(classe);
39      -         filtersList = new ArrayList<SearchFilter>(filtersMap.values());
40      -         masksList = new ArrayList<InputMask>(masksMap.values());
```

```

41     }
42
43     protected <E>SearchFilterBase buildBasicFilter(String name,String
label,InputMask mask,List<E> domain,Validator... validator){
44         SearchFilterBase filter = new
SearchFilterBase(null,name,label,displayer);
45         SearchFilterBase.Field field = filter.new Field();
46         filter.setField(field);
47         SearchFilterBase.Input<E> input = filter.new Input<E>();
48         input.getValidators().addAll(Arrays.asList(validator));
49         input.setMask(mask);
50         input.setDomain(domain);
51         field.getInputFields().add(input);
52         return filter;
53     }
54
55     protected SearchFilterBase buildBasicFilter(String name,String
label,InputMask mask,Validator... validator){
56         return buildBasicFilter(name,label,mask,null,validator);
57     }
58
59     protected <E>SearchFilterBase buildBasicFilter(String name,String
label,List<E> domain,Validator... validator){
60         return buildBasicFilter(name,label,null,domain,validator);
61     }
62
63     protected SearchFilterBase buildBasicFilter(String name,String
label,Validator... validator){
64         return buildBasicFilter(name,label,null,null,validator);
65     }
66
67     public abstract T buildExample();
68
69     protected void buildFacade(){
70         facade = new ArrayList<FieldFacade>();
71     }
72
73     protected void buildFilter(){
74         filtersMap = new HashMap<String,SearchFilter>();
75     }
76
77     protected void buildMask(){
78         masksMap = new HashMap<String,InputMask>();
79     }
80
81     @Override
82     public List<FieldFacade> getFacade() {
83         return facade;
84     }
85
86     @Override
87     public List<SearchFilter> getFilters() {

```

```
88         return filtersList;
89     }
90
91     @Override
92     public List<InputMask> getMasks() {
93         return masksList;
94     }
95
96     public ManageExample<T> getManageExample() {
97         return manageExample;
98     }
99
100    public void setManageExample(ManageExample<T> manageExample) {
101        this.manageExample = manageExample;
102    }
103
104    public Map<String, SearchFilter> getFiltersMap() {
105        return filtersMap;
106    }
107
108    public void setFiltersMap(Map<String, SearchFilter> filtersMap) {
109        this.filtersMap = filtersMap;
110    }
111
112    public Map<String, InputMask> getMasksMap() {
113        return masksMap;
114    }
115
116    public void setMasksMap(Map<String, InputMask> masksMap) {
117        this.masksMap = masksMap;
118    }
119
120    public List<SearchFilter> getFiltersList() {
121        return filtersList;
122    }
123
124    public void setFiltersList(List<SearchFilter> filtersList) {
125        this.filtersList = filtersList;
126    }
127
128    public List<InputMask> getMasksList() {
129        return masksList;
130    }
131
132    public void setMasksList(List<InputMask> masksList) {
133        this.masksList = masksList;
134    }
135
136    public MessageDisplayer getDisplayer() {
137        return displayer;
138    }
139
```

```

140 -         public void setDisplayer(MessageDisplayer displayer) {
141 -             this.displayer = displayer;
142 -         }
143 -
19 + public abstract class AbstractSearchable<T> implements Searchable<T>, Serializable {
20 +
21 +     private static final long serialVersionUID = 8694703534298665402L;
22 +
23 +     private List<FieldFacade> facade;
24 +     private Map<String, SearchFilter> filtersMap;
25 +     private Map<String, InputMask> masksMap;
26 +     private MessageDisplayer displayer;
27 +     private ManageExample<T> manageExample;
28 +
29 +     private List<SearchFilter> filtersList;
30 +     private List<InputMask> masksList;
31 +
32 +     public AbstractSearchable(Class<T> classe, MessageDisplayer displayer) {
33 +         this.displayer = displayer;
34 +         buildMask();
35 +         buildFilter();
36 +         buildFacade();
37 +         this.manageExample = new ManageExample<T>(classe);
38 +         filtersList = new ArrayList<SearchFilter>(filtersMap.values());
39 +         masksList = new ArrayList<InputMask>(masksMap.values());
40 +     }
41 +
42 +     protected <E> SearchFilterBase buildBasicFilter(String name, String label,
43 +         FilterConfig<E> config) {
44 +         SearchFilterBase filter = new SearchFilterBase(null, name, label,
45 +             displayer);
46 +         SearchFilterBase.Field field = filter.new Field();
47 +         filter.setField(field);
48 +         SearchFilterBase.Input<E> input = filter.new Input<E>();
49 +         input.getValidators().addAll(Arrays.asList(config.getValidators()));
50 +         input.setMask(config.getMask());
51 +         input.setDomain(Arrays.asList(config.getDomain()));
52 +         field.getInputFields().add(input);
53 +         return filter;
54 +     }
55 +
56 +     protected static class FilterConfig<E> {
57 +         private InputMask mask;
58 +         private E[] domain;
59 +         private Validator[] validators;
60 +
61 +         public FilterConfig(InputMask mask, E[] domain, Validator... validators) {
62 +             this.mask = mask;
63 +             this.domain = domain;
64 +             this.validators = validators;
65 +         }
66 +     }

```

```
65 +         public InputMask getMask() {
66 +             return mask;
67 +         }
68 +
69 +         public E[] getDomain() {
70 +             return domain;
71 +         }
72 +
73 +         public Validator[] getValidators() {
74 +             return validators;
75 +         }
76 +     }
77 +
78 +     protected <E> SearchFilterBase buildBasicFilter(String name, String label,
79 + InputMask mask, Validator... validator) {
80 +         return buildBasicFilter(name, label, new FilterConfig<>(mask, null,
81 + validator));
82 +     }
83 +
84 +     protected <E> SearchFilterBase buildBasicFilter(String name, String label,
85 + List<E> domain, Validator... validator) {
86 +         return buildBasicFilter(name, label, new FilterConfig<>(null,
87 + domain.toArray(), validator));
88 +     }
89 +
90 +     protected <E> SearchFilterBase buildBasicFilter(String name, String label,
91 + Validator... validator) {
92 +         return buildBasicFilter(name, label, new FilterConfig<>(null, null,
93 + validator));
94 +     }
95 +
96 +     public abstract T buildExample();
97 +
98 +     protected void buildFacade() {
99 +         facade = new ArrayList<FieldFacade>();
100 +     }
101 +
102 +     protected void buildFilter() {
103 +         filtersMap = new HashMap<String, SearchFilter>();
104 +     }
105 +
106 +     protected void buildMask() {
107 +         masksMap = new HashMap<String, InputMask>();
108 +     }
109 +
110 +     @Override
111 +     public List<FieldFacade> getFacade() {
112 +         return facade;
113 +     }
114 +
115 +     @Override
116 +     public List<SearchFilter> getFilters() {
```

```
111 +         return filtersList;
112 +     }
113 +
114 +     @Override
115 +     public List<InputMask> getMasks() {
116 +         return masksList;
117 +     }
118 +
119 +     public ManageExample<T> getManageExample() {
120 +         return manageExample;
121 +     }
122 +
123 +     public void setManageExample(ManageExample<T> manageExample) {
124 +         this.manageExample = manageExample;
125 +     }
126 +
127 +     public Map<String, SearchFilter> getFiltersMap() {
128 +         return filtersMap;
129 +     }
130 +
131 +     public void setFiltersMap(Map<String, SearchFilter> filtersMap) {
132 +         this.filtersMap = filtersMap;
133 +     }
134 +
135 +     public Map<String, InputMask> getMasksMap() {
136 +         return masksMap;
137 +     }
138 +
139 +     public void setMasksMap(Map<String, InputMask> masksMap) {
140 +         this.masksMap = masksMap;
141 +     }
142 +
143 +     public List<SearchFilter> getFiltersList() {
144 +         return filtersList;
145 +     }
146 +
147 +     public void setFiltersList(List<SearchFilter> filtersList) {
148 +         this.filtersList = filtersList;
149 +     }
150 +
151 +     public List<InputMask> getMasksList() {
152 +         return masksList;
153 +     }
154 +
155 +     public void setMasksList(List<InputMask> masksList) {
156 +         this.masksList = masksList;
157 +     }
158 +
159 +     public MessageDisplayer getDisplayer() {
160 +         return displayer;
161 +     }
162 +
```

```
163 + public void setDisplayer(MessageDisplayer displayer) {  
164 +     this.displayer = displayer;  
165 + }  
144 166 }
```

0 comments on commit `6b72278`

Please [sign in](#) to comment.