

Commit

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


Refatoração em bo

Browse files

LoginBO - Técnicas: Extract Method, Replace with Symbolic Constant, Decompose Conditional

PessoaBO - Técnicas: Extract Method, Move Method

UfBO - Técnicas: Extract Method

 moisesocosta committed on Jun 21

1 parent [9d330b1](#) commit [807ee02](#)

Showing 3 changed files with 66 additions and 82 deletions.

Whitespace

Ignore whitespace

Split

Unified

42

src/br/sistema/crud/jdbc/bo/LoginBO.java



```
...      ...      @@ -1,32 +1,30 @@
1      1      package br.sistema.crud.jdbc.bo;
2      2
3      3      import br.sistema.crud.jdbc.dao.LoginDAO;
4      - import br.sistema.crud.jdbc.dao.PessoaDAO;
5      4      import br.sistema.crud.jdbc.dto.LoginDTO;
6      - import br.sistema.crud.jdbc.dto.PessoaDTO;
7      5      import br.sistema.crud.jdbc.exception.NegocioException;
8      6
9      7      public class LoginBO {
10     -
11     -         public boolean logar(LoginDTO loginDTO) throws NegocioException{
12     -             boolean resultado = false;
13     -             try{
14     -                 if(loginDTO.getNome() == null || "".equals(loginDTO)){
15     -                     throw new NegocioException("Logn Obrigatorio");
16     -                 }else if(loginDTO.getSenha() == null ||
17     -                 "".equals(loginDTO.getSenha())){
18     -                     throw new NegocioException("Logn Obrigatorio");
```

```

18      8
19      -
20      -
21      -
22      -
23      -
24      -
25      -
26      -
27      -
28      -
29      -
30      -
31      9 +
32      10 +
33      11 +
34      12 +
35      13 +
36      14 +
37      15 +
38      16 +
39      17 +
40      18 +
41      19 +
42      20 +
43      21 +
44      22 +
45      23 +
46      24 +
47      25 +
48      26 +
49      27 +
50      28 +
51      29 +
52      30 +
53      31 +
54      32 +
55      33 +
56      34 +
57      35 +
58      36 +
59      37 +
60      38 +
61      39 +
62      40 +
63      41 +
64      42 +
65      43 +
66      44 +
67      45 +
68      46 +
69      47 +
70      48 +
71      49 +
72      50 +
73      51 +
74      52 +
75      53 +
76      54 +
77      55 +
78      56 +
79      57 +
80      58 +
81      59 +
82      60 +
83      61 +
84      62 +
85      63 +
86      64 +
87      65 +
88      66 +
89      67 +
90      68 +
91      69 +
92      70 +
93      71 +
94      72 +
95      73 +
96      74 +
97      75 +
98      76 +
99      77 +
100     78 +
101     79 +
102     80 +
103     81 +
104     82 +
105     83 +
106     84 +
107     85 +
108     86 +
109     87 +
110     88 +
111     89 +
112     90 +
113     91 +
114     92 +
115     93 +
116     94 +
117     95 +
118     96 +
119     97 +
120     98 +
121     99 +
122     100 +
123     101 +
124     102 +
125     103 +
126     104 +
127     105 +
128     106 +
129     107 +
130     108 +
131     109 +
132     110 +
133     111 +
134     112 +
135     113 +
136     114 +
137     115 +
138     116 +
139     117 +
140     118 +
141     119 +
142     120 +
143     121 +
144     122 +
145     123 +
146     124 +
147     125 +
148     126 +
149     127 +
150     128 +
151     129 +
152     130 +
153     131 +
154     132 +
155     133 +
156     134 +
157     135 +
158     136 +
159     137 +
160     138 +
161     139 +
162     140 +
163     141 +
164     142 +
165     143 +
166     144 +
167     145 +
168     146 +
169     147 +
170     148 +
171     149 +
172     150 +
173     151 +
174     152 +
175     153 +
176     154 +
177     155 +
178     156 +
179     157 +
180     158 +
181     159 +
182     160 +
183     161 +
184     162 +
185     163 +
186     164 +
187     165 +
188     166 +
189     167 +
190     168 +
191     169 +
192     170 +
193     171 +
194     172 +
195     173 +
196     174 +
197     175 +
198     176 +
199     177 +
200     178 +
201     179 +
202     180 +
203     181 +
204     182 +
205     183 +
206     184 +
207     185 +
208     186 +
209     187 +
210     188 +
211     189 +
212     190 +
213     191 +
214     192 +
215     193 +
216     194 +
217     195 +
218     196 +
219     197 +
220     198 +
221     199 +
222     200 +
223     201 +
224     202 +
225     203 +
226     204 +
227     205 +
228     206 +
229     207 +
230     208 +
231     209 +
232     210 +
233     211 +
234     212 +
235     213 +
236     214 +
237     215 +
238     216 +
239     217 +
240     218 +
241     219 +
242     220 +
243     221 +
244     222 +
245     223 +
246     224 +
247     225 +
248     226 +
249     227 +
250     228 +
251     229 +
252     230 +
253     231 +
254     232 +
255     233 +
256     234 +
257     235 +
258     236 +
259     237 +
260     238 +
261     239 +
262     240 +
263     241 +
264     242 +
265     243 +
266     244 +
267     245 +
268     246 +
269     247 +
270     248 +
271     249 +
272     250 +
273     251 +
274     252 +
275     253 +
276     254 +
277     255 +
278     256 +
279     257 +
280     258 +
281     259 +
282     260 +
283     261 +
284     262 +
285     263 +
286     264 +
287     265 +
288     266 +
289     267 +
290     268 +
291     269 +
292     270 +
293     271 +
294     272 +
295     273 +
296     274 +
297     275 +
298     276 +
299     277 +
300     278 +
301     279 +
302     280 +
303     281 +
304     282 +
305     283 +
306     284 +
307     285 +
308     286 +
309     287 +
310     288 +
311     289 +
312     290 +
313     291 +
314     292 +
315     293 +
316     294 +
317     295 +
318     296 +
319     297 +
320     298 +
321     299 +
322     300 +
323     301 +
324     302 +
325     303 +
326     304 +
327     305 +
328     306 +
329     307 +
330     308 +
331     309 +
332     310 +
333     311 +
334     312 +
335     313 +
336     314 +
337     315 +
338     316 +
339     317 +
340     318 +
341     319 +
342     320 +
343     321 +
344     322 +
345     323 +
346     324 +
347     325 +
348     326 +
349     327 +
350     328 +
351     329 +
352     330 +
353     331 +
354     332 +
355     333 +
356     334 +
357     335 +
358     336 +
359     337 +
360     338 +
361     339 +
362     340 +
363     341 +
364     342 +
365     343 +
366     344 +
367     345 +
368     346 +
369     347 +
370     348 +
371     349 +
372     350 +
373     351 +
374     352 +
375     353 +
376     354 +
377     355 +
378     356 +
379     357 +
380     358 +
381     359 +
382     360 +
383     361 +
384     362 +
385     363 +
386     364 +
387     365 +
388     366 +
389     367 +
390     368 +
391     369 +
392     370 +
393     371 +
394     372 +
395     373 +
396     374 +
397     375 +
398     376 +
399     377 +
400     378 +
401     379 +
402     380 +
403     381 +
404     382 +
405     383 +
406     384 +
407     385 +
408     386 +
409     387 +
410     388 +
411     389 +
412     390 +
413     391 +
414     392 +
415     393 +
416     394 +
417     395 +
418     396 +
419     397 +
420     398 +
421     399 +
422     400 +
423     401 +
424     402 +
425     403 +
426     404 +
427     405 +
428     406 +
429     407 +
430     408 +
431     409 +
432     410 +
433     411 +
434     412 +
435     413 +
436     414 +
437     415 +
438     416 +
439     417 +
440     418 +
441     419 +
442     420 +
443     421 +
444     422 +
445     423 +
446     424 +
447     425 +
448     426 +
449     427 +
450     428 +
451     429 +
452     430 +
453     431 +
454     432 +
455     433 +
456     434 +
457     435 +
458     436 +
459     437 +
460     438 +
461     439 +
462     440 +
463     441 +
464     442 +
465     443 +
466     444 +
467     445 +
468     446 +
469     447 +
470     448 +
471     449 +
472     450 +
473     451 +
474     452 +
475     453 +
476     454 +
477     455 +
478     456 +
479     457 +
480     458 +
481     459 +
482     460 +
483     461 +
484     462 +
485     463 +
486     464 +
487     465 +
488     466 +
489     467 +
490     468 +
491     469 +
492     470 +
493     471 +
494     472 +
495     473 +
496     474 +
497     475 +
498     476 +
499     477 +
500     478 +
501     479 +
502     480 +
503     481 +
504     482 +
505     483 +
506     484 +
507     485 +
508     486 +
509     487 +
510     488 +
511     489 +
512     490 +
513     491 +
514     492 +
515     493 +
516     494 +
517     495 +
518     496 +
519     497 +
520     498 +
521     499 +
522     500 +
523     501 +
524     502 +
525     503 +
526     504 +
527     505 +
528     506 +
529     507 +
530     508 +
531     509 +
532     510 +
533     511 +
534     512 +
535     513 +
536     514 +
537     515 +
538     516 +
539     517 +
540     518 +
541     519 +
542     520 +
543     521 +
544     522 +
545     523 +
546     524 +
547     525 +
548     526 +
549     527 +
550     528 +
551     529 +
552     530 +
553     531 +
554     532 +
555     533 +
556     534 +
557     535 +
558     536 +
559     537 +
560     538 +
561     539 +
562     540 +
563     541 +
564     542 +
565     543 +
566     544 +
567     545 +
568     546 +
569     547 +
570     548 +
571     549 +
572     550 +
573     551 +
574     552 +
575     553 +
576     554 +
577     555 +
578     556 +
579     557 +
580     558 +
581     559 +
582     560 +
583     561 +
584     562 +
585     563 +
586     564 +
587     565 +
588     566 +
589     567 +
590     568 +
591     569 +
592     570 +
593     571 +
594     572 +
595     573 +
596     574 +
597     575 +
598     576 +
599     577 +
600     578 +
601     579 +
602     580 +
603     581 +
604     582 +
605     583 +
606     584 +
607     585 +
608     586 +
609     587 +
610     588 +
611     589 +
612     590 +
613     591 +
614     592 +
615     593 +
616     594 +
617     595 +
618     596 +
619     597 +
620     598 +
621     599 +
622     600 +
623     601 +
624     602 +
625     603 +
626     604 +
627     605 +
628     606 +
629     607 +
630     608 +
631     609 +
632     610 +
633     611 +
634     612 +
635     613 +
636     614 +
637     615 +
638     616 +
639     617 +
640     618 +
641     619 +
642     620 +
643     621 +
644     622 +
645     623 +
646     624 +
647     625 +
648     626 +
649     627 +
650     628 +
651     629 +
652     630 +
653     631 +
654     632 +
655     633 +
656     634 +
657     635 +
658     636 +
659     637 +
660     638 +
661     639 +
662     640 +
663     641 +
664     642 +
665     643 +
666     644 +
667     645 +
668     646 +
669     647 +
670     648 +
671     649 +
672     650 +
673     651 +
674     652 +
675     653 +
676     654 +
677     655 +
678     656 +
679     657 +
680     658 +
681     659 +
682     660 +
683     661 +
684     662 +
685     663 +
686     664 +
687     665 +
688     666 +
689     667 +
690     668 +
691     669 +
692     670 +
693     671 +
694     672 +
695     673 +
696     674 +
697     675 +
698     676 +
699     677 +
700     678 +
701     679 +
702     680 +
703     681 +
704     682 +
705     683 +
706     684 +
707     685 +
708     686 +
709     687 +
710     688 +
711     689 +
712     690 +
713     691 +
714     692 +
715     693 +
716     694 +
717     695 +
718     696 +
719     697 +
720     698 +
721     699 +
722     700 +
723     701 +
724     702 +
725     703 +
726     704 +
727     705 +
728     706 +
729     707 +
730     708 +
731     709 +
732     710 +
733     711 +
734     712 +
735     713 +
736     714 +
737     715 +
738     716 +
739     717 +
740     718 +
741     719 +
742     720 +
743     721 +
744     722 +
745     723 +
746     724 +
747     725 +
748     726 +
749     727 +
750     728 +
751     729 +
752     730 +
753     731 +
754     732 +
755     733 +
756     734 +
757     735 +
758     736 +
759     737 +
760     738 +
761     739 +
762     740 +
763     741 +
764     742 +
765     743 +
766     744 +
767     745 +
768     746 +
769     747 +
770     748 +
771     749 +
772     750 +
773     751 +
774     752 +
775     753 +
776     754 +
777     755 +
778     756 +
779     757 +
780     758 +
781     759 +
782     760 +
783     761 +
784     762 +
785     763 +
786     764 +
787     765 +
788     766 +
789     767 +
790     768 +
791     769 +
792     770 +
793     771 +
794     772 +
795     773 +
796     774 +
797     775 +
798     776 +
799     777 +
800     778 +
801     779 +
802     780 +
803     781 +
804     782 +
805     783 +
806     784 +
807     785 +
808     786 +
809     787 +
810     788 +
811     789 +
812     790 +
813     791 +
814     792 +
815     793 +
816     794 +
817     795 +
818     796 +
819     797 +
820     798 +
821     799 +
822     800 +
823     801 +
824     802 +
825     803 +
826     804 +
827     805 +
828     806 +
829     807 +
830     808 +
831     809 +
832     810 +
833     811 +
834     812 +
835     813 +
836     814 +
837     815 +
838     816 +
839     817 +
840     818 +
841     819 +
842     820 +
843     821 +
844     822 +
845     823 +
846     824 +
847     825 +
848     826 +
849     827 +
850     828 +
851     829 +
852     830 +
853     831 +
854     832 +
855     833 +
856     834 +
857     835 +
858     836 +
859     837 +
860     838 +
861     839 +
862     840 +
863     841 +
864     842 +
865     843 +
866     844 +
867     845 +
868     846 +
869     847 +
870     848 +
871     849 +
872     850 +
873     851 +
874     852 +
875     853 +
876     854 +
877     855 +
878     856 +
879     857 +
880     858 +
881     859 +
882     860 +
883     861 +
884     862 +
885     863 +
886     864 +
887     865 +
888     866 +
889     867 +
890     868 +
891     869 +
892     870 +
893     871 +
894     872 +
895     873 +
896     874 +
897     875 +
898     876 +
899     877 +
900     878 +
901     879 +
902     880 +
903     881 +
904     882 +
905     883 +
906     884 +
907     885 +
908     886 +
909     887 +
910     888 +
911     889 +
912     890 +
913     891 +
914     892 +
915     893 +
916     894 +
917     895 +
918     896 +
919     897 +
920     898 +
921     899 +
922     900 +
923     901 +
924     902 +
925     903 +
926     904 +
927     905 +
928     906 +
929     907 +
930     908 +
931     909 +
932     910 +
933     911 +
934     912 +
935     913 +
936     914 +
937     915 +
938     916 +
939     917 +
940     918 +
941     919 +
942     920 +
943     921 +
944     922 +
945     923 +
946     924 +
947     925 +
948     926 +
949     927 +
950     928 +
951     929 +
952     930 +
953     931 +
954     932 +
955     933 +
956     934 +
957     935 +
958     936 +
959     937 +
960     938 +
961     939 +
962     940 +
963     941 +
964     942 +
965     943 +
966     944 +
967     945 +
968     946 +
969     947 +
970     948 +
971     949 +
972     950 +
973     951 +
974     952 +
975     953 +
976     954 +
977     955 +
978     956 +
979     957 +
980     958 +
981     959 +
982     960 +
983     961 +
984     962 +
985     963 +
986     964 +
987     965 +
988     966 +
989     967 +
990     968 +
991     969 +
992     970 +
993     971 +
994     972 +
995     973 +
996     974 +
997     975 +
998     976 +
999     977 +
1000    978 +
1001    979 +
1002    980 +
1003    981 +
1004    982 +
1005    983 +
1006    984 +
1007    985 +
1008    986 +
1009    987 +
1010    988 +
1011    989 +
1012    990 +
1013    991 +
1014    992 +
1015    993 +
1016    994 +
1017    995 +
1018    996 +
1019    997 +
1020    998 +
1021    999 +
1022    1000 +
1023    1001 +
1024    1002 +
1025    1003 +
1026    1004 +
1027    1005 +
1028    1006 +
1029    1007 +
1030    1008 +
1031    1009 +
1032    1010 +
1033    1011 +
1034    1012 +
1035    1013 +
1036    1014 +
1037    1015 +
1038    1016 +
1039    1017 +
1040    1018 +
1041    1019 +
1042    1020 +
1043    1021 +
1044    1022 +
1045    1023 +
1046    1024 +
1047    1025 +
1048    1026 +
1049    1027 +
1050    1028 +
1051    1029 +
1052    1030 +
1053    1031 +
1054    1032 +
1055    1033 +
1056    1034 +
1057    1035 +
1058    1036 +
1059    1037 +
1060    1038 +
1061    1039 +
1062    1040 +
1063    1041 +
1064    1042 +
1065    1043 +
1066    1044 +
1067    1045 +
1068    1046 +
1069    1047 +
1070    1048 +
1071    1049 +
1072    1050 +
1073    1051 +
1074    1052 +
1075    1053 +
1076    1054 +
1077    1055 +
1078    1056 +
1079    1057 +
1080    1058 +
1081    1059 +
1082    1060 +
1083    1061 +
1084    1062 +
1085    1063 +
1086    1064 +
1087    1065 +
1088    1066 +
1089    1067 +
1090    1068 +
1091    1069 +
1092    1070 +
1093    1071 +
1094    1072 +
1095    1073 +
1096    1074 +
1097    1075 +
1098    1076 +
1099    1077 +
1100    1078 +
1101    1079 +
1102    1080 +
1103    1081 +
1104    1082 +
1105    1083 +
1106    1084 +
1107    1085 +
1108    1086 +
1109    1087 +
1110    1088 +
1111    1089 +
1112    1090 +
1113    1091 +
1114    1092 +
1115    1093 +
1116    1094 +
1117    1095 +
1118    1096 +
1119    1097 +
1120    1098 +
1121    1099 +
1122    1100 +
1123    1101 +
1124    1102 +
1125    1103 +
1126    1104 +
1127    1105 +
1128    1106 +
1129    1107 +
1130    1108 +
1131    1109 +
1132    1110 +
1133    1111 +
1134    1112 +
1135    1113 +
1136    1114 +
1137    1115 +
1138    1116 +
1139    1117 +
1140    1118 +
1141    1119 +
1142    1120 +
1143    1121 +
1144    1122 +
1145    1123 +
1146    1124 +
1147    1125 +
1148    1126 +
1149    1127 +
1150    1128 +
1151    1129 +
1152    1130 +
1153    1131 +
1154    1132 +
1155    1133 +
1156    1134 +
1157    1135 +
1158    1136 +
1159    1137 +
1160    1138 +
1161    1139 +
1162    1140 +
1163    1141 +
1164    1142 +
1165    1143 +
1166    1144 +
1167    1145 +
1168    1146 +
1169    1147 +
1170    1148 +
1171    1149 +
1172    1150 +
1173    1151 +
1174    1152 +
1175    1153 +
1176    1154 +
1177    1155 +
1178    1156 +
1179    1157 +
1180    1158 +
1181    1159 +
1182    1160 +
1183    1161 +
1184    1162 +
1185    1163 +
1186    1164 +
1187    1165 +
1188    1166 +
1189    1167 +
1190    1168 +
1191    1169 +
1192    1170 +
1193    1171 +
1194    1172 +
1195    1173 +
1196    1174 +
1197    1175 +
1198    1176 +
1199    1177 +
1200    1178 +
1201    1179 +
1202    1180 +
1203    1181 +
1204    1182 +
1205    1183 +
1206    1184 +
1207    1185 +
1208    1186 +
1209    1187 +
1210    1188 +
1211    1189 +
1212    1190 +
1213    1191 +
1214    1192 +
1215    1193 +
1216    1194 +
1217    1195 +
1218    1196 +
1219    1197 +
1220    1198 +
1221    1199 +
1222    1200 +
12
```

```

23      27      exception.printStackTrace();
26      30      }
27      31
28      32      public String[][] listagem(List<Integer> idsPessoas) throws
NegocioException {
29      -      int numCols = 10;
30      -      String[][] listaRetorno = null;
31      33      +      final int numCols = 10;
32      34      try {
33      -      PessoaDAO pessoaDAO = new PessoaDAO();
34      -
35      35      List<PessoaDTO> lista = pessoaDAO.listarTodos();
36      -      listaRetorno = new String[lista.size()][numCols];
37      36      +      String[][] listaRetorno = new String[lista.size()]
[numCols];
38      37
39      38      for (int i = 0; i < lista.size(); i++) {
40      39      PessoaDTO pessoa = lista.get(i);
41      49      50      listaRetorno[i][8] = "ALTER";
42      50      51      listaRetorno[i][9] = "DEL";
43      51      52      }
44      53      +
45      54      +      return listaRetorno;
46      52      55      } catch(Exception exception) {
47      53      56      throw new NegocioException(exception.getMessage());
48      54      57      }
49      55      -      return listaRetorno;
50      56      58      }
51      57      59
52      58      60      public boolean validaNome(String nome) throws ValidacaoException {
53      59      61      boolean ehValido = true;
54      60      -      if (nome == null || nome.equals("")) {
55      61      62      +      if (nome == null || nome.isEmpty()) {
56      62      63      ehValido = false;
57      63      -      throw new ValidacaoException("Campo nome é obrigatório!");
58      64      +      throw new ValidacaoException("Campo nome é obrigatório!");
59      65      } else if (nome.length() > 30) {
60      64      66      ehValido = false;
61      65      -      throw new ValidacaoException("Campo nome comporta no máximo
30 chars!");
62      67      +      throw new ValidacaoException("Campo nome comporta no máximo
30 chars!");
63      68      }
64      69      return ehValido;
65      70      }
66      71
67      72      public boolean validaCpf(String cpf) throws ValidacaoException {
68      73      boolean ehValido = true;
69      72      -      if (cpf == null || cpf.equals("")) {
70      74      +      if (cpf == null || cpf.isEmpty()) {
71      73      75      ehValido = false;
72      74      -      throw new ValidacaoException("Campo CPF é obrigatório!");

```

```

76 +         throw new ValidacaoException("Campo CPF é obrigatório!");
75 77         } else if (cpf.length() != 11) {
76 78             ehValido = false;
77 -         throw new ValidacaoException("Campo CPF deve ter 11
           dígitos!");
79 +         throw new ValidacaoException("Campo CPF deve ter 11
           dígitos!");
78 80         } else {
79 81             char[] digitos = cpf.toCharArray();
80 82             for (char digito : digitos) {
81 83                 if (!Character.isDigit(digito)) {
82 84                     ehValido = false;
83 -                     throw new ValidacaoException("Campo CPF é
           somente numérico!");
85 +                     throw new ValidacaoException("Campo CPF é
           somente numérico!");
84 86                 }
85 87             }
86 88         }
89 91
90 92         public boolean validaEndereco(EnderecoDTO enderecoDTO) throws
ValidacaoException {
91 93             boolean ehValido = true;
92 -             if (enderecoDTO.getLogradouro() == null ||
           enderecoDTO.getLogradouro().equals("")) {
94 +             if (enderecoDTO.getLogradouro() == null ||
           enderecoDTO.getLogradouro().isEmpty()) {
93 95                 ehValido = false;
94 -                 throw new ValidacaoException("Campo Logradouro é
           obrigatório!");
95 -             } else if (enderecoDTO.getBairro() == null ||
           enderecoDTO.getBairro().equals("")) {
96 +                 throw new ValidacaoException("Campo Logradouro é
           obrigatório!");
97 +             } else if (enderecoDTO.getBairro() == null ||
           enderecoDTO.getBairro().isEmpty()) {
96 98                 ehValido = false;
97 -                 throw new ValidacaoException("Bairro Obrigatorio");
98 -             } else if (enderecoDTO.getNumero() == null ||
           enderecoDTO.getNumero().equals(0)) {
99 +                 throw new ValidacaoException("Bairro Obrigatório");
100 +             } else if (enderecoDTO.getNumero() == null ||
           enderecoDTO.getNumero() == 0) {
99 101                 ehValido = false;
100 -                 throw new ValidacaoException("Numero Obrigatorio");
101 -             } else if (enderecoDTO.getCep() == null ||
           enderecoDTO.getCep().equals(0)) {
102 +                 throw new ValidacaoException("Número Obrigatório");
103 +             } else if (enderecoDTO.getCep() == null || enderecoDTO.getCep() ==
           0) {
102 104                 ehValido = false;
103 -                 throw new ValidacaoException("CEP Obrigatorio");

```

```

105 +         throw new ValidacaoException("CEP Obrigatório");
104 106     }
105 106 -
106 106 -
107 107 +
107 108     return ehValido;
108 109     }
109 110
110 111     public boolean validaDtNasc(String dtNasc) throws ValidacaoException {
111 112         boolean ehValido = true;
112 112 -         if (dtNasc == null || dtNasc.equals("")) {
113 113 +         if (dtNasc == null || dtNasc.isEmpty()) {
113 114             ehValido = false;
114 114 -         throw new ValidacaoException("Campo Dt. Nasc. é
obrigatório!");
115 115 +         throw new ValidacaoException("Campo Dt. Nasc. é
obrigatório!");
115 116     } else {
116 116 -         ehValido = false;
117 117         try {
118 118             dateFormat.parse(dtNasc);
119 119         } catch (ParseException e) {
120 120 -         throw new ValidacaoException("Formato inválido de
data!");
120 120 +         ehValido = false;
121 121 +         throw new ValidacaoException("Formato inválido de
data!");
121 122     }
122 123     }
123 124     return ehValido;
124 125     }
125 126
126 127     public String[][] listaConsulta(String nome, Long cpf, char sexo, String
orderBy) throws NegocioException {
127 127 -         int numCols = 6;
128 128 -         String[][] listaRetorno = null;
128 128 +         final int numCols = 6;
129 129         try {
130 130 -             PessoaDAO pessoaDAO = new PessoaDAO();
131 131 -
132 130             List<PessoaDTO> lista = pessoaDAO.filtrarPessoa(nome, cpf,
String.valueOf(sexo), orderBy);
133 131 -             listaRetorno = new String[lista.size()][numCols];
131 131 +             String[][] listaRetorno = new String[lista.size()]
[numCols];
134 132
135 133             for (int i = 0; i < lista.size(); i++) {
136 134                 PessoaDTO pessoa = lista.get(i);
141 139                 listaRetorno[i][4] = pessoa.getSexo() == 'M' ?
"Masculino" : "Feminino";
142 140                 listaRetorno[i][5] =
dateFormat.format(pessoa.getDtNascimento());

```

```

143      141      }
      142      +
      143      +          return listaRetorno;
144      144      } catch(Exception exception) {
145      145      throw new NegocioException(exception.getMessage());
146      146      }
147      147      -          return listaRetorno;
148      147      }
149      148
150      149      public void removePessoa(Integer idPessoa, Integer idEndereco) throws
NegocioException {
151      150      try {
152      151      -          PessoaDAO pessoaDAO = new PessoaDAO();
153      151      pessoaDAO.deletar(idPessoa);
154      152      pessoaDAO.deletarEndereco(idEndereco);
155      153      } catch(Exception exception) {
159      157
160      158      public void removeTudo() throws NegocioException {
161      159      try {
162      161      -          PessoaDAO pessoaDAO = new PessoaDAO();
163      160      pessoaDAO.deletarTudo();
164      161      -
165      161      } catch(Exception e) {
166      162      throw new NegocioException(e.getMessage());
167      163      }
168      164      }
169      165
170      166      public PessoaDTO buscaPorId(Integer idPessoa) throws NegocioException{
171      166      -          PessoaDTO pessoaDTO = null;
172      167      -
173      167      try {
174      167      -          PessoaDAO pessoaDAO = new PessoaDAO();
175      167      -          pessoaDTO = pessoaDAO.buscarPorId(idPessoa);
176      168      +          return pessoaDAO.buscarPorId(idPessoa);
177      169      } catch (Exception e) {
178      170      throw new NegocioException(e.getMessage(), e);
179      171      -
180      171      }
181      172      -          return pessoaDTO;
182      172      }
183      173      }

```

▼ 28 ■■■ src/br/sistema/crud/jdbc/bo/UfBO.java 

```

...      ...      @@ -1,28 +1,24 @@
1      1      package br.sistema.crud.jdbc.bo;
2      2
3      3      - import java.util.ArrayList;
4      3      import java.util.List;
5      4

```

```
6      5      import br.sistema.crud.jdbc.dao.UfDAO;
7      - import br.sistema.crud.jdbc.dto.UfDTO;
8      6      import br.sistema.crud.jdbc.exception.NegocioException;
9      7      + import br.sistema.crud.jdbc.model.Uf;
10     8
11     9      public class UfBO {
12    10
13    -      public List<UfDTO> listaUfs() throws NegocioException{
14    -          List<UfDTO> lista = null;
15    -
16    -      try{
17    -          UfDAO ufDAO = new UfDAO();
18    -          lista = ufDAO.listaEstado();
19    -
20    -      }catch (Exception e){
21    -          throw new NegocioException(e.getMessage(),e);
22    11      +      private UfDAO ufDAO;
23    12      +
24    13      +      public UfBO() {
25    14      +          this.ufDAO = new UfDAO();
26    15      +      }
27    16      +
28    17      +      public List<Uf> listarUfs() throws NegocioException {
29    18      +          try {
30    19      +              return ufDAO.listarUfs();
31    20      +          } catch (Exception e) {
32    21      +              throw new NegocioException(e.getMessage(), e);
33    22      +          }
34    23      -          return lista;
35    24      -
36    25      -
37    26      -      }
38    27      -
39    28      -      }
```

0 comments on commit 807ee02

Please [sign in](#) to comment.