

[| 자유 게시판 | >](#)

## 명령어 처리 함수 개선판


**odask** 챗봇 마스터 🤖 1:1 채팅

2020.03.30. 16:24 조회 102

댓글 2 URL 복사 ⋮

### 공유할 소스 용도

=&gt; (ex: 가위바위보 게임)

### 소스 코드

=&gt;

```
const PERMISSION = ['admin', 'manager', 'moderator', 'member', 'guest']

// msg = '!테스트 홍길동 발사'
const _msg = msg.split(' ') // ['!테스트', '홍길동', '발사']
const $msg = _msg[0] // !테스트
const date = new Date()

const msgHeader = '!' // 메시지 구분자 (!메세지 에서 =>!<=)

let pindex = PERMISSION.indexOf(/*유저별 권한*/)

function commandExec(localCommand) {
  let ret = []
  let localCommandTemp = {}

  function chk() {
    if (!this.enabled) return false
    if (pindex > PERMISSION.indexOf(this.permission)) return false
    return true
  }

  function vrf() {
    //타입체크
    if (typeof this !== 'object' && !$msg) return false

    if (this === true) return true
    let ret = {}
    if (typeof this === 'function') return this()
    if (typeof this === 'object') {
      if (this.key) {
        const header = $msg.split(msgHeader, 1)[0]
        if (header === '') {
          ret.mode = 'default'
        } else if (header === 'd') {
          ret.mode = 'debug'
        } else return false
        const key = $msg.substring(header.length + 1)
        if (this.key.indexOf(key) === -1) return false
        ret.key = key
        ret.keyIndex = this.key.indexOf(key)
      }
      if (this.length) {
        let pass = true
        if (this.length[0].length === 1) {
          if (_msg.length !== this.length[0][0]) pass = false
        } else if (!(_msg.length >= this.length[0][0] && _msg.length <= this.length[0][1]))
pass = false

        if (!pass) {
          if (this.length.length === 2) this.length[1] === true ? replier.reply('잘못된 입
력 형식입니다.') : replier.reply(this.length[1])
        }
      }
    }
  }
}
```

```

        replier.reply(this.length[1])
        return false
    }
}
if (this.format) {
    let pass = true
    for (let i in this.format[0]) {
        if (_msg[+i + 1] && this.format[0][i] && !this.format[0][i].test(_msg[+i + 1]))
pass = false
    }
    if (!pass) {
        if (this.format.length == 2) this.format[1] === true ? replier.reply('잘못된 입
력 형식입니다.') : replier.reply(this.format[1])
        return false
    }
}
//pass arg to check()
if (this.check && !this.check(ret)) return false
return ret
}
}

function checker(lcmd, lcmdtemp) {
    this.forEach((e) => {
        lcmd[e.id] = lcmd[e.id] || {}
        lcmdtemp[e.id] = {}
        for (let c in e.setting) {
            lcmdtemp[e.id][c] = typeof lcmd[e.id][c] === 'undefined' ? e.setting[c] : lcmd[e.id]
[c]

            lcmdtemp[e.id].name = e.name
        }
        if (e.cate == 'group') {
            if (chk.call(lcmd[e.id])) {
                checker.call(e.main, lcmd, lcmdtemp);
            }
        } else if (e.cate == 'command') {
            if (chk.call(lcmd[e.id])) {
                let verified = vrf.call(e.main.verify)
                verified && ret.push([e.main.execute, verified])
            }
        }
    })
}
checker.call(this, localCommand, localCommandTemp)
return [localCommandTemp, ret]
}

function commandListFormat(localCommand, args) {
    let ret = []

    function chk() {
        if (!this.enabled) return false
        if (pindex > PERMISSION.indexOf(this.permission)) return false
        return true
    }

    function checker(lcmd) {
        this.forEach((e) => {
            lcmd[e.id] = lcmd[e.id] || e.setting
            if (e.cate == 'group') {
                args == 'printAll' && ret.push('\n분류: ' + e.name + ' (' + lcmd[e.id].permission +
 '/' + (lcmd[e.id].enabled ? 'enabled' : 'disabled') + ')\n| ' + e.description)
                if (chk.call(lcmd[e.id]) && !e.hide) {
                    args == 'print' && ret.push('\n분류: ' + e.name + ' (' + lcmd[e.id].permission +
 ')\n| ' + e.description)
                    checker.call(e.main, lcmd);
                }
            } else if (e.cate == 'command') {
                args == 'printAll' && ret.push(' > ' + e.name + ' (' + lcmd[e.id].permission + '/'
(1 - 15 - 17 - 18 - 19 - 20 - 21 - 22 - 23 - 24 - 25 - 26 - 27 - 28 - 29 - 30 - 31 - 32 - 33 - 34 - 35 - 36 - 37 - 38 - 39 - 40 - 41 - 42 - 43 - 44 - 45 - 46 - 47 - 48 - 49 - 50 - 51 - 52 - 53 - 54 - 55 - 56 - 57 - 58 - 59 - 60 - 61 - 62 - 63 - 64 - 65 - 66 - 67 - 68 - 69 - 70 - 71 - 72 - 73 - 74 - 75 - 76 - 77 - 78 - 79 - 80 - 81 - 82 - 83 - 84 - 85 - 86 - 87 - 88 - 89 - 90 - 91 - 92 - 93 - 94 - 95 - 96 - 97 - 98 - 99 - 100 - 101 - 102 - 103 - 104 - 105 - 106 - 107 - 108 - 109 - 110 - 111 - 112 - 113 - 114 - 115 - 116 - 117 - 118 - 119 - 120 - 121 - 122 - 123 - 124 - 125 - 126 - 127 - 128 - 129 - 130 - 131 - 132 - 133 - 134 - 135 - 136 - 137 - 138 - 139 - 140 - 141 - 142 - 143 - 144 - 145 - 146 - 147 - 148 - 149 - 150 - 151 - 152 - 153 - 154 - 155 - 156 - 157 - 158 - 159 - 160 - 161 - 162 - 163 - 164 - 165 - 166 - 167 - 168 - 169 - 170 - 171 - 172 - 173 - 174 - 175 - 176 - 177 - 178 - 179 - 180 - 181 - 182 - 183 - 184 - 185 - 186 - 187 - 188 - 189 - 190 - 191 - 192 - 193 - 194 - 195 - 196 - 197 - 198 - 199 - 200 - 201 - 202 - 203 - 204 - 205 - 206 - 207 - 208 - 209 - 210 - 211 - 212 - 213 - 214 - 215 - 216 - 217 - 218 - 219 - 220 - 221 - 222 - 223 - 224 - 225 - 226 - 227 - 228 - 229 - 230 - 231 - 232 - 233 - 234 - 235 - 236 - 237 - 238 - 239 - 240 - 241 - 242 - 243 - 244 - 245 - 246 - 247 - 248 - 249 - 250 - 251 - 252 - 253 - 254 - 255 - 256 - 257 - 258 - 259 - 260 - 261 - 262 - 263 - 264 - 265 - 266 - 267 - 268 - 269 - 270 - 271 - 272 - 273 - 274 - 275 - 276 - 277 - 278 - 279 - 280 - 281 - 282 - 283 - 284 - 285 - 286 - 287 - 288 - 289 - 290 - 291 - 292 - 293 - 294 - 295 - 296 - 297 - 298 - 299 - 300 - 301 - 302 - 303 - 304 - 305 - 306 - 307 - 308 - 309 - 310 - 311 - 312 - 313 - 314 - 315 - 316 - 317 - 318 - 319 - 320 - 321 - 322 - 323 - 324 - 325 - 326 - 327 - 328 - 329 - 330 - 331 - 332 - 333 - 334 - 335 - 336 - 337 - 338 - 339 - 340 - 341 - 342 - 343 - 344 - 345 - 346 - 347 - 348 - 349 - 350 - 351 - 352 - 353 - 354 - 355 - 356 - 357 - 358 - 359 - 360 - 361 - 362 - 363 - 364 - 365 - 366 - 367 - 368 - 369 - 370 - 371 - 372 - 373 - 374 - 375 - 376 - 377 - 378 - 379 - 380 - 381 - 382 - 383 - 384 - 385 - 386 - 387 - 388 - 389 - 390 - 391 - 392 - 393 - 394 - 395 - 396 - 397 - 398 - 399 - 400 - 401 - 402 - 403 - 404 - 405 - 406 - 407 - 408 - 409 - 410 - 411 - 412 - 413 - 414 - 415 - 416 - 417 - 418 - 419 - 420 - 421 - 422 - 423 - 424 - 425 - 426 - 427 - 428 - 429 - 430 - 431 - 432 - 433 - 434 - 435 - 436 - 437 - 438 - 439 - 440 - 441 - 442 - 443 - 444 - 445 - 446 - 447 - 448 - 449 - 450 - 451 - 452 - 453 - 454 - 455 - 456 - 457 - 458 - 459 - 460 - 461 - 462 - 463 - 464 - 465 - 466 - 467 - 468 - 469 - 470 - 471 - 472 - 473 - 474 - 475 - 476 - 477 - 478 - 479 - 480 - 481 - 482 - 483 - 484 - 485 - 486 - 487 - 488 - 489 - 490 - 491 - 492 - 493 - 494 - 495 - 496 - 497 - 498 - 499 - 500 - 501 - 502 - 503 - 504 - 505 - 506 - 507 - 508 - 509 - 510 - 511 - 512 - 513 - 514 - 515 - 516 - 517 - 518 - 519 - 520 - 521 - 522 - 523 - 524 - 525 - 526 - 527 - 528 - 529 - 530 - 531 - 532 - 533 - 534 - 535 - 536 - 537 - 538 - 539 - 540 - 541 - 542 - 543 - 544 - 545 - 546 - 547 - 548 - 549 - 550 - 551 - 552 - 553 - 554 - 555 - 556 - 557 - 558 - 559 - 560 - 561 - 562 - 563 - 564 - 565 - 566 - 567 - 568 - 569 - 570 - 571 - 572 - 573 - 574 - 575 - 576 - 577 - 578 - 579 - 580 - 581 - 582 - 583 - 584 - 585 - 586 - 587 - 588 - 589 - 590 - 591 - 592 - 593 - 594 - 595 - 596 - 597 - 598 - 599 - 600 - 601 - 602 - 603 - 604 - 605 - 606 - 607 - 608 - 609 - 610 - 611 - 612 - 613 - 614 - 615 - 616 - 617 - 618 - 619 - 620 - 621 - 622 - 623 - 624 - 625 - 626 - 627 - 628 - 629 - 630 - 631 - 632 - 633 - 634 - 635 - 636 - 637 - 638 - 639 - 640 - 641 - 642 - 643 - 644 - 645 - 646 - 647 - 648 - 649 - 650 - 651 - 652 - 653 - 654 - 655 - 656 - 657 - 658 - 659 - 660 - 661 - 662 - 663 - 664 - 665 - 666 - 667 - 668 - 669 - 670 - 671 - 672 - 673 - 674 - 675 - 676 - 677 - 678 - 679 - 680 - 681 - 682 - 683 - 684 - 685 - 686 - 687 - 688 - 689 - 690 - 691 - 692 - 693 - 694 - 695 - 696 - 697 - 698 - 699 - 700 - 701 - 702 - 703 - 704 - 705 - 706 - 707 - 708 - 709 - 710 - 711 - 712 - 713 - 714 - 715 - 716 - 717 - 718 - 719 - 720 - 721 - 722 - 723 - 724 - 725 - 726 - 727 - 728 - 729 - 730 - 731 - 732 - 733 - 734 - 735 - 736 - 737 - 738 - 739 - 740 - 741 - 742 - 743 - 744 - 745 - 746 - 747 - 748 - 749 - 750 - 751 - 752 - 753 - 754 - 755 - 756 - 757 - 758 - 759 - 760 - 761 - 762 - 763 - 764 - 765 - 766 - 767 - 768 - 769 - 770 - 771 - 772 - 773 - 774 - 775 - 776 - 777 - 778 - 779 - 780 - 781 - 782 - 783 - 784 - 785 - 786 - 787 - 788 - 789 - 790 - 791 - 792 - 793 - 794 - 795 - 796 - 797 - 798 - 799 - 800 - 801 - 802 - 803 - 804 - 805 - 806 - 807 - 808 - 809 - 810 - 811 - 812 - 813 - 814 - 815 - 816 - 817 - 818 - 819 - 820 - 821 - 822 - 823 - 824 - 825 - 826 - 827 - 828 - 829 - 830 - 831 - 832 - 833 - 834 - 835 - 836 - 837 - 838 - 839 - 840 - 841 - 842 - 843 - 844 - 845 - 846 - 847 - 848 - 849 - 850 - 851 - 852 - 853 - 854 - 855 - 856 - 857 - 858 - 859 - 860 - 861 - 862 - 863 - 864 - 865 - 866 - 867 - 868 - 869 - 870 - 871 - 872 - 873 - 874 - 875 - 876 - 877 - 878 - 879 - 880 - 881 - 882 - 883 - 884 - 885 - 886 - 887 - 888 - 889 - 890 - 891 - 892 - 893 - 894 - 895 - 896 - 897 - 898 - 899 - 900 - 901 - 902 - 903 - 904 - 905 - 906 - 907 - 908 - 909 - 910 - 911 - 912 - 913 - 914 - 915 - 916 - 917 - 918 - 919 - 920 - 921 - 922 - 923 - 924 - 925 - 926 - 927 - 928 - 929 - 930 - 931 - 932 - 933 - 934 - 935 - 936 - 937 - 938 - 939 - 940 - 941 - 942 - 943 - 944 - 945 - 946 - 947 - 948 - 949 - 950 - 951 - 952 - 953 - 954 - 955 - 956 - 957 - 958 - 959 - 960 - 961 - 962 - 963 - 964 - 965 - 966 - 967 - 968 - 969 - 970 - 971 - 972 - 973 - 974 - 975 - 976 - 977 - 978 - 979 - 980 - 981 - 982 - 983 - 984 - 985 - 986 - 987 - 988 - 989 - 990 - 991 - 992 - 993 - 994 - 995 - 996 - 997 - 998 - 999 - 1000 - 1001 - 1002 - 1003 - 1004 - 1005 - 1006 - 1007 - 1008 - 1009 - 1010 - 1011 - 1012 - 1013 - 1014 - 1015 - 1016 - 1017 - 1018 - 1019 - 1020 - 1021 - 1022 - 1023 - 1024 - 1025 - 1026 - 1027 - 1028 - 1029 - 1030 - 1031 - 1032 - 1033 - 1034 - 1035 - 1036 - 1037 - 1038 - 1039 - 1040 - 1041 - 1042 - 1043 - 1044 - 1045 - 1046 - 1047 - 1048 - 1049 - 1050 - 1051 - 1052 - 1053 - 1054 - 1055 - 1056 - 1057 - 1058 - 1059 - 1060 - 1061 - 1062 - 1063 - 1064 - 1065 - 1066 - 1067 - 1068 - 1069 - 1070 - 1071 - 1072 - 1073 - 1074 - 1075 - 1076 - 1077 - 1078 - 1079 - 1080 - 1081 - 1082 - 1083 - 1084 - 1085 - 1086 - 1087 - 1088 - 1089 - 1090 - 1091 - 1092 - 1093 - 1094 - 1095 - 1096 - 1097 - 1098 - 1099 - 1100 - 1101 - 1102 - 1103 - 1104 - 1105 - 1106 - 1107 - 1108 - 1109 - 1110 - 1111 - 1112 - 1113 - 1114 - 1115 - 1116 - 1117 - 1118 - 1119 - 1120 - 1121 - 1122 - 1123 - 1124 - 1125 - 1126 - 1127 - 1128 - 1129 - 1130 - 1131 - 1132 - 1133 - 1134 - 1135 - 1136 - 1137 - 1138 - 1139 - 1140 - 1141 - 1142 - 1143 - 1144 - 1145 - 1146 - 1147 - 1148 - 1149 - 1150 - 1151 - 1152 - 1153 - 1154 - 1155 - 1156 - 1157 - 1158 - 1159 - 1160 - 1161 - 1162 - 1163 - 1164 - 1165 - 1166 - 1167 - 1168 - 1169 - 1170 - 1171 - 1172 - 1173 - 1174 - 1175 - 1176 - 1177 - 1178 - 1179 - 1180 - 1181 - 1182 - 1183 - 1184 - 1185 - 1186 - 1187 - 1188 - 1189 - 1190 - 1191 - 1192 - 1193 - 1194 - 1195 - 1196 - 1197 - 1198 - 1199 - 1200 - 1201 - 1202 - 1203 - 1204 - 1205 - 1206 - 1207 - 1208 - 1209 - 1210 - 1211 - 1212 - 1213 - 1214 - 1215 - 1216 - 1217 - 1218 - 1219 - 1220 - 1221 - 1222 - 1223 - 1224 - 1225 - 1226 - 1227 - 1228 - 1229 - 1230 - 1231 - 1232 - 1233 - 1234 - 1235 - 1236 - 1237 - 1238 - 1239 - 1240 - 1241 - 1242 - 1243 - 1244 - 1245 - 1246 - 1247 - 1248 - 1249 - 1250 - 1251 - 1252 - 1253 - 1254 - 1255 - 1256 - 1257 - 1258 - 1259 - 1260 - 1261 - 1262 - 1263 - 1264 - 1265 - 1266 - 1267 - 1268 - 1269 - 1270 - 1271 - 1272 - 1273 - 1274 - 1275 - 1276 - 1277 - 1278 - 1279 - 1280 - 1281 - 1282 - 1283 - 1284 - 1285 - 1286 - 1287 - 1288 - 1289 - 1290 - 1291 - 1292 - 1293 - 1294 - 1295 - 1296 - 1297 - 1298 - 1299 - 1300 - 1301 - 1302 - 1303 - 1304 - 1305 - 1306 - 1307 - 1308 - 1309 - 1310 - 1311 - 1312 - 1313 - 1314 - 1315 - 1316 - 1317 - 1318 - 1319 - 1320 - 1321 - 1322 - 1323 - 1324 - 1325 - 1326 - 1327 - 1328 - 1329 - 1330 - 1331 - 1332 - 1333 - 1334 - 1335 - 1336 - 1337 - 1338 - 1339 - 1340 - 1341 - 1342 - 1343 - 1344 - 1345 - 1346 - 1347 - 1348 - 1349 - 1350 - 1351 - 1352 - 1353 - 1354 - 1355 - 1356 - 1357 - 1358 - 1359 - 1360 - 1361 - 1362 - 1363 - 1364 - 1365 - 1366 - 1367 - 1368 - 1369 - 1370 - 1371 - 1372 - 1373 - 1374 - 1375 - 1376 - 1377 - 1378 - 1379 - 1380 - 1381 - 1382 - 1383 - 1384 - 1385 - 1386 - 1387 - 1388 - 1389 - 1390 - 1391 - 1392 - 1393 - 1394 - 1395 - 1396 - 1397 - 1398 - 1399 - 1400 - 1401 - 1402 - 1403 - 1404 - 1405 - 1406 - 1407 - 1408 - 1409 - 1410 - 1411 - 1412 - 1413 - 1414 - 1415 - 1416 - 1417 - 1418 - 1419 - 1420 - 1421 - 1422 - 1423 - 1424 - 1425 - 1426 - 1427 - 1428 - 1429 - 1430 - 1431 - 1432 - 1433 - 1434 - 1435 - 1436 - 1437 - 1438 - 1439 - 1440 - 1441 - 1442 - 1443 - 1444 - 1445 - 1446 - 1447 - 1448 - 1449 - 1450 - 1451 - 1452 - 1453 - 1454 - 1455 - 1456 - 1457 - 1458 - 1459 - 1460 - 1461 - 1462 - 1463 - 1464 - 1465 - 1466 - 1467 - 1468 - 1469 - 1470 - 1471 - 1472 - 1473 - 1474 - 1475 - 1476 - 1477 - 1478 - 1479 - 1480 - 1481 - 1482 - 1483 - 1484 - 1485 - 1486 - 1487 - 1488 - 1489 - 1490 - 1491 - 1492 - 1493 - 1494 - 1495 - 1496 - 1497 - 1498 - 1499 - 1500 - 1501 - 1502 - 1503 - 1504 - 1505 - 1506 - 1507 - 1508 - 1509 - 1510 - 1511 - 1512 - 1513 - 1514 - 1515 - 1516 - 1517 - 1518 - 1519 - 1520 - 1521 - 1522 - 1523 - 1524 - 1525 - 1526 - 1527 - 1528 - 1529 - 1530 - 1531 - 1532 - 1533 - 1534 - 1535 - 1536 - 1537 - 1538 - 1539 - 1540 - 1541 - 1542 - 1543 - 1544 - 1545 - 1546 - 1547 - 1548 - 1549 - 1550 - 1551 - 1552 - 1553 - 1554 - 1555 - 1556 - 1557 - 1558 - 1559 - 1560 - 1561 - 1562 - 1563 - 1564 - 1565 - 1566 - 1567 - 1568 - 1569 - 1570 - 1571 - 1572 - 1573 - 1574 - 1575 - 1576 - 1577 - 1578 - 1579 - 1580 - 1581 - 1582 - 1583 - 1584 - 1585 - 1586 - 1587 - 1588 - 1589 - 1590 - 1591 - 1592 - 1593 - 1594 - 1595 - 1596 - 1597 - 1598 - 1599 - 1600 - 1601 - 1602 - 1603 - 1604 - 1605 - 1606 - 1607 - 1608 - 1609 - 1610 - 1611 - 1612 - 1613 - 1614 - 1615 - 1616 - 1617 - 1618 - 1619 - 1620 - 1621 - 1622 - 1623 - 1624 - 1625 - 1626 - 1627 - 1628 - 1629 - 1630 - 1631 - 1632 - 1633 - 1634 - 1635 - 1636 - 1637 - 1638 - 1639 - 1640 - 1641 - 1642 - 1643 - 1644 - 1645 - 1646 - 1647 - 1648 - 1649 - 1650 - 1651 - 1652 - 1653 - 1654 - 1655 - 1656 - 1657 - 1658 - 1659 - 1660 - 1661 - 1662 - 1663 - 1664 - 1665 - 1666 - 1667 - 1668 - 1669 - 1670 - 1671 - 1672 - 1673 - 1674 - 1675 - 1676 - 1677 - 1678 - 1679 - 1680 - 1681 - 1682 - 1683 - 1684 - 1685 - 1686 - 1687 - 1688 - 1689 - 1690 - 1691 - 1692 - 1693 - 1694 - 1695 - 1696 - 1697 - 1698 - 1699 - 1700 - 1701 - 1702 - 1703 - 1704 - 1705 - 1706 - 1707 - 1708 - 1709 - 1710 - 1711 - 1712 - 1713 - 1714 - 1715 - 1716 - 1717 - 1718 - 1719 - 1720 - 1721 - 1722 - 1723 - 1724 - 1725 - 1726 - 1727 - 1728 - 1729 - 1730 - 1731 - 1732 - 1733 - 1734 - 1735 - 1736 - 1737 - 1738 - 1739 - 1740 - 1741 - 1742 - 1743 - 1744 - 1745 - 1746 - 1747 - 1748 - 1749 - 1750 - 1751 - 1752 - 1753 - 1754 - 1755 - 1756 - 1757 - 1758 - 1759 - 1760 - 1761 - 1762 - 1763 - 1764 - 1765 - 1766 - 1767 - 1768 - 1769 - 1770 - 1771 - 1772 - 1773 - 1774 - 1775 - 1776 - 1777 - 1778 - 1779 - 1780 - 1781 - 1782 - 1783 - 1784 - 1785 - 1786 - 1787 - 1788 - 1789 - 1790 - 1791 - 1792 - 1793 - 1794 - 1795 - 1796 - 1797 - 1798 - 1799 - 1800 - 1801 - 1802 - 1803 - 1804 - 1805 - 1806 - 1807 - 1808 - 1809 - 1810 - 1811 - 1812 - 1813 - 1814 - 1815 - 1816 - 1817 - 1818 - 1819 - 1820 - 1821 - 1822 - 1823 - 1824 - 1825 - 1826 - 1827 - 1828 - 1829 - 1830 - 1831 - 1832 - 1833 - 1834 - 1835 - 1836 - 1837 - 1838 - 1839 - 1840 - 1841 - 1842 - 1843 - 1844 - 1845 - 1846 - 1847 - 1848 - 1849 - 1850 - 1851 - 1852 - 1853 - 1854 - 1855 - 1856 - 1857 - 1858 - 1859 - 1860 - 1861 - 1862 - 1863 - 1864 - 1865 - 1866 - 1867 - 1868 - 1869 - 1870 - 1871 - 1872 - 1873 - 1874 - 1875 - 1876 - 1877 - 1878 - 1879 - 1880 - 1881 - 1882 - 1883 - 1884 - 1885 - 1886 - 1887 - 1888 - 1889 - 1890 - 1891 - 1892 - 1893 - 1894 - 1895 - 1896 - 1897 - 1898 - 1899 - 1900 - 1901 - 1902 - 1903 - 1904 - 1905 - 1906 - 1907 - 1908 - 1909 - 1910 - 1911 - 1912 - 1913 - 1914 - 1915 - 1916 - 1917 - 1918 - 1919 - 1920 - 1921 - 1922 - 1923 - 1924 - 1925 - 1926 - 1927 - 1928 - 1929 - 1930 - 1931 - 1932 - 1933 - 1934 - 1935 - 1936 - 1937 - 1938 - 1939 - 1940 - 1941 - 1942 - 1943 - 1944 - 1945 - 1946 - 1947 - 1948 - 1949 - 1950 - 1951 - 1952 - 1953 - 1954 - 1955 - 1956 - 1957 - 1958 - 1959 - 1960 - 1961 - 1962 - 1963 - 1964 - 1965 - 1966 - 1967 - 1968 - 1969 - 1970 - 1971 - 1972 - 1973 - 1974 - 1975 - 1976 - 1977 - 1978 - 1979 - 1980 - 1981 - 1982 - 1983 - 1984 - 1985 - 1986 - 1987 - 1988 - 1989 - 1990 - 1991 - 1992 - 1993 - 1994 - 1995 - 1996 - 1997 - 1998 - 1999 - 2000 - 2001 - 2002 - 2003 - 2004 - 2005 - 2006 - 2007 - 2008 - 2009 - 2010 - 2011 - 2012 - 2013 - 2014 - 2015 - 2016 - 2017 - 2018 - 2019 - 2020 - 2021 - 2022 - 2023 - 2024 - 2025 - 2026 - 2027 - 2028 - 2029 - 2030 - 2031 - 2032 - 2033 - 2034 - 2035 - 2036 - 2037 - 2038 - 2039 - 2040 - 2041 - 2042 - 2043 - 2044 - 2045 - 2046 - 2047 - 2048 - 2049 - 2050 - 2051 - 2052 - 2053 - 2054 - 2055 - 2056 - 2057 - 2058 - 2059 - 2060 - 2061 - 2062 - 2063 - 2064 - 2065 - 2066 - 2067 - 2068 - 2069 - 2070 - 2071 - 2072 - 2073 - 2074 - 2075 - 2076 - 2077 - 2078 - 2079 - 2080 - 2081 - 2082 - 2083 - 2084 - 2085 - 2086 - 2087 - 2088 - 2089 - 2090 - 2091 - 2092 - 2093 - 2094 - 2095 - 2096 - 2097 - 2098 - 2099 - 2100 - 2101 - 2102 - 2103 - 2104 - 2105 - 2106 - 2107 - 2108 - 2109 - 2110 - 2111 - 2112 - 2113 - 2114 - 2115 - 2116 - 2117 - 2118 - 2119 - 2120 - 2121 - 2122 - 2123 - 2124 - 2125 - 2126 - 2127 - 2128 - 2129 - 2130 - 2131 - 2132
```

```

+ (lcmd[e.id].enabled ? 'enabled' : 'disabled') + ')\n    | ' + e.description)
    if (chk.call(lcmd[e.id])) {
        args == 'print' && ret.push('    >' + e.name + ' (' + lcmd[e.id].permission + ')
\n    | ' + e.description)
    }
}
});
}
checker.call(this, localCommand);
(args == 'printAll' || args == 'print') && replier.reply('[[ 명령어 ]]$1$2'.format(blank, ret.join('\n')))
}

const commandList = [{
    cate: 'group',
    id: 1,
    name: '비활성',
    description: 'for test purpose',
    setting: {
        enabled: true,
        permission: 'guest'
    },
    main: [{
        cate: 'command',
        id: 1,
        name: '테스트',
        description: '!테스트',
        //hide: true,
        setting: {
            enabled: true,
            permission: 'guest'
        },
        main: {
            verify: {
                key: ['테스트'],
                length: [
                    [1]
                ],
                //format: [[/^(?!0.)\d+$/], /^(?!0.)\d+$/], '아!'],
            },
            execute: function () {
                replier.reply('asdf')
            }
        }
    },
    ],
    ],
}]

let executed = commandExec.call(commandList, 전역권한관리객체)
전역권한관리객체 = executed[0]
for (let cmd in executed[1]) {
    executed[1][cmd][0](executed[1][cmd][1])
}

```

## 개요

명령어 권한 확인부터 인식, 출력에 이르기까지 전과정을 통합

## 설명

사전 설정:

```
const PERMISSION = ['admin', 'manager', 'moderator', 'member', 'guest']
```

권한 그룹, 원하대로 마음대로 바꿔 사용해도 됨

```
let pindex = PERMISSION.indexOf(/*유저별 권한*/)
```

유저별 권한: PERMISSION의 key 중 하나(string), 알아서 권한 검사하는 함수 만들어서 실제 권한에 따라 할당

```
let executed = commandExec.call(commandList, 전역권한관리객체)
```

전역권한관리객체: 방별로 명령어 권한 다르게 설정 가능하게 해주는 DB, 그냥 빈 전역객체 하나 만들면 됨  
다만 리로드할때마다 초기화되는걸 원하지 않으면 알아서 세이브&로드 구현

function commandExec: 명령어 실행시 검사하는 함수

function commandListFormat: !도움말 등의 명령어 목록 출력시 텍스트 포맷해주는 함수

object(array) commandList: 명령어 본체

commandList의 구성:

cate: group 또는 cate: command로 이루어진 트리 구조

group은 명령어를 담는 폴더같은 역할로, 명령어를 분류해주고 상위 권한의 조정으로 전체 하위 권한에 영향을 줄 수 있음

각 원소의 프로퍼티:

cate: 'command', **command or group => 원소 종류 구분**

id: 1, **unique id => 이름이 바뀌거나 위치가 바뀌더라도 개별 설정한 권한에 영향을 미치지 않게 해주는 역할, 아무거나 적어도 되지만 다른 원소(group, command 모두 포함)와 겹치면 안됨**

name: '테스트', **도움말 출력 시 나오는 이름, 겹치지 않게 / 코드 볼때 이해 편하게**

description: '!테스트', **도움말 출력 시 나오는 설명 / 코드 볼때 이해 편하게**

//hide: true, **hide: true 로 설정 시 활성화된 명령어라도 일반 !도움말 시에 표시되지 않음**

setting: { **스크립트가 로드된 상태에서 수정 가능한 권한**

enabled: true, **해당 명령어가 활성화되어 있는지**

permission: 'guest' **해당 명령어의 권한**

},

만약 cate: group이라면 main은 group 또는 command 객체로 구성된 배열

main: [{...}, {...}, ...]

만약 cate: command라면

main: {

verify: {

만약 verify가 object가 아닌 Boolean true 라면 무조건 execute를 실행

만약 verify가 object가 아닌 function이라면 해당 함수의 리턴값이 truthy하면 execute를 실행

만약 verify가 object라면 object에 아래에 나열된 키가 존재한다면 우측의 주석에 따라 검사 (존재하지 않는다면 검사하지 않음), 하나라도 통과 실패한다면 return

key: ['테스트, 테에스트트'], **메세지가 반응하는 시작 키워드(구분자는 기본적으로 ! 이므로 이경우에는 !테스트 ~~ 또는 !테에스트트 ~~~ 포맷 허용**

length: [[3], '포맷 오류 1!'], **메세지의 단어 제한, [[x(y)](errmsg)] => x로 하면 단어수가 x인 메세지만 통과, x,y로 하면 단어가 x~y인 메세지만 통과, errmsg에 값을 넣으면 포맷 통과 실패시 메세지 출력 - true를 넣으면 기본 설정 오류메세지를 출력, string을 넣으면 해당 string을 출력**

format: [/정규식/, /정규식/], '포맷 오류 2!'] **두번째 단어부터 정규식을 검사 (예를 들어 이 예제에서는 메세지가 !테스트 A B 형태이니 format 배열의 index 0과 index 1은 각각 A와 B에 매칭, null등 falsy한 값을 넣으면 검사 스킵**

check: function(args) { **해당 function의 return 값이 truthy 하면 통과**

~~~~

}

},

execute: function (args) {

**check를 모두 통과했으면 최종적으로 실행될 구문**

}

**check(혹은 verify: function(){} 형식)와 execute 함수의 args 파라미터에 대하여**

**verify 과정에서 자동으로 생성되고 반환되는 객체를 받는다**

**verify.key 에서**

**메세지가 !테스트 처럼 일반적이면 args.mode = 'default', d!테스트 처럼 앞에 d가 붙으면 'debug'**

**메세지가 !테스트 라고 하면 args.key = '테스트', args.keyIndex = 0 ([!테스트, '테에스트트']에서 0번째 index이므로)**

**verify.check 또는 verify: function(){} 에도 args가 인자로 전달되므로 함수 내부에서 args를 조정하면 최종 return 값에 반영됨 이는 최종적으로 execute에 (넣는다면) 인자로 들어가 명령어 처리를 편하게 할 수 있음**

}

commandListFormat의 메소드:

호출: **commandListFormat.call(commandList, 전역권한관리객체, 'print'|'printAll')**

print: **자신이 실행 가능한 명령어만(hide 제외) 출력, printAll: 모든 명령어 출력**

**값이 return되는게 아닌 바로 reply되니까 알아서 바꿔쓰시던지 그냥쓰시던지**

참고: 명령어 순서 또는 이름이 바뀌더라도 고유 id만 같다면 방 별 개별설정된 권한은 보존됨

전역권한관리객체:

예시) 이름으로 명령어/그룹 찾아서 방 별 권한(command-setting(enabled, permission)) 변경하는 명령어

ex) !cmd 테스트(명령어/그룹이름) enabled false

ex) !cmd 테스트 permission admin

```
{
  cate: 'command',
  id: 4,
  name: '권한 변경',
  description: '관리자 전용',
  hide: true,
  setting: {
    enabled: true,
    permission: 'admin'
  },
  main: {
    verify: {
      key: ['cmd'],
      length: [
        [4]
      ],
    },
    check: function () {
      if (_msg[2] == 'enabled' && (_msg[3] == 'true' || _msg[3] == 'false')) {
        _msg[3] = JSON.parse(_msg[3])
        return true
      }
      if (_msg[2] == 'permission' && PERMISSION.indexOf(_msg[3]) != -1) return true
    },
    execute: function () {
      try {
        Object.keys(전역권한관리객체).map(v => [v, 전역권한관리객체[v]]).find(cmd => cmd[1].name == _msg[1])[1][_msg[2]] = _msg[3]
        replier.reply('OK')
      } catch (e) {
        replier.reply('존재하지 않는 명령어입니다.')
      }
    }
  }
},
```

( 말머리 등록 해주세요! )



odosk님의 게시글 더보기 >

❤️ 좋아요 1 💬 댓글 2

🔗 공유 | 신고

댓글 등록순 최신순 🔍

댓글알림 ☐



마른얼음  
이건 진짜 지렸다

2020.03.30. 16:37 답글쓰기



히어로 송 울프  
나도 지리고 싶당

2020.03.30. 16:42 답글쓰기

Hibot

댓글을 남겨보세요




등록

'| 자유 게시판 |' 게시판 글

이 게시판 새글 구독하기 ☐

|                                                 |               |             |
|-------------------------------------------------|---------------|-------------|
| 질문이요! <span>[9]</span>                          | 와와와와          | 2020.03.30. |
| 딱 한번만 비슷한 질문 또 올립니다..ㅜ죄송합니다 ☹️ <span>[8]</span> | 별사탕           | 2020.03.30. |
| 명령어 처리 함수 개선판 <span>[2]</span>                  | odosk         | 2020.03.30. |
| 등업언제 해주세요? <span>[13]</span>                    | 맛있는 연어 스테이크   | 2020.03.30. |
| 간단한 질문 ☹️ <span>[5]</span>                      | KR   끝말러   KR | 2020.03.30. |

이 카페 인기글




"시초가 중국발" 발언 관련 해명글 및 부탁의 글

안녕하세요

트위치봇

♡5 💬3

♡0 💬5



네이버에서 지역별 날씨 정보 크롤링하기 / 스왑 알아서 주의 카톡방 나가기

Dark Tornado

hajeol4

♡0 💬15

간편 자동응답과 한글코딩을 메인으로 제공하는 카톡봇 개발 예정

성빈

♡1 💬16

안녕하세요

KHP

♡0 💬6

오픈이발 괴롭히기(메봇R)

ERROR

♡0 💬7

가르치기 리로드..

Milk2

♡0 💬9



혹시 여기서 변수를 저장하는 부분이 어딘지