**Distinct Method to filter Duplicate**

// Get Result

TypedQuery<WarrantyMaster> result = em.createQuery(query);

list = result.getResultList();

list = list.stream().filter(*distinctByKey*(o -> Arrays.*asList*(o.getWarrantyId()))).collect(Collectors.*toList*());

list.sort(Comparator.*comparing*(WarrantyMaster :: getWarrantyDescription ));

// Map

**for** (WarrantyMaster data : list) {

WarrantyMasterRes res = **new** WarrantyMasterRes();

res = mapper.map(data, WarrantyMasterRes.**class**);

res.setCoreAppCode(data.getCoreAppCode());

resList.add(res);

}

} **catch** (Exception e) {

e.printStackTrace();

log.info(e.getMessage());

**return** **null**;

}

**return** resList;

}

**private** **static** <T> java.util.function.Predicate<T> distinctByKey(java.util.function.Function<? **super** T, ?> keyExtractor) {

Map<Object, Boolean> seen = **new** ConcurrentHashMap<>();

**return** t -> seen.putIfAbsent(keyExtractor.apply(t), Boolean.***TRUE***) == **null**;

}