package com.twitter.search.core.earlybird.facets;

/\*\*

\* Counts facet occurrences and provides the top items

\* at the end. Actual subclass can implement this functionality differently: e.g. by using

\* a heap (priority queue) or a hashmap with pruning step.

\* The type R represents the facet results, which can e.g. be a thrift class.

\*/

public abstract class FacetAccumulator<R> {

/\*\* Called to notify the accumulator that the given termID has occurred in a document

\* Returns the current count of the given termID.

\*/

public abstract int add(long termID, int scoreIncrement, int penaltyIncrement, int tweepCred);

/\*\* After hit collection is done this can be called to

\* retrieve the items that occurred most often \*/

public abstract R getTopFacets(int n);

/\*\* After hit collection is done this can be called to retrieve all the items accumulated

\* (which may not be all that occurred) \*/

public abstract R getAllFacets();

/\*\* Called to reset a facet accumulator for re-use. This is an optimization

\* which takes advantage of the fact that these accumulators may allocate

\* large hash-tables, and we use one per-segment, which may be as many as 10-20 \*\*/

public abstract void reset(FacetLabelProvider facetLabelProvider);

/\*\* Language histogram accumulation and retrieval. They both have no-op default implementations.

\*/

public void recordLanguage(int languageId) { }

public LanguageHistogram getLanguageHistogram() {

return LanguageHistogram.EMPTY\_HISTOGRAM;

}

}