package com.twitter.search.core.earlybird.index.inverted;

import java.io.IOException;

import org.apache.lucene.index.TermsEnum;

import org.apache.lucene.util.BytesRef;

import com.twitter.search.common.util.io.flushable.Flushable;

import com.twitter.search.core.earlybird.index.EarlybirdIndexSegmentAtomicReader;

/\*\*

\* A two-way mapping between terms and their interned value (termID).

\*

\* Implementation of this interface must guarantee that termIDs are dense, starting at 0;

\* so they are good to be used as indices in arrays.

\*/

public interface TermDictionary extends Flushable {

int TERM\_NOT\_FOUND = EarlybirdIndexSegmentAtomicReader.TERM\_NOT\_FOUND;

/\*\*

\* Returns the number of terms in this dictionary.

\*/

int getNumTerms();

/\*\*

\* Create a TermsEnum object over this TermDictionary for a given index.

\* @param index

\*/

TermsEnum createTermsEnum(OptimizedMemoryIndex index);

/\*\*

\* Lookup a term in this dictionary.

\* @param term the term to lookup.

\* @return the term id for this term, or TERM\_NOT\_FOUND

\* @throws IOException

\*/

int lookupTerm(BytesRef term) throws IOException;

/\*\*

\* Get the term for given id and possibly its payload.

\* @param termID the term that we want to get.

\* @param text MUST be non-null. It will be filled with the term.

\* @param termPayload if non-null, it will be filled with the payload if the term has any.

\* @return Returns true, iff this term has a term payload.

\*/

boolean getTerm(int termID, BytesRef text, BytesRef termPayload);

}