Query Entity Configuration

/**
 * Query entity is a description of [IgniteCache cache] entry (composed of key and value)
 * in a way of how it must be indexed and can be queried.
 */
public class QueryEntity {
    private String keyType;
    private String valType;

    // Map of field names to type names.
    private LinkedHashMap<String, String> flds;

    // Collection of indexes.
    private Collection<QueryIndex> idxs;

    // In addition to the obvious getters and setters,
    // we should also have these convenience methods.
    // All these methods should throw an exception
    // in case if a duplicate field or index already exists.
    public void addField(String name, String type);
    public void addField(String name, Class<?> type);
    public void addIndex(QueryIndex idx);
    }

    // I only show constructors here, but we should also have
    // corresponding setter methods.
    public class QueryIndex {
        private LinkedHashMap<String, Boolean> fields;
        private QueryIndexType idxType;

        // Creates index for one field.
        //
        // If index is sorted, then ascending sorting is used by default.
        // To specify sort order, use the next method.
        //
        // This constructor should also have a corresponding setter method.
        public QueryIndex(String field, QueryIndexType type) {...}

        // Creates index for one field. The last boolean parameter
        // is ignored for non-sorted indexes.
        //
        // This constructor should also have a corresponding setter method.
        public QueryIndex(String field, QueryIndexType type, boolean asc) {...}

        // Creates index for multiple fields.
        //
        // If index is sorted, then ascending sorting is used by default.
        // To specify sort order, use the next method.
        //
        // This constructor should also have a corresponding setter method.
        public QueryIndex(LinkedHashMap<String, Boolean> fields, QueryIndexType type) {...}

        // Creates index for multiple fields.
        // Fields are passed in as a map, with field name as a key and sort order
        // as a value (true for ascending). The value is ignored for non-sorted indexes.
        //
        // This constructor is useful for sorted indexes, where it is necessary to specify
        // a separate sort order for each field.
        //
        // This constructor should also have a corresponding setter method.
        public QueryIndex(LinkedHashMap<String, Boolean> fields, QueryIndexType type) {...}

        // Basic getters.
        public LinkedHashMap<String, Boolean> getFields();
        public QueryIndexType getType();
        public List<String> getFieldNames();
        public boolean hasField(String field);
// Returns null if field does not exist.
public Boolean getSortOrder(String field);
}

enum QueryIndexType {
    SORTED, FULLTEXT, GEOSPATIAL
}