NAME

mtbl_merger − merge multiple MTBL data sources into a single output

SYNOPSIS

#include <mtbl.h>

Merger objects:

struct mtbl_merger *
mtbl_merger_init(const struct mtbl_merger_options ***mopt*);**

void
mtbl_merger_destroy(struct mtbl_merger ****m*);**

void
mtbl_merger_add_source(struct mtbl_merger ****m*, const struct mtbl_source ****s***);**

const struct mtbl_source *
mtbl_merger_source(struct mtbl_merger ****m*);**

Merger options:

struct mtbl_merger_options *
mtbl_merger_options_init(void);

void
mtbl_merger_options_destroy(struct mtbl_merger_options ****mopt*);**

void
mtbl_merger_options_set_merge_func(
struct mtbl_merger_options ****mopt*,*
mtbl_merge_func *fp*,
void ****clos*);*

typedef void
(*mtbl_merge_func)(void ****clos*,*
const uint8_t ****key*, **size_t** len_key,*
const uint8_t ****val0*, **size_t** len_val0,*
const uint8_t ****val1*, **size_t** len_val1,*
uint8_t ****merged_val*, **size_t ****len_merged_val***);*

DESCRIPTION

Multiple MTBL data sources may be merged together using the **mtbl_merger** interface, which reads key−value entries from one or more sources and provides these entries in sorted order. The sorted entries may be consumed via the **mtbl_source**(3) and **mtbl_iter**(3) interfaces.

Because the MTBL format does not allow duplicate keys, the caller must provide a function which will accept a key and two conflicting values for that key and return a replacement value. This function may be called multiple times for the same key if more than two sources are being merged.

mtbl_merger objects are created with the **mtbl_merger_init**() function, which requires a non−NULL *mopt* argument which has been configured with a merge function *fp*.

One or more **mtbl_reader** objects must be provided as input to the **mtbl_merger** object by calling **mtbl_merger_add_source**(). After the desired sources have been configured, **mtbl_merger_source**() should be called in order to consume the merged output via the **mtbl_source**(3) interface.

Merger options

merge_func

This option specifies a merge function callback, consisting of a function pointer *fp* and a pointer to

user data *clos* which will be passed as the first argument to *fp*. The merge function callback will be used during iteration over the **mtbl_merger** object to merge entries with duplicate keys in the input sources.

The remaining arguments to the merge function are:

key — pointer to the key for which there exist duplicate values.

len_key — length of the key.

val0 — pointer to the first value.

len_val_0 — length of the first value.

val1 — pointer to the second value.

len_val_1 — length of the second value.

merged_val — pointer to where the callee should place its merged value.

len_merged_val — pointer to where the callee should place the length of its merged value.

merged_val must be allocated with the system allocator, and the **mtbl_merger** interface takes responsibility for free()ing the value once it is no longer needed.

The callee may provide an empty value as the merged value, in which case *merged_val* must still contain an allocated, non−NULL value and *len_merged_val* must contain the value 0.

The callee may indicate an error by returning NULL in the *merged_val* argument, which will abort iteration over the **mtbl_merger** object.

RETURN VALUE

If the merge function callback is unable to provide a merged value (that is, it fails to return a non−NULL value in its *merged_val* argument), the merge process will be aborted, and any iterators over the **mtbl_merger** object (via the **mtbl_source**(3) interface) will return **mtbl_res_failure**.