
Name

rd_editcut — Rivendell Edit Cut C Library Function

Synopsis

```
#include <rivwebcapi/rd_editcut.h>

int    RD_EditCut(cut[],    edit_cut_values,    hostname[],    username[],
passwd[], ticket[], cartnumber, cutnumber, user_agent[], numrecs);

struct rd_cut * cut[];
struct edit_cut_values edit_cut_values;
const char hostname[];
const char username[];
const char passwd[];
const char ticket[];
const unsigned cartnumber;
const unsigned cutnumber;
const char user_agent[];
unsigned * numrecs;
```

Description

RD_EditCut is the function to use to edit the fields within a cut that already exists in the Rivendell Database.

This function edits a pre-existing cut. User must provide the cart number, cut number and any fields which they would like to change. The structure edit_cut_values must be used to tell the API which fields will be changed.

Table 1. RD_EditCart function call fields

FIELD NAME	FIELD TYPE	MEANING	REMARKS
*rd_cut	Pointer to rd_cut structure	Memory location to store cut information	Mandatory
edit_cut_values	edit_cut_values structure	This structure contains the new cut information to update	Mandatory
hostname	Character Array	Name Of Rivendell DB Host	Mandatory
username	Character Array	Rivendell User Name	Mandatory When NO Ticket Provided
passwd	Character Array	Rivendell User Password	Mandatory When NO Ticket Provided
ticket	Character Array	Rivendell Authentication Ticket	Mandatory When NO User/Password Pair Provided.
cartnumber	unsigned integer	Cart Number	Mandatory
cutnumber	unsigned integer	Cut Number	Mandatory

FIELD NAME	FIELD TYPE	MEANING	REMARKS
user_agent	Character Array	User Agent Value put into HTTP request	Optional (default is Rivendell-C-API/x.x.x)
*numrecs	pointer to integer	memory location for number of records returned	Mandatory

This routine expects an input structure of type edit_cut_values - listed below. Each field has a boolean flag field (starting with use_) which designates whether to update the field's value or not. One (1) = true - Use the value in the field or Zero (0) - Ignore the field.

The edit_cut_values structure must be pre-filled with zeroes and has the following format:

```

struct edit_cut_values {
    int cut_evergreen;
    int use_cut_evergreen;
    char cut_description[257];
    int use_cut_description;
    char cut_outcue[257];
    int use_cut_outcue;
    char cut_isrc[49];
    int use_cut_isrc;
    char cut_isci[129];
    int use_cut_isci;
    struct tm cut_start_datetime;
    int use_cut_start_datetime;
    struct tm cut_end_datetime;
    int use_cut_end_datetime;
    int cut_sun;
    int use_cut_sun;
    int cut_mon;
    int use_cut_mon;
    int cut_tue;
    int use_cut_tue;
    int cut_wed;
    int use_cut_wed;
    int cut_thu;
    int use_cut_thu;
    int cut_fri;
    int use_cut_fri;
    int cut_sat;
    int use_cut_sat;
    char cut_start_daypart[14];
    int use_cut_start_daypart;
    char cut_end_daypart[14];
    int use_cut_end_daypart;
    unsigned cut_weight;
    int use_cut_weight;
    unsigned cut_validity;
    int use_cut_validity;
    unsigned cut_coding_format;
    int use_cut_coding_format;

```

```
unsigned cut_sample_rate;
int use_cut_sample_rate;
unsigned cut_bit_rate;
int use_cut_bit_rate;
unsigned cut_channels;
int use_cut_channels;
int cut_play_gain;
int use_cut_play_gain;
int cut_start_point;
int use_cut_start_point;
int cut_end_point;
int use_cut_end_point;
int cut_fadeup_point;
int use_cut_fadeup_point;
int cut_fadedown_point;
int use_cut_fadedown_point;
int cut_segue_start_point;
int use_cut_segue_start_point;
int cut_segue_end_point;
int use_cut_segue_end_point;
int cut_segue_gain;
int use_cut_segue_gain;
int cut_hook_start_point;
int use_cut_hook_start_point;
int cut_hook_end_point;
int use_cut_hook_end_point;
int cut_talk_start_point;
int use_cut_talk_start_point;
int cut_talk_end_point;
int use_cut_talk_end_point;
};
```

When successful function will return the number of records sent (numrecs) and a rd_cut structure which is stored in the provided memory locations. The rd_cut structure has the following fields:

```
struct rd_cut {
    char cut_name[11];
    unsigned cut_cart_number;
    unsigned cut_cut_number;
    int cut_evergreen;
    char cut_description[257];
    char cut_outcue[257];
    char cut_isrc[49];
    char cut_isci[129];
    unsigned cut_length;
    struct tm cut_origin_datetime;
    struct tm cut_start_datetime;
    struct tm cut_end_datetime;
    int cut_sun;
    int cut_mon;
    int cut_tue;
```

```
int cut_wed;
int cut_thu;
int cut_fri;
int cut_sat;
char cut_start_daypart[9];
char cut_end_daypart[9];
char cut_origin_name[257];
unsigned cut_weight;
struct tm cut_last_play_datetime;
unsigned cut_play_counter;
unsigned cut_local_counter;
unsigned cut_validity;
unsigned cut_coding_format;
unsigned cut_sample_rate;
unsigned cut_bit_rate;
unsigned cut_channels;
int cut_play_gain;
int cut_start_point;
int cut_end_point;
int cut_fadeup_point;
int cut_fadedown_point;
int cut_segue_start_point;
int cut_segue_end_point;
int cut_segue_gain;
int cut_hook_start_point;
int cut_hook_end_point;
int cut_talk_start_point;
int cut_talk_end_point;
};
```

All character arrays above are the sizes listed and are null-terminated.
Character encoding is UTF-8.

RETURN VALUE

On success, zero is returned. Using the provided parameters an `rd_cut` structure is returned and the number of records is returned.

If a server error occurs a -1 is returned. If a client error occurs a specific error number is returned.

ERRORS

400 Invalid Parameter(s).

403 User Authentication Error.

403 Edit Audio Forbidden.

404 No Such Cart Exists.

nnn Unknown Error Occurred.