

---

## Name

rd\_listcartcuts — Rivendell List Cart/Cuts C Library Function

## Synopsis

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

int RD_ListCartCuts(cart[], hostname[], username[], passwd[], ticket[],
cartnumber, user_agent[], numrecs);

struct rd_cart * cart[];
const char hostname[];
const char username[];
const char passwd[];
const char ticket[];
const unsigned cartnumber;
const char user_agent[];
unsigned * numrecs;

struct rd_cut *RD_ListCartCuts_GetCut(cart, cut_rec);

struct rd_cart * cart;
int cut_rec;

void RD_ListCartCuts_Free(cart);

struct rd_cart * cart;
```

## Description

**RD\_ListCartCuts** is the function to use to list the fields within a single cart that already exists in the Rivendell Database. Unlike **RD\_ListCart**(7), **RD\_ListCartCuts**(7) will also fetch the full list of cuts associated with the requested cart.

**Table 1. RD\_ListCartCuts function call fields**

FIELD NAME	FIELD TYPE	MEANING	REMARKS
*cart	Pointer to rd_cart structure	Memory location to store cart information	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
user_agent	Character Array	User Agent Value put into HTTP request	Optional (default is Rivendell-C-API/x.x.x)

FIELD NAME	FIELD TYPE	MEANING	REMARKS
*numrecs	pointer to integer	memory location for number of records returned	Mandatory

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

```
struct rd_cart {
    unsigned cart_number;           /* Cart Number */
    unsigned cart_type;             /* Cart Type */
    char cart_grp_name[41];         /* Group Name */
    char cart_title[1021];         /* Cart Title */
    char cart_artist[1021];        /* Artist */
    char cart_album[1021];         /* Album */
    int cart_year;                 /* Year */
    char cart_label[257];          /* Label */
    char cart_client[257];         /* Client */
    char cart_agency[257];         /* Agency */
    char cart_publisher[257];      /* Publisher */
    char cart_composer[257];       /* Composer */
    char cart_conductor[257];      /* Conductor */
    char cart_user_defined[1021];  /* User Defined */
    int cart_usage_code;           /* Usage Code */
    int cart_forced_length;        /* Forced Length */
    int cart_average_length;       /* AverageLength */
    int cart_length_deviation;     /* Length Deviation */
    int cart_average_segue_length; /* Average Segue Length */
    int cart_average_hook_length;  /* Average Hook Length */
    unsigned cart_cut_quantity;    /* Cut Quantity */
    unsigned cart_last_cut_played; /* Last Cut Played */
    unsigned cart_validity;        /* Validity */
    int cart_enforce_length;       /* Enforce Length Flag */
    int cart_asynchronous;         /* Asynchronous Flag */
    char cart_owner[257];          /* Owner */
    struct tm cart_metadata_datetime; /* Metadata Datetime */
    char cart_notes[4096];         /* Notes */
    struct rd_cut **cart_cuts;     /* Cut list */
};
```

All character arrays use UTF-8 encoding and are null-terminated.

## Accessing Cut Information

Information about the cuts associated with the cart can be accessed from the returned rd\_cart structure through use of the **RD\_ListCartCuts\_GetCut()** function. The *cut\_rec* parameter should range between 0 and one less than the value of the *cart\_cut\_quantity* member of the cart's rd\_cart structure.

Cut information is returned in the form of a rd\_cut structure, which has the following fields:

```
struct rd_cut {
```

```
char cut_name[41];           /* Cut Name */
unsigned cut_cart_number;    /* Parent Cart Number */
unsigned cut_cut_number;    /* Cut Number */
int cut_evergreen;          /* Boolean */
char cut_description[257];  /* Description */
char cut_outcue[257];       /* Outcue */
char cut_isrc[49];          /* International Standard Recording C
char cut_isci[129];         /* Industry Standard Commercial Ident
unsigned cut_length;        /* Milliseconds */
struct tm cut_origin_datetime; /* Origin Datetime */
struct tm cut_start_datetime; /* Start Datetime */
struct tm cut_end_datetime; /* End Datetime */
int cut_sun;                /* Playable on Sunday */
int cut_mon;                /* Playable on Monday */
int cut_tue;                /* Playable on Tuesday */
int cut_wed;                /* Playable on Wednesday */
int cut_thu;                /* Playable on Thursday */
int cut_fri;                /* Playable on Friday */
int cut_sat;                /* Playable on Saturday */
char cut_start_daypart[15]; /* Start Daypart */
char cut_end_daypart[15];   /* End Daypart */
char cut_origin_name[257];  /* Hostname of Ingestion System */
char cut_origin_login_name[1021]; /* RD Username on Ingestion System */
char cut_source_hostname[1021]; /* Hostname of Originating System */
unsigned cut_weight;        /* Cut Weighting */
struct tm cut_last_play_datetime; /* Datetime of Last OnAir Play-out */
unsigned cut_play_counter;   /* Number of Times Played */
unsigned cut_local_counter;  /* Plays Since Rotation Changed */
unsigned cut_validity;       /* Cut Validity */
unsigned cut_coding_format;  /* 0 = PCM, 1 = MPEG */
unsigned cut_sample_rate;    /* Samples per Second */
unsigned cut_bit_rate;       /* Bits per Second */
unsigned cut_channels;       /* Audio Channels */
int cut_play_gain;           /* dBFS */
int cut_start_point;         /* Milliseconds from Start */
int cut_end_point;           /* Milliseconds from Start */
int cut_fadeup_point;        /* Milliseconds from Start */
int cut_fadedown_point;     /* Milliseconds from Start */
int cut_segue_start_point;   /* Milliseconds from Start */
int cut_segue_end_point;     /* Milliseconds from Start */
int cut_segue_gain;          /* dBFS */
int cut_hook_start_point;    /* Milliseconds from Start */
int cut_hook_end_point;      /* Milliseconds from Start */
int cut_talk_start_point;    /* Milliseconds from Start */
int cut_talk_end_point;      /* Milliseconds from Start */
};
```

All character arrays use UTF-8 encoding and are null-terminated.

## Freeing Memory

When the returned `rd_cart` structure is no longer required, it should be freed by passing it to the **RD\_ListCartCuts\_Free()** function.

## RETURN VALUE

On success, zero is returned. Using the provided parameters an `rd_cart` 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 Missing Cart.

403 User Authentication Error.

404 No Such Cart Exists.

nnn Unknown Error Occurred.