Nintendo Wi-Fi Connection

NITRO-DWC Programming Manual

General-Purpose Ranking Edition

Version 1.0.4

The contents in this document are highly confidential and should be handled accordingly.

NTR-06-0460-001-B
Released: September 12, 2006

Confidential

These coded instructions, statements, and computer programs contain proprietary information of Nintendo of America Inc. and/or Nintendo Company Ltd. and are protected by Federal copyright law. They may not be disclosed to third parties or copied or duplicated in any form, in whole or in part, without the prior written consent of Nintendo.

Table of Contents

Code

Tables

Figures

Revision History

Version	Revision Date	Description
1.0.4	06/26/2006	Revised text specific to 4.2.3, CSV File Formats.
1.0.3	07/21/2006	2.1.3.3 Revised notes specific to cases where rank order cannot be identified.
1.0.2	06/19/2006	2.1.3.3 Added notes specific to cases where rank-order cannot be identified.
1.0.1	06/12/2006	3.6 Corrected errors in "Screen Composition" 3.8.3 Corrected errors in "Delete Entry dialog box"
1.0.0	06/06/2006	Initial version.

1 Introduction

This document explains how to use the General-Purpose Ranking library of the NITRO-DWC and its related administration tool. It also explains the Web development interface.

The following features are provided by General-Purpose Ranking library:

- Upload scores
- Get one's own rank order
- Get the top ranking list
- Get a ranking list of players with scores near one's own score
- Get lists of ranking among Friends and among Rivals

In order to realize these features, the General-Purpose Ranking library provides a set of APIs for communicating with the Ranking server.

The General-Purpose Ranking library makes internal use of `DWC_GHTTP` to communicate with the Ranking server. Thus, the General-Purpose Ranking library and `DWC_GHTTP` cannot be used simultaneously. Furthermore, all communication errors associated with general-purpose ranking are generated by `DWC_GHTTP`, so the error codes depend on `DWC_GHTTP`.

To read more about the actual use of this library, see the Nintendo Wi-Fi Connection Programming Guide.

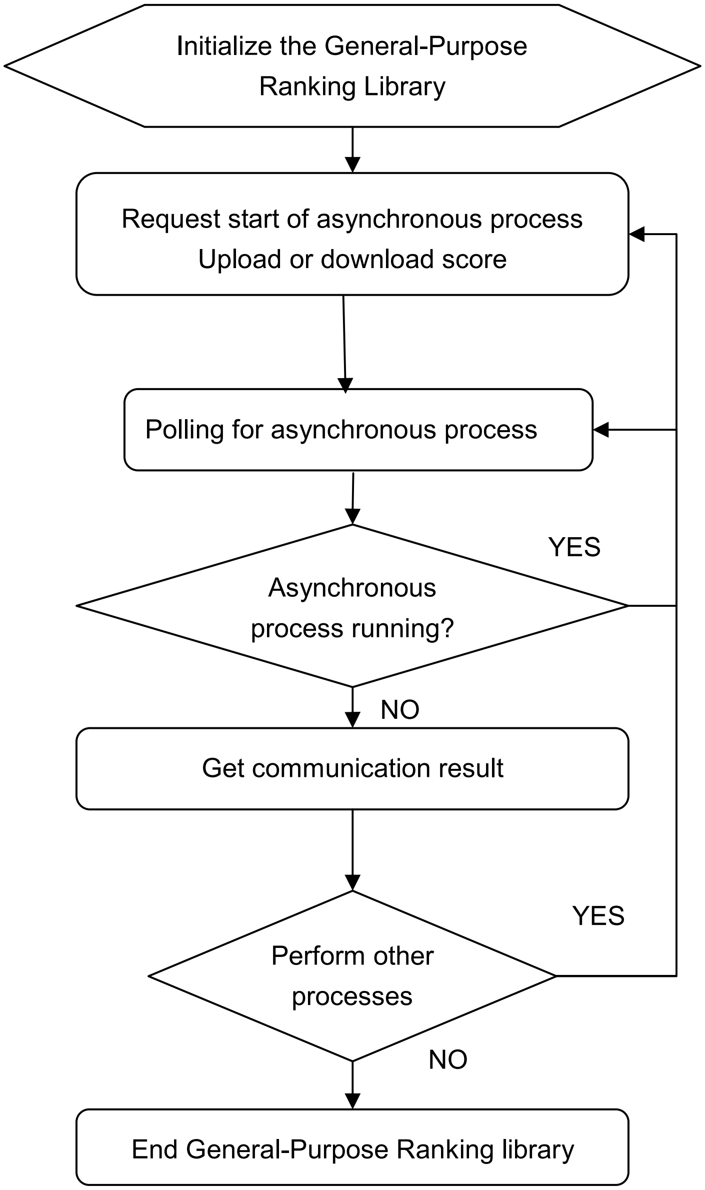
2 How to Use the Library

This chapter explains how to use the General-Purpose Ranking library.

2.1 Process Flow

The process flow for the General-Purpose Ranking library is show in Figure 2-1.

Figure 2-1 General-Purpose Ranking Library Process Flow


Initialize the General-Purpose Ranking Library
Request start of asynchronous process Upload or download score
Polling for asynchronous process
Asynchronous process running?
YES
NO
Get communication result
Perform other processes
YES
NO
End General-Purpose Ranking library

Note: The General-Purpose Ranking library must be terminated when an asynchronous process is cancelled or when an error is generated.

2.1.1 Initializing the General-Purpose Ranking library

The first step involves calling the `DWC_RnkInitialize` function to initialize the General-Purpose Ranking library. The arguments of this function take the following data:

- Initialization data

  This specifies the initialization data character string for the General-Purpose Ranking library that is disclosed by Nintendo after you have submitted a design statement. Every title is assigned its own unique initialization data, so you should manage this data with great care.

- User data

  This specifies the `DWCUserData` structure for the user data. The account data must be for an authenticated account. Initialization will fail if it is an unauthenticated account that has never been connected to Nintendo Wi-Fi Connection.

The General-Purpose Ranking server has been prepared in two versions: one for development and one for production. Which server is used as the connection target is determined at the time of initialization by the authentication server login setting of the `DWC_SetAuthServer` function. If `DWC_CONNECTINET_AUTH_RELEASE` is specified in the function, the connection target is the production server. If `DWC_CONNECTINET_AUTH_TEST` is specified, the target is the development server.

The development server and the production server have independent databases, so the ranking in each can be constructed differently.

Make sure to use the production server for your commercial product.

Code 2-1 Initialization of the Ranking library

```
DWCRnkError res;

// Initialize the Ranking library
  res = DWC_RnkInitialize(RANKING_INITDATA, &userdata );

if ( res != DWC_RNK_SUCCESS ){

break; // Error process on failure to initialize

}
```

2.1.2 Uploading Scores

Scores are uploaded using an asynchronous process. Call the `DWC_RnkPutScoreAsync` function to begin this process. Only one asynchronous process can execute at a time.

Polling is conducted from the start of the asynchronous process until the process is completed by the `DWC_RnkProcess` function (described below).

You can specify the category ID, the region, the score, and the user-defined data for a score that you are uploading.

The category ID is an identifier used because a game title can have multiple ranking lists. For the category ID you can specify any value between 0 and 100 (`DWC_RNK_CATEGORY_MAX`).

The region can be specified as `DWC_RNK_REGION_JP`, `DWC_RNK_REGION_US`, or `DWC_RNK_REGION_EU` (for Japan, the USA, and Europe, respectively). The region specified here can be used as a filter when downloading scores. You could, for example, get one ranking for Japan and another ranking for both Japan and the USA combined.

The user-defined data can take any binary data up to a maximum of 768 bytes (`DWC_RNK_DATA_MAX`).

If the user's own score is already registered with the same category ID, a newly uploaded score will always overwrite the existing score.

Code 2-2 Request to Start Uploading Scores

```
DWCRnkError res;

// Request to start uploading scores
  res = DWC_RnkPutScoreAsync( 10,                  // category
                              DWC_RNK_REGION_JP,   // region
                              1234,                // score
                              (void*)"test data",  // user-defined data
                              strlen("test data") + 1 );

if( res != DWC_RNK_SUCCESS ){

break; // Error process on failure

  }
```

2.1.3 Downloading Scores

Scores are downloaded using an asynchronous process. Call the `DWC_RnkGetScoreAsync` function to begin this process. Only one asynchronous process can execute at a time.

Polling is conducted from the start of the asynchronous process until the process is completed by the `DWC_RnkProcess` function (described below).

There are four different `get_modes` for score downloading. Each mode gets a different set of data. Table 2-1 shows these four modes and what each mode gets.

Table 2-1 The get_modes for Downloading Scores

get_mode	Data Retrieved
Rank order	Gets the user's own place in the ranking. Ranking is determined by comparing registered scores. A parameter specifies whether to rank in ascending or descending order.
Top-rankings list	Gets a list of rankings, starting with the top score and continuing for the specified number of scores.
Near-rankings list	Gets a list of scores ranked in order near your score. The list contains the specified number of scores.
Friend-rankings list	Gets a list of scores for as many as 64 friends, ranked in order of score.

2.1.3.1 How to Specify the DWCRnkGetParam Structure

This section explains how to specify the parameters of the `DWCRnkGetParam` structure, which gets passed as an argument to the `DWC_RnkGetScoreAsync` function.

`DWCRnkGetParam` is defined as a structure that internalizes multiple unions, so you need to specify the parameters in the appropriate fields for the different `get_modes`.

The DWCRnkGetParam.size field

This parameter, which specifies the size of the structure, is common to all `get_modes`. The size of the structure for each `get_mode` is shown below:

Table 2-2 Size Specified in Parameter for Each get_mode

Rank order get_mode (`DWC_RNK_GET_MODE_ORDER`)	`sizeof( DWCRnkGetParam.order )`
Top-rankings list get_mode (`DWC_RNK_GET_MODE_TOPLIST`)	`sizeof( DWCRnkGetParam.toplist )`
Near-rankings list get_mode (`DWC_RNK_GET_MODE_NEAR`)	`sizeof( DWCRnkGetParam.near )`
Friend-rankings list get_mode (`DWC_RNK_GET_MODE_FRIENDS`)	`sizeof( DWCRnkGetParam.friends )`

The DWCRnkGetParam.order field

These parameters are specified for the *rank order* `get_mode`.

Table 2-3 Parameters Specified for *rank order* get_mode

`DWCRnkGetParam.order.sort`	Specifies the sort order for the scores: `DWC_RNK_ORDER_ASC`: ascending `DWC_RNK_ORDER_DES`: descending
`DWCRnkGetParam.order.since`	Gets the rankings that have changed since the specified number of minutes ago. If 0 is specified, the rankings for all data are downloaded. For example, if 180 is specified, then rankings of the scores that have been updated within the past 3 hours (.i.e., 180 minutes) are downloaded.

The DWCRnkGetParam.toplist field

These parameters are specified for the *top-rankings list* `get_mode`.

Table 2-4 Parameters Specified for *top-rankings list* get_mode

`DWCRnkGetParam.toplist.sort`	Specifies the sort order for the scores: `DWC_RNK_ORDER_ASC`: ascending `DWC_RNK_ORDER_DES`: descending
`DWCRnkGetParam.toplist.since`	Gets the rankings that have changed since the specified number of minutes ago. If 0 is specified, the rankings for all of the data are downloaded. For example, if 180 is specified, then rankings that have been updated within the past 3 hours (.i.e., 180 minutes) are downloaded.
`DWCRnkGetParam.toplist.limit`	Specifies the maximum number of listed rankings to get. Can take a value between 1 and 10 (`DWC_RNK_GET_MAX`).

The WCRnkGetParam.near field

These parameters are specified for the *near-rankings list* `get_mode`.

Table 2-5 Parameters Specified for the *near-rankings list* get_mode

`DWCRnkGetParam.near.sort`	Specifies the sort order for the scores: `DWC_RNK_ORDER_ASC`: ascending `DWC_RNK_ORDER_DES`: descending
`DWCRnkGetParam.near.since`	Gets the rankings that have changed since the specified number of minutes ago. If 0 is specified, the rankings for all of the data are downloaded. For example, if 180 is specified then rankings that have been updated within the past 3 hours (.i.e., 180 minutes) are downloaded.
`DWCRnkGetParam.near.limit`	Specifies the maximum number of listed rankings to get. Can take a value between 2 and 10 (`DWC_RNK_GET_MAX`). Since user's own score is invariably at the head of the list, you must specify a value of at least 2.

The DWCRnkGetParam.friend field

These parameters are specified for the *Friend-rankings list* `get_mode`

Table 2-6 Parameters Specified for the *friend-rankings list* get_mode

`DWCRnkGetParam.friends.sort`	Specifies the sort order for the scores: `DWC_RNK_ORDER_ASC`: ascending `DWC_RNK_ORDER_DES`: descending
`DWCRnkGetParam.friends.since`	Gets the rankings that have changed since the specified number of minutes ago. If 0 is specified, the rankings for all of the data are downloaded. For example, if 180 is specified, then rankings that have been updated within the past 3 hours (.i.e., 180 minutes) are downloaded.
`DWCRnkGetParam.friends.limit`	Specifies the maximum number of listed rankings to get. Can take a value between 2 and 10 (`DWC_RNK_GET_MAX`). Since user's own score is invariably at the head of the list, you must specify a value of at least 2.

`DWCRnkGetParam.friends.`
`friends[64]`
	Specifies a list of Friend GS profile IDs. The list can have up to 64 (`DWC_RNK_FRIENDS_MAX`) values. If the list does not have 64 values, store the entries from the start and zero-fill the remaining region.

Code 2-3 Request to Start Score Download to Get Rank Order

```
DWCRnkError res;

  // Start request to get rank order
  DWCRnkGetParam       param;        // Parameter for getting rank order
  param.size = sizeof( param.order );
  param.order.since = 0;
  param.order.sort = DWC_RNK_ORDER_ASC;

  res = DWC_RnkGetScoreAsync( DWC_RNK_GET_MODE_ORDER,     // mode
                              10,                  // category
                              DWC_RNK_REGION_JP,   // region
                              &param );            // parameter

if( res != DWC_RNK_SUCCESS ){

break; // Failure error process

  }
```

Code 2-4 Request to Start Score Download to Get Top-Ranking List

```
  DWCRnkError res;

  // Start request to get rank order
  DWCRnkGetParam       param; // The parameter when getting top rankings
  param.size = sizeof( param.toplist );
  param.toplist.since = 0;
  param.toplist.sort = DWC_RNK_ORDER_ASC;
  param.toplist.limit = 10;

  res = DWC_RnkGetScoreAsync( DWC_RNK_GET_MODE_TOPLIST, // mode
                              10,                  // category
                              DWC_RNK_REGION_JP,   // region
                              &param );            // parameter

if( res != DWC_RNK_SUCCESS ){

break; // Failure error process

  }
```

2.1.3.2 Get Communication Result in *rank order* get_mode

When `get_mode` is *rank order*, once the process to download the score has ended normally, you can use the `DWC_RnkResGetOrder` function to get the communications result. Note that this function will fail if `DWC_RnkGetScoreAsync` was called in any other `get_mode` setting.

If the user's own score has not been uploaded to the server, the function returns 0.

2.1.3.3 Get Communication Result in *top-*, *near-* and f*riend- rankings list* get_modes

Normally, when the get_mode is set to one of the ranking list modes, you can use the `DWC_RnkResGetRowCount` function to get the number of rows in the list of score rankings once the download process has ended. You can also access the list by using the `DWC_RnkResGetRow` function to get the data for each row. (See note 1.)

The *order* field in the `DWCRnkData` structure stores a rank-order value only in the row that holds the user's own score. Since 0 is stored in all the other rows in the *order* field, you will need to implement suitable numbering inside your game. (See note 2.)

When multiple users have the same score, there is no defined way of ordering them in the list.

If the user's own score has not been registered to the server and an attempt is made to get a near-rankings list or a friend-rankings list, the score will be treated as 0. In addition, the region code will be returned as -1 and the user-defined data will be empty.

Note 1: The pointer to the user-defined data of the `DWCRnkData` structure obtained by the `DWC_RnkResGetRow` function (i.e., the void* userdata member) directly references the internal communication buffer. Consequently, the content of this buffer is lost when the Ranking library terminate or the next asynchronous process starts.

Note 2: When `get_mode` is *near-rankings list* and it cannot be determined how many others with the same score rank higher than the user, it is impossible to get a correct ranking for those players who are higher up in the rank order than the user. Depending on how the list is displayed, the user might notice the discrepancy in the list that is retrieved using the top-ranking list get mode. Because the specification takes the load on the server into account, there is fundamentally no way to resolve this issue. Instead, you will need to implement some kind of compromise, such as displaying only the user's own rank, broadening the score range so there is less of a chance that scores overlap, or utilizing some form of numbering that may incorporate a certain margin of error.

2.1.3.4 Get the Communication Result, Get Totals

Call the `DWC_RnkResGetTotal` function to get the total number of ranked scores for the rank-order value obtained by the `DWC_RnkGetScoreAsync` function. This total value is the total number of scores that met the filtering conditions specified when `DWC_RnkGetScoreAsync` was called.

2.1.3.5 Get List of Rankings among Rivals

In *friend-rankings list* `get_mode`, you can specify the GS profile IDs of Rivals in the GS profile ID list. Doing this gets you a list of rankings among Rivals Note that exchanges of data among users who are not registered Friends must conform to the Nintendo Wi-Fi Connection Guidelines for Project Planning.

Code 2-5 Accessing the Obtained Ranking List

```
  DWCRnkError res;
  u32 count;

  // Get the number of rows in the obtained list
  res = DWC_RnkResGetRowCount( &count );

if( res != DWC_RNK_SUCCESS ){

goto exit; // Error processing on failure

  }

  // Get one row at a time and output for debugging
  for( i=0; i<count; i++ ){
         DWCRnkData data;

         if( DWC_RnkResGetRow( &data, (u32)i ) != DWC_RNK_SUCCESS ){
                break; // Error
         }

         OS_TPrintf("%dth score=%d pid=%d rgn=%d update=%d data=%s ¥n",
                       data.order, data.score, data.pid, data.region,
data.lastupdate, data.userdata );
  }
```

2.1.4 Polling the Asynchronous Process to Get the Communication Result Status

Conduct polling once the asynchronous process has started by periodically calling the `DWC_RnkProcess` function. We recommend calling the function at a frequency of around once every frame (this also works at 30 fps).

The `DWC_RnkProcess` function returns `DWC_RNK_SUCCESS` while the asynchronous process is executing. When there are no more tasks, the function returns `DWC_RNK_PROCESS_NOTASK`. By monitoring this return value, you can detect when the asynchronous process has ended.

If an error occurs during the process, the function returns `DWC_RNK_IN_ERROR`. If there is an error, subsequent processing cannot continue. Terminate the General-Purpose Ranking library and try starting again from the initialization process.

Code 2-6 Asynchronous Process Polling

```
  // Asynchronous process
  while( (res = DWC_RnkProcess()) == DWC_RNK_SUCCESS ){

        // Wait for V blank
        GameWaitVBlankIntr();

}

switch( res ){
case DWC_RNK_PROCESS_NOTASK:
        // Asynchronous processing completed
        break;
case DWC_RNK_IN_ERROR:
        // Error processing on failure
        goto exit;
}

// Get communication result
switch( DWC_RnkGetState() ){
case DWC_RNK_STATE_COMPLETED:
        // Successful
        break;

case DWC_RNK_STATE_ERROR:
        // Error processing on failure
        goto exit;
}
```

2.2 Terminating the General-Purpose Ranking Library

Terminate by calling the `DWC_RnkShutdown` function after all processing has completed or the process has quit because of an error. Calling this function releases the memory that was used by the General-Purpose Ranking library and resets its internal state.

The pointer returned by the `DWC_RnkResGetRow` function directly references the Receive buffer, so it becomes invalidated when the General-Purpose Ranking library terminates If you need to save data, make sure to do so before terminating the library.

2.3 Canceling a Process and Error Handling

To cancel an asynchronous process while it is executing, call the `DWC_RnkCancelProcess` function.

When a process is canceled, the General-Purpose Ranking library enters into an error state. Since subsequent processing will not proceed when an error has been generated, you need to terminate the library and try again starting from the initialization process.

Because of the nature of the Internet, a network error may lead to the generation of an error while a process is executing even when the General-Purpose Ranking library is being properly used. For this reason, be sure to implement a suitable error-handling process.

2.4 About Error Codes

The General-Purpose Ranking library uses the `DWC_GHTTP` for internal communications. Consequently, the DWC error codes for communications errors are set by `DWC_GHTTP`.

2.5 About the rank_sample Sample Program

The DWC package includes a sample program that uses the General-Purpose Ranking library. The game name for this sample is dwctest.

Note that all developers use common initialization data with this sample. As a result, the ranking data you register on the server may be viewed and altered by other developers. Do not develop a real game using the initialization data from this sample.

2.6 Dependency on DWC_GHTTP

The General-Purpose Ranking library uses `DWC_GHTTP` for internal communications. Therefore, the API to initialize and terminate `DWC_GHTTP` internally calls the functions that initialize and terminate the General-Purpose Ranking library. Consequently, you cannot conduct communications simultaneously with both the General-Purpose Ranking library and `DWC_GHTTP`.

2.7 Amount of Memory Used

The General-Purpose Ranking library uses the DWC_Alloc function internally to obtain memory.

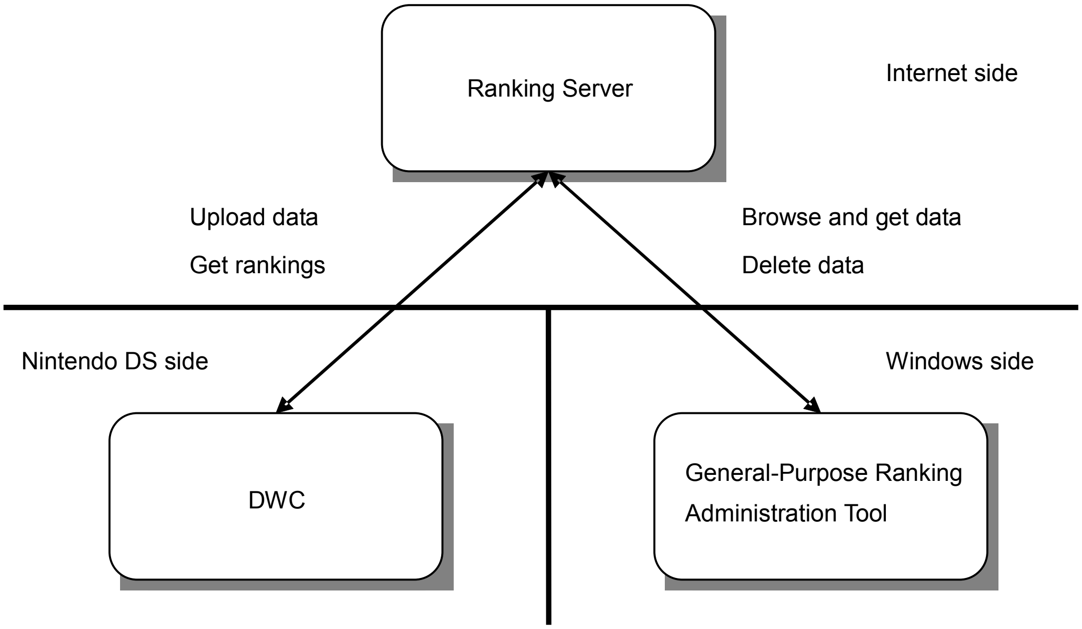
As a rule of thumb, the amount of memory used for uploading scores = 2 x (size of user-defined data + approx. 200 bytes). The amount of memory used for downloading scores = approx. 200 bytes + the total volume of score data being downloaded. This volume depends on the size of user-defined data and the maximum number of items in the obtained ranking list.

3 General-Purpose Ranking Administration Tool

3.1 Overview

The General-Purpose Ranking Administration Tool is a Windows application for managing data that has been uploaded to the Ranking server using the DWC's General-Purpose Ranking library. The tool can be used to browse, get, and delete data.

Figure 3-1 Organization



3.2 File Structure

The General-Purpose Ranking Administration Tool is composed of the following files:

- `DWCRankingAdmin.exe` Application executable file
- `DWCRankingAdmin.exe.config` Application settings file

3.3 Execution Environment

3.3.1 Install .NET Framework

In order use the General-Purpose Ranking Administration Tool, the Microsoft .NET Framework version 2.0 or later must be installed on the computer where the tool will run.

3.3.2 Install the Settings File

You need an encryption key so that the General-Purpose Ranking Administration Tool can exchange encrypted communications with the Ranking server.

The Settings file is the file `admin_setting.txt` distributed by Nintendo. Place this file in the same folder as `DWCRankingAdmin.exe`.

3.3.3 Configure Communications

Since the General-Purpose Ranking Administration Tool communicates using the Internet, you need to run the tool on a computer that is connected to the Internet.

The Internet Explorer settings are used as the communications settings by the tool.

3.4 Communications Load on the Ranking Server

The General-Purpose Ranking Administration Tool can place a very significant load on the Ranking server because it has the ability to gather large amounts of information at once. The persons using the tool should manage projects individually. Only a small number of people should use the tool at the same time.

To reduce the load on the server, data access is limited to a maximum of 100 entries for each time data is displayed.

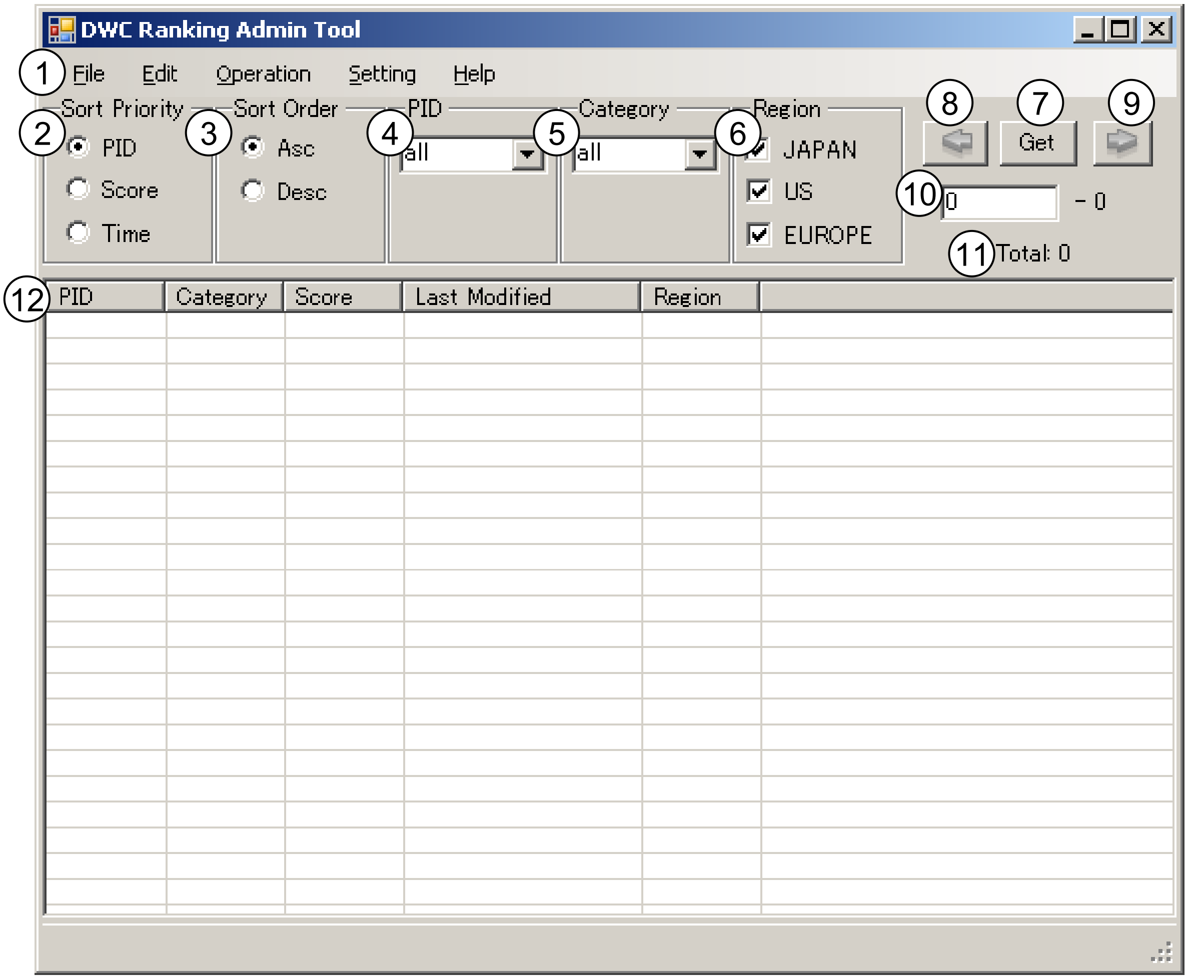
3.5 Starting the General-Purpose Ranking Administration Tool

Start the General-Purpose Ranking Administration Tool by executing the file `DWCRankingAdmin.exe`.

3.6 Screen Structure

Figure 3-2 shows the layout of the main window of the DWC Ranking Administration Tool.

Figure 3-2 DWC Ranking Administration Tool Main Window



When you start the General-Purpose Ranking Administration Tool, a main window like the one in Figure 3-2 is displayed. This main window displays data and specifies conditions. The various features in the different parts of the window are explained below.

①. This is the application menu. The various features of this menu are explained below.

②. Specifies the way to sort the data for display.

- PID	Sorts based on the order of the GS profile IDs.
- Score	Sorts based on the order of the scores.
- Time	Sorts based on the order of the latest update.

③. Specifies the sort order.

- Asc	Sorts in ascending order.
- Desc	Sorts in descending order.

④. Specifies the PID for the data to display. Enter a decimal number or specify "all" to show data for all PIDs.

⑤. Specifies the category ID for the data to display. Enter a decimal number or specify "all" to show data for all category IDs.

⑥. Specifies the region(s) for the data to display. Each region can be separately turned on and off.

- JAPAN	Japan
- US	North America
- EUROPE	Europe

⑦. Press this button to display a data list according to the conditions that have been specified.

⑧. The 'previous' button. This button is enabled when data has been displayed using an offset greater than 0. Press this button to display data using the offset that results from subtracting 100 from the current offset value.

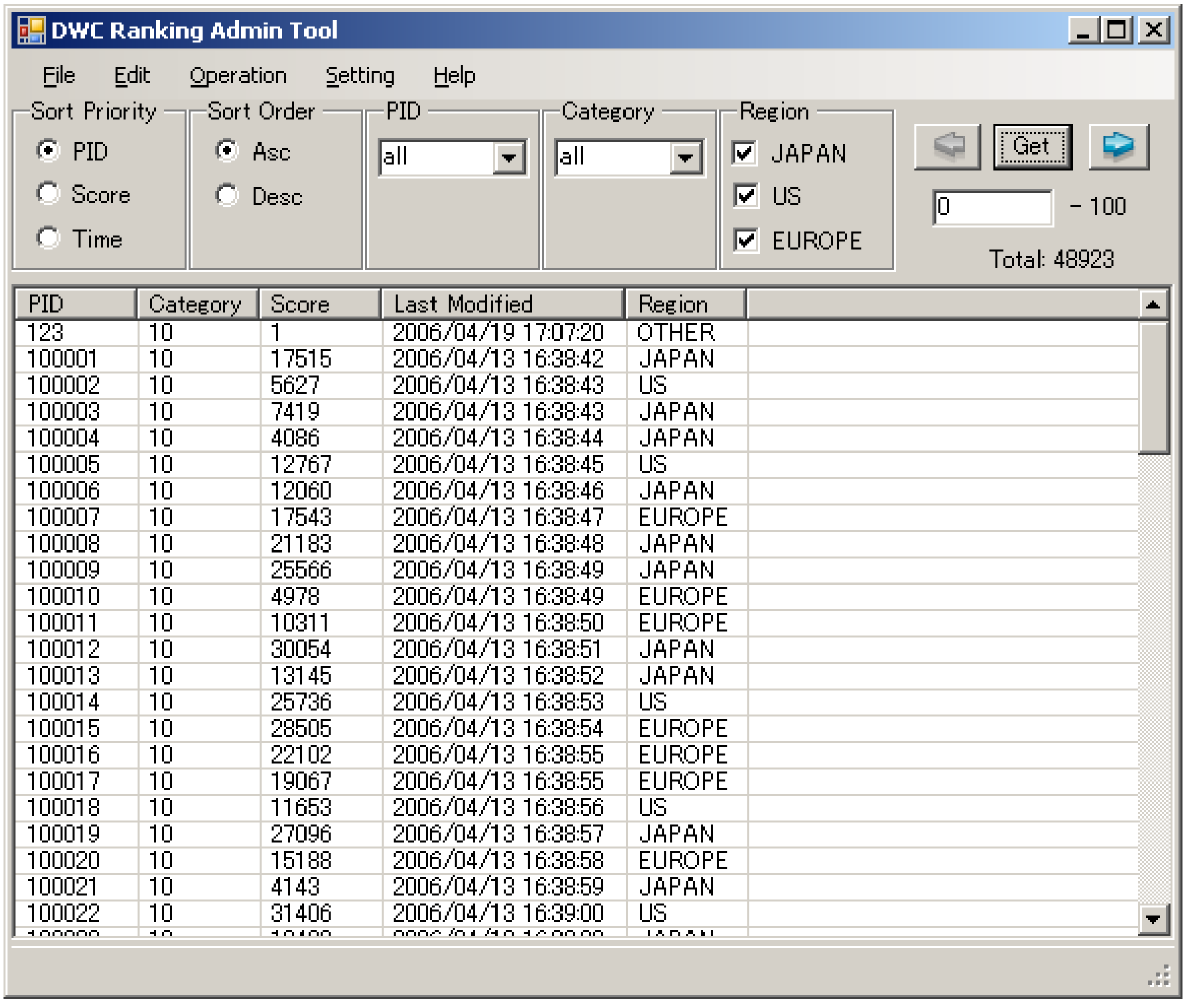
⑨. The 'next' button. This button is enabled when there is additional data. Press this button to display data using the offset that results from adding 100 to the current offset value.

⑩. Use this textbox to enter an offset value for the display of data. The offset serves as the starting point for the display of 100 sets of data, starting in order from the specified offset value.

⑪. Shows the number of sets of data that meet the specified conditions. The value shown here gets updated when data is displayed.

⑫. The data list. The data retrieved is shown in the form of a list here. An example is shown in Figure 3-3.

Figure 3-3 Example of Displayed Data



Sort Priority: ● PID ○ Score ○ Time

Sort Order: ● Asc ○ Desc

PID: all

Category: all

Region: ☑ JAPAN ☑ US ☑ EUROPE

0 - 100

Total: 48923

PID	Category	Score	Last Modified	Region
123	10	1	2006/04/19 17:07:20	OTHER
100001	10	17515	2006/04/13 16:38:42	JAPAN
100002	10	5627	2006/04/13 16:38:43	US
100003	10	7419	2006/04/13 16:38:43	JAPAN
100004	10	4086	2006/04/13 16:38:44	JAPAN
100005	10	12767	2006/04/13 16:38:45	US
100006	10	12060	2006/04/13 16:38:46	JAPAN
100007	10	17543	2006/04/13 16:38:47	EUROPE
100008	10	21183	2006/04/13 16:38:48	JAPAN
100009	10	25566	2006/04/13 16:38:49	JAPAN
100010	10	4978	2006/04/13 16:38:49	EUROPE
100011	10	10311	2006/04/13 16:38:50	EUROPE
100012	10	30054	2006/04/13 16:38:51	JAPAN
100013	10	13145	2006/04/13 16:38:52	JAPAN
100014	10	25736	2006/04/13 16:38:53	US
100015	10	28505	2006/04/13 16:38:54	EUROPE
100016	10	22102	2006/04/13 16:38:55	EUROPE
100017	10	19067	2006/04/13 16:38:55	EUROPE
100018	10	11653	2006/04/13 16:38:56	US
100019	10	27096	2006/04/13 16:38:57	JAPAN
100020	10	15188	2006/04/13 16:38:58	EUROPE
100021	10	4143	2006/04/13 16:38:59	JAPAN
100022	10	31406	2006/04/13 16:39:00	US

3.7 Menus

This section describes the various menus available in the application.

3.7.1 File Menu

Exit	Exits the application.

3.7.2 Edit Menu

Copy	Copies selected data from the data list to the clipboard.
Select All	Selects all data on display in the data list

3.7.3 Operation Menu

get CSV Opens the **Get CSV** dialog box in order to get a CSV file. See section 3.8.1.

get UserData Opens the **Get UserData** dialog box to get the user-defined data. See section 3.8.2.

Delete Entry Opens the **Delete Entry** dialog box to delete data. (See 3.8.3)

3.7.4 Setting Menu

use UTC When enabled, time is shown using UTC (Coordinated Universal Time).

Public Server When enabled, the connection is made to the production Ranking server.

3.7.5 Help Menu

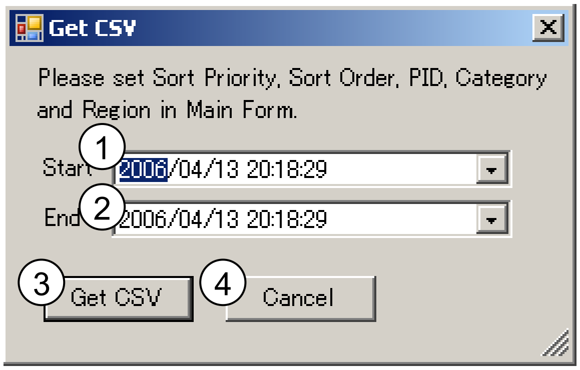
Version Displays version information.

3.8 Dialog Boxes

3.8.1 Get CSV Dialog Box

The Get CSV dialog box is used to get data in CSV format. The conditions for the sort order, the PID, the category ID, and the region as set in the Main Form are followed here when getting CSV files.

Figure 3-4 Get CSV dialog box



Please set Sort Priority, Sort Order, PID, Category and Region in Main Form.

Start 2006/04/13 20:18:29

End 2006/04/13 20:18:29

The Get CSV dialog box also has these additional features (see Figure 3-4):

①. Gets data starting from the date and time specified.

②. Gets data up until the date and time specified.

③. Starts the process of getting the CSV file.

④. Cancels the process. The dialog box is closed and no CSV file is retrieved.

The **Save File** dialog box opens when the process of getting the file has completed. Specify where to save the file in this dialog box.

The retrieved CSV file has one entry per line. Each entry has the following format:

100000 [tab] 10 [tab] 4 [tab] Thu Apr 13 00:38:41 PDT 2006 [tab] 5758 [tab] dGVzdCBkYXRhAA==

The tab serves as the delimiter. From left to right, the data elements included in the entry are the PID, the category ID, the region code, the time of the last update, the score, and the user-defined data (base64 encoded).

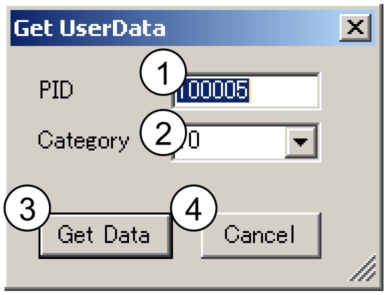
Up to 5,000 entries can be retrieved in a single CSV file. A warning is displayed if this limit has been exceeded. If this occurs, change the specified conditions to reduce the number of entries in the file (such as specifying a shorter time period) then retrieve the CSV file again.

It can take longer than a minute to retrieve the CSV file if the file contains a large number of entries. Since retrieving the CVS file is a processing-intensive task, you should only do this once every 30 minutes or so.

3.8.2 The Get UserData Dialog Box

The Get User Data dialog box is used to get user-defined data. The data is saved to a file in binary format.

Figure 3-5 Get UserData Dialog Box



The dialog box has the following parts (see Figure 3-5):

①. Specifies the PID for the data to get. If data has been selected from the data list, the PID of the selected data is already filled in when the **Get UserData** dialog box opens.

②. Specifies the category ID for the data to get. If data have been selected from the data list, the category ID of the selected data is already filled in when the **Get UserData** dialog box opens. Note that "all" cannot be specified.

③. Starts the process of getting the user-defined data file.

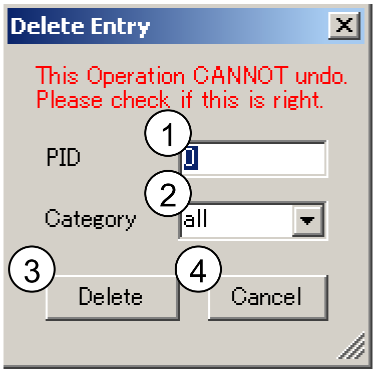
④. Cancels the process of getting the file and returns to the Main Form.

The Save File dialog opens when the process of getting the file has completed. Specify where to save the file in this dialog box.

3.8.3 The Delete Entry Dialog Box

The Delete Entry dialog box is used to delete data. Note that once data has been deleted, it cannot be recovered.

Figure 3-6 Delete Entry Dialog Box



Delete Entry

This Operation CANNOT undo.
Please check if this is right.

PID: 0

Category: all

The dialog box has the following parts (see Figure 3-6):

①. Specifies the PID for the data to delete. If data has been selected from the data list, the PID of the selected data is already entered in this box when the dialog opens.

②. Specifies the category ID for the data to delete. If data has been selected from the data list, the category ID of the selected data is already entered in this box when the dialog opens. If you specify “all,” entries with all the category IDs will be deleted.

③. Starts the process of deleting the data.

④. Cancels the process and returns to the main form.

The data list is automatically updated when data is deleted.

3.9 Shortcut Keys

The following shortcut keys can be used with the General-Purpose Ranking Administration Tool:

CTRL+C	Copies the selected data from the data list to the clipboard.
CTRL+A	Selects all data displayed in the data list.
CTRL+S	Opens the **Get CSV** dialog box in order to get a CSV file.
CTRL+U	Opens the **Get UserData** dialog box in order to get user-defined data.
CTRL+D	Opens the **Delete Entry** dialog box in order to delete data.

4 Web Service Development

4.1 About Web Services

The General-Purpose Ranking library has an interface for retrieving collected data via the Internet. You can utilize this feature to provide various services, such as displaying rankings on a game's homepage.

4.2 How to Use

This section describes how to use this feature.

4.2.1 How to Access

Access the addresses shown below to get General-Purpose Ranking data via the Internet. You can obtain a tab-delimited CSV file as the HTTP response.

- To access the production server:
http://gamestats.gs.nintendowifi.net/*[game name]*/web/admin/getcsv.asp

- To access the development server:
http://sdkdev.gamespy.com/games/*[game name]*/web/admin/getcsv.asp

Note: *[game name]* takes the unique character string assigned to each game title. It is included in the design statement that gets returned to you.

4.2.2 About Security

For security reasons, there are restrictions on the IP addresses allowed to access the General-Purpose Ranking Web Service. If you are going use the Web service, be sure to inform Nintendo of the global IP address from which you will access the service.

There are no restrictions on access to the development server.

4.2.3 About the Data Format

Data obtained from the General-Purpose Ranking Web Service is in the format of a tab-delimited CSV file, with one entry displayed on each line.

In order, the data elements included in each entry are the GS Profile ID, the category ID, the region, the time of the last update, the score, and the user-defined data (base64 encoded).

4.2.4 About Server Loads

Use of the Web Service places a very large load on the Ranking server. For this reason, only get data at a frequency of around once every 30 minutes, set an appropriate value for the *since* parameter when getting data, and avoid getting the same data again.

4.2.5 get Parameters

When requesting data from the Ranking server, you can specify parameters that define how to get the data. These "get" parameters are attached to the end of the URL in the form of a character string, as shown in the example below:

http://…/getcsv.asp?[*parameter name 1*]=[*value 1*]&[*parameter name 2*]=[*value 2*]&...

Below is a list of the parameters and values that can be specified. You do not need to specify all of the parameters, as when a parameter is not specified its default value is used.

sort — Specifies how to sort the data. The default is 0.

0: The obtained data is sorted in ascending order based on score.
1: The obtained data is sorted in descending order based on score.

region — Specifies which region(s) of data to get data for. The values are OR'ed together. The default value is 7 (all regions)

1: Japan
2: North America
4: Europe

pid — Specifies the PID for the data to be retrieved. Takes a numerical value. The default is to get the data for all PIDs.

category — Specifies the category ID for the data to be retrieved. Takes a numerical value. The default is to get the data for all category IDs.

limit — Specifies the maximum number of data sets to retrieve. Takes a numerical value between 1 and 5000. The default is 100.

since — When this is specified, only data updated since the specified date and time is retrieved.

The default is to retrieve everything without any date/time limit.
The parameter follows this format: year-month-day-hour-minute-second.
The parameter should be specified using UTC (Coordinated Universal Time)

Below is an example of how the URL would be specified to obtain a maximum of 1,000 sets of data updated since 13:00 hours of April 1, 2006 with the category ID of 10:

http://…/getcsv.asp?category=10&limit=1000&since=2006-4-1-13-0-0