liblocation Overview

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Document serial number: 000004892117

Table of Contents

1 Library Overview

Purpose and Features

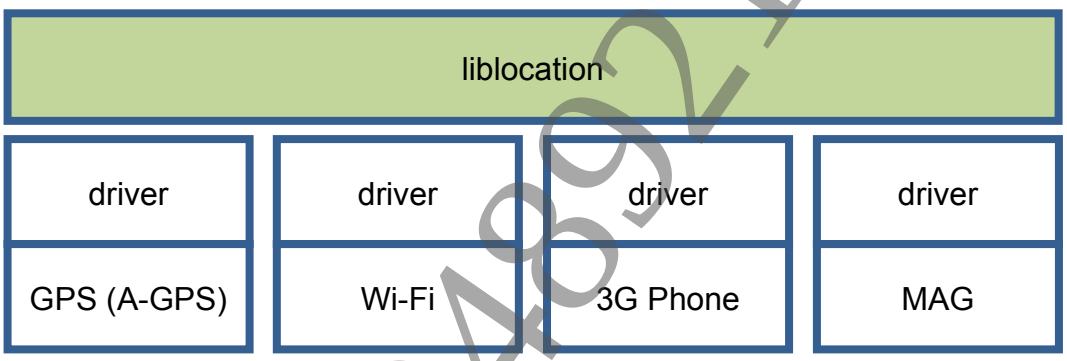
liblocation is a library that provides geographic information (such as, the current location and direction) from a location calculating device using signals of GPS (A-GPS),Wi-Fi, etc. Using liblocation, applications can obtain the best available geographic information.

Main Functions

The following main functions are offered by the liblocation library.

- Obtainment of current location and direction of local device using hardware
- Continual location calculation using a callback function of the above functions
- Dialog display and prompting user to allow use of location information
- GPS emulation using a defined format file

liblocation can use multiple settings simultaneously for one process.



Location Information

liblocation identifies the latitude, longitude, and altitude using signals of GPS (A-GPS), Wi-Fi and mobile communication and outputs them as location information. Each location calculating device has a different processing speed, accuracy, and area where the device can be used; location information is selected from the most suitable device according to the situation.

Note that the location calculating device installed on a unit may vary depending on the model, and consequently the location calculation characteristics are also different. Careful consideration must be given to this point during development. Particularly, GPS (A-GPS) and mobile communication calculations are not implemented in the Development Kit and Testing Kit, and therefore it is not possible to check them during application development and debugging. For this reason, it is recommended to check the behavior of the system software, etc. for reference using a retail unit. The characteristics of each location calculating device are as follows.

GPS (A-GPS)

GPS (A-GPS) is useful in a wide area, but it is not suitable to be used indoors. Also, its location calculation time tends to be longer; it may sometimes take several minutes until an effective value is obtained. Although network is not necessarily required, the processing speed varies significantly depending on the network connection status or surrounding environments. Timeout may often occur when the value cannot be obtained with one-shot calculation through a function such as `sceLocationGetLocationWithTimeout()`. In the case that location calculation using GPS is mainly assumed, consider performing timeout adjustment through `sceLocationGetLocationWithTimeout()` or using the callback type calculation. This location calculating device cannot be used for Wi-Fi models.

Wi-Fi

Normally, Wi-Fi cannot be used outside its service area, as well as in areas where no Wi-Fi access point exists. The location calculation speed is extremely fast in areas where processing is possible, taking only several seconds to complete in most cases. It can also be used indoors. Network communication is required, but not always performing. This location calculating device can be used for all models.

The service area where the location calculation is possible can be checked via the following website.

- http://www.skyhookwireless.com/location-technology/coverage.php

(The above reference destination has been confirmed as of July 11, 2014. Note that pages may have been subsequently moved or its contents modified.)

Mobile Communication

Mobile communication covers a wider area than Wi-Fi, and it is also suitable to be used indoors. Network communication is always required during location calculation. This location calculating device cannot be used for Wi-Fi models.

Direction Information

liblocation obtains direction information using libmotion with a magnetometer sensor and accelerometer, etc., and outputs the magnetic north after correcting it to the true north by adding the declination data according to the current position that has been calculated.

For the basic characteristics and usage methods of a magnetometer sensor and other sensors, refer to the "libmotion Overview" document.

Embedding into Program

Include liblocation.h in the source program. (Additionally, a number of header files are automatically included.)

Load also the PRX module in the program as follows.

```
if ( sceSysmoduleLoadModule(SCE_SYSMODULE_LOCATION_EXTENSION) != SCE_OK ) {
    // Error processing
}
```

When building the program, link libSceLocation_stub.a and libSceLocationExtension_stub.a.

Sample Programs

The following files are provided as sample programs that use liblocation for reference purposes.

sample_code/input_output_devices/api_liblocation/get_location_simple/

This sample exemplifies basic usage to obtain the current location.

sample_code/input_output_devices/api_liblocation/get_location_callback/

This sample periodically obtains current location events from liblocation.

sample_code/input_output_devices/api_liblocation/get_location_wifihistory/

This sample uses the stored Wi-Fi access points feature.

sample_code/input_output_devices/api_liblocation/error_handling/

This sample shows how to handle typical errors that may occur during liblocation use. It also shows the location information usage permission obtainment routine in cases where `SCE_LOCATION_MODE_CONFIRM_AUTO` is not set in `sceLocationInit()`'s *mode*.

sample_code/input_output_devices/api_liblocation/cancel_get_location/

This sample shows how to cancel `sceLocationGetLocation()`.

sample_code/input_output_devices/api_liblocation/get_heading_simple/

This sample exemplifies basic usage to obtain directions.

sample_code/input_output_devices/api_liblocation/set_emulation_file/

This sample exemplifies transitioning liblocation to the GPS emulation mode. In addition, the items required for carrying out location calculation, such as, error handling and dialog displays, are presented.

sample_code/input_output_devices/api_liblocation/set_thread_parameter/

This sample changes the thread priority of liblocation. liblocation callbacks are set to be received when an application's thread priority is raised.

sample_code/input_output_devices/api_liblocation/heading_calibration/

This sample shows the dialog for providing assistance on calibration of magnetometer sensors. It also presents the method for periodically obtaining directions.

2 Usage Procedure

Setting the Param File

Mark the **Use liblocation in order to get location** checkbox of the App Setting button through Param File Editor so that applications can obtain location information. For details, refer to the "App Setting Button" section in the "Param File Editor User's Guide" document. In addition, refer to the "Measures during Development" section of the "3 User Permission Mechanism" chapter in this document during the development of applications. This setting is not required when only the direction needs to be obtained.

Basic Usage Procedure

This section explains the basic procedure for obtaining current location information. The usage procedure for obtaining direction information is the same.

(1) Load liblocation PRX module

Using `sceSysmoduleLoadModule()`, load the liblocation PRX module. Specify `SCE_SYSMODULE_LOCATION_EXTENSION` for the PRX ID.

(2) Initialize liblocation

After loading PRX of liblocation, execute the `sceLocationInit()` function only once. Specify the location calculation method for the argument. Use "`SCE_LOCATION_MODE_V3 | SCE_LOCATION_MODE_CONFIRM_AUTO`" for general usage.

Then liblocation can be used through `sceLocationOpen()`. Also, get a handler for use during liblocation access.

To use liblocation with different settings simultaneously, specify the settings separately using separate handlers obtained through multiple `sceLocationOpen()` function accesses.

(3) Get location information

Get location information through `sceLocationGetLocation()`. The latitude, longitude, altitude, accuracy, travel direction, travel speed, etc., are saved to the `SceLocationLocationInfo` structure.

(4) Support of user usage permission

Set **Parental Controls -> Features -> Location Data** to **Not Blocked** and select the **Settings -> Location Data -> Use Location Data** check box

(5) Close liblocation

Execute `sceLocationClose()` and then execute `sceLocationTerm()` to close liblocation.

(6) Unload the liblocation PRX module

Unload the liblocation module with `sceSysmoduleUnloadModule()`.

Continual Processing Procedure Using a Callback

liblocation allows continual processing through callback registration. An event occurs when the distance - set upon the previous registration of a callback function - is reached. The continual processing procedure for obtaining location information is described below (processing procedure for direction information is the same).

(1) Load the liblocation PRX module and initialize liblocation

The procedure here is the same as the "Basic Usage Procedure" section.

(2) Set and start the callback function

Register and start the callback function with `sceLocationStartLocationCallback()`. The main arguments to be set are the distance to cover in order to trigger the next callback and the callback function address. The next callback occurs when the distance from the point where the last callback occurred reaches the specified distance. If the set distance is 0, callback occurs continually in accordance with internal sampling.

The latitude, longitude, altitude, accuracy, travel direction, travel speed, etc., can be obtained from the argument `SceLocationLocationInfo` structure within the callback function.

(3) Support of user usage permission

The procedure here is the same as the "Basic Usage Procedure" section.

(4) Stop callback function

Stop the callback function with `sceLocationStopLocationCallback()`.

(5) Close liblocation and unload the liblocation PRX module.

The procedure here is the same as the "Basic Usage Procedure" section.

Processing Procedure Using Stored Wi-Fi Access Points

When location using Wi-Fi cannot be calculated because network connection is not possible, this feature can be used to store the surrounding Wi-Fi access points to later calculate the location when network connection becomes possible. Although the current location cannot be obtained with this feature, it is useful for obtaining past locations.

(1) Load the liblocation PRX module

Using `sceSysmoduleLoadModule()`, load the liblocation PRX module. Specify `SCE_SYSMODULE_LOCATION_EXTENSION` for the PRX ID.

(2) Initialize liblocation

The procedure here is the same as the "Basic Usage Procedure" section.

(3) Get current location information

Get location information with `sceLocationGetLocationWifiHistory()`. When calculation succeeds, the latitude and longitude, altitude, horizontal error, travel direction, and travel speed can be obtained as the `SceLocationLocationInfo` structure. If the return value is `SCE_LOCATION_INFO_UNDETERMINED_LOCATION`, `SceLocationLocationInfo.timestamp` will be stored to later access the stored Wi-Fi access points.

(4) Support of user usage permission

The procedure here is the same as the "Basic Usage Procedure" section.

(5) Obtain stored Wi-Fi access points

When calculation in step (3) fails and the return value is `SCE_LOCATION_INFO_UNDETERMINED_LOCATION`, use *`timestamp`* obtained at another time and execute `sceLocationQueryLocationWifiHistory()`. If network connection is available, a query will be made to the Wi-Fi location database for the Wi-Fi calculation element obtained in step (3) and if the element exists in the Wi-Fi location database, the result of the location calculation will be returned.

(6) Close liblocation and unload the liblocation PRX module

The procedure here is the same as the "Basic Usage Procedure" section.

Limitations

The following limitations apply to the use of liblocation.

Limitation on the Number of Simultaneous Executions of sceLocationOpen()

liblocation can perform up to 8 `sceLocationOpen()` executions, including those executed by other processes. If this limit is exceeded during `sceLocationOpen()` executions, `SCE_LOCATION_ERROR_TOO_MANY_HANDLES` is returned and liblocation can no longer be used.

Limitation on the Location Calculation Method

liblocation allows specification of the location calculation method with *`lmethod`* and *`hmethod`* during `sceLocationOpen()` execution, but location may not necessarily be calculated with the specified method. This is because the location calculation method dynamically changes as the result of requests from other applications or for internal routine reasons. Even in such a case, the calculation method will be changed to one that can maintain current performance. However, in the case of systems that do not have a mobile communication function (this includes the Development Kit and Testing Kit), calculation using GPS or 3G cell towers, etc., is not possible. Thus, even if a location calculation method entailing one of these is specified, location calculation by Wi-Fi will be selected.

Using the `sceLocationGetMethod()` function, it is possible to obtain the location calculation method that is currently being used.

This limitation does not apply to the direction calculation method, and the direction value is obtained according to the *`hmethod`* that is set at the time of `sceLocationOpen()` execution.

Limitations on the Stored Wi-Fi Access Point Feature

Although the stored Wi-Fi access point feature can store several hundred data, there is no guarantee that a certain volume can always be used by each application because this storage area is shared by all applications. If data exceeds the upper limit of the storage volume, old data will be replaced by new data regardless of whether data has been read/unread by `sceLocationQueryLocationWifiHistory()`.

True North Direction May Not Be Indicated

Direction calculation is done by adding the declination data obtained from the location information to the value of the built-in magnetometer sensor in order to output the true north. It may not be possible to calculate the declination data until new location information is obtained, and therefore the old declination data obtained previously may be added (in which case magnetic north is indicated at initial start-up or after initialization). However, because changes in the declination data owing to shifts in location are quite small in relation to the accuracy of the magnetometer sensor, they can be practically ignored unless major shifts in location of 100 kilometers or more occur.

Direction May Not Be Obtained or Value May Be Inaccurate Immediately After the Execution of sceLocationOpen()

When other applications do not use a magnetometer sensor through direction calculation of liblocation or through libmotion, a magnetometer sensor starts to operate using `sceLocationOpen()`. Immediately after the start, the magnetometer sensor is being initialized or is in unstable state because the automatic calibration is not performed yet, and therefore an error may return with no value obtained or the obtained value may not be accurate. It is recommended to obtain a value at least 500 milliseconds to 1 second after the execution of `sceLocationOpen()`.

Obtainment of Location Information or Direction Information May Result in an Error Due to Power Configuration Control

Depending on the power configuration changing the GPU clock frequency of PlayStation®Vita, the Wi-Fi, mobile communication functions, motion sensor, etc., may not be usable. In this case, the obtainment of current location information and direction information may result in an error.

3 User Permission Mechanism

Overview

The information indicating the current location belongs to the user's personal information and applications cannot use this information without the end user's permission. liblocation recognizes the user's usage permission status for each application and outputs location information only to those applications that have received permission. Selection of whether or not to obtain location and selection through parental control are also done at the overall system level. For this reason, it is necessary for applications to implement the user permission mechanism and obtain the end user's permission to use location information.

Applications may implement the user permission mechanism in one of the following ways:

- Have liblocation obtain permission for use automatically
  We normally recommend this method, however, the dialog for obtaining permission to use location information may be displayed on the screen when a one-shot calculation function such as `sceLocationGetLocation()` is executed, or during continual calculation by `sceLocationStartLocationCallback()`.
- Implement a location information usage permission obtainment routine in the application
  If you wish to prevent the dialog for obtaining permission to use location information from appearing at unwanted times, you can use this method to control how it is displayed.

If an application does not execute the user permission mechanism, the application's name will not be displayed under **Settings -> Location Data**, and the application will not be able to use location information.

The user permission mechanism does do not affect direction information obtainment.

Automatic Obtainment of Permission to Use Location Information with liblocation

By specifying `SCE_LOCATION_MODE_CONFIRM_AUTO` in `sceLocationInit()`'s *mode*, the system will, as necessary, execute the user permission mechanism automatically and display the dialog for obtaining permission to use location information. During one-shot calculations with `sceLocationGetLocationWithTimeout()`, a dialog will be displayed while the function is being executed. The dialog will also be displayed before each callback during continual calculation with `sceLocationStartLocationCallback()`. The dialog will not always be displayed, but will be displayed only once at the time of the first calculation or after changing the conditions for displaying the dialog for obtaining permission to use location information when location information is used (see below). If the dialog is displayed but the user refuses to give his/her permission, or conditions have not been changed and the dialog is not displayed but permission is not given, the `SCE_LOCATION_INFO_DENIED_BY_USER` error will be stored in each calculation function's return value and result variable.

Even when the obtainment of permission to use location information is set to automatic, it is possible to obtain information on whether the use of location information is permitted or not with the function `sceLocationGetPermission()`.

Obtainment of Permission to Use Location Information by Applications

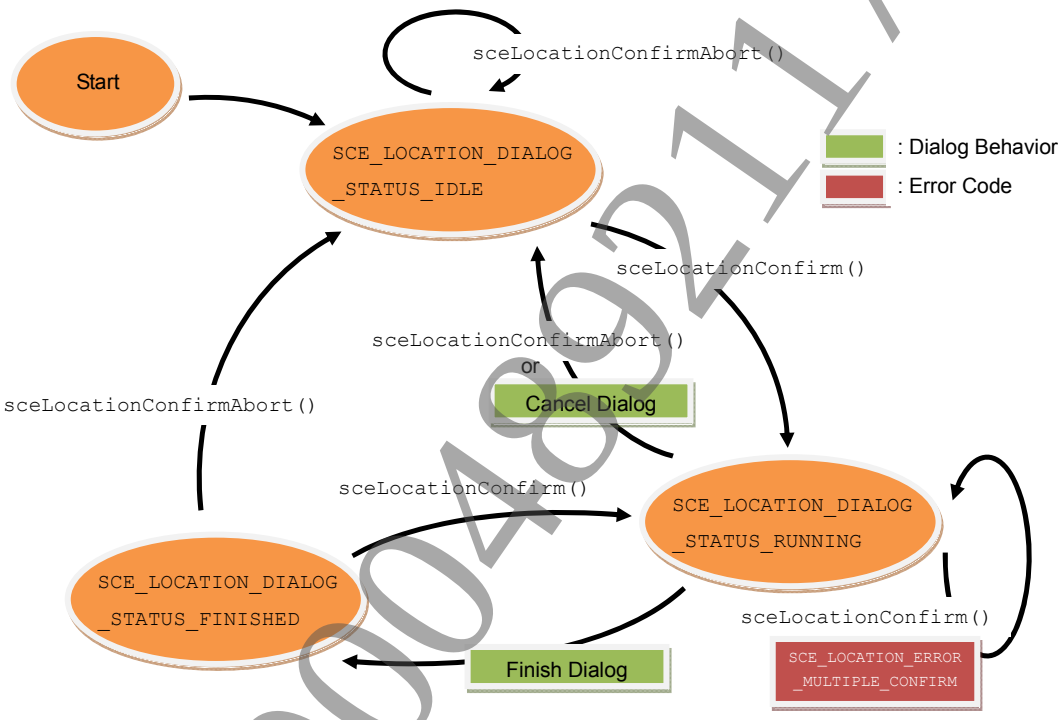
In cases where you wish to prevent the dialog for obtaining permission to use location information from being displayed at unwanted times, to limit the interval of dialog display to a given period of time, or to control the user permission mechanism in detail, it will be necessary for the application to implement the user permission mechanism by executing the relevant functions instead of setting the obtainment of permission to use location information to automatic.

Implementation Method

Specify SCE_LOCATION_MODE_CONFIRM_MANUAL in sceLocationInit()'s *mode*.

If, when obtainment of location information is attempted with either sceLocationGetLocation() or SceLocationLocationInfoCallback() of liblocation but location information cannot be obtained owing to the conditions for displaying the dialog for obtaining permission to use location information when location information is used (see below), the SCE_LOCATION_INFO_DENIED_BY_USER error is returned. When this error occurs, display a dialog with the sceLocationConfirm() function and prompt the user for location information usage permission. sceLocationConfirm() being a non-blocking function, it is necessary to wait until the dialog status is obtained with sceLocationConfirmGetStatus() and the dialog display ends. To have the application use the result, obtain the result with sceLocationConfirmGetResult() when the status is SCE_LOCATION_DIALOG_STATUS_FINISHED. The status transition diagram is shown below.

Figure 1 State Transitions of the Dialog for Obtaining Permission to Use Location Information



Restricting the Display of the Dialog for Obtaining Permission to Use Location Information

If the user denies usage permission through the dialog when liblocation tries to obtain location information, SCE_LOCATION_INFO_DENIED_BY_USER is returned every time sceLocationGetLocation() or SceLocationLocationInfoCallback() is executed, but having to display the permission dialog every time this value is returned can be extremely cumbersome for the user. It is therefore recommended to have the system memorize the permission status previously obtained and not display this dialog if conditions are the same. However, this does not apply when the permission status has been changed through settings, etc.

In liblocation, it is possible to check if there are any changes in the usage permission status between the last application termination and the next start-up, through the *updated* member variable of the SceLocationPermissionInfo structure obtained with sceLocationGetPermission(). When the return value of sceLocationGetLocation() or *result* of SceLocationLocationInfoCallback() is SCE_LOCATION_INFO_DENIED_BY_USER, check the *updated* member variable with sceLocationGetPermission(), and if the variable is SCE_LOCATION_PERMISSION_UPDATED, then execute sceLocationConfirm().

Do not - on your own - record the member variables of `SceLocationPermissionInfo` other than *`updated`* and do not also record whether the error code returned when obtaining the value is `SCE_LOCATION_INFO_DENIED_BY_USER` in order to use these records to control the dialog display. Such operation may generate a contradiction upon initializing the system setting, or may cause a problem when a future expansion of the permission dialog display function cannot be handled.

Migration from Versions Preceding SDK 2.100

In accordance with TRC [R3145] requirements, when implementing the obtainment of permission to use location information by an application on SDK versions preceding 2.100, it was necessary to check permission information with `sceLocationGetPermission()` and to obtain permission to use location information with `sceLocationConfirm()` immediately after application start-up.

This requirement was canceled from SDK 2.100 onwards, making it no longer necessary to execute `sceLocationGetPermission()` immediately after application start-up. Instead, processing equivalent to `sceLocationConfirm()` come to be performed automatically in the system when executing `sceLocationOpen()`. For this reason, it is possible that the dialog may be displayed while `sceLocationOpen()` is being executed, and for `sceLocationOpen()` to be blocked during that time. Therefore, if an application's development is in progress since before SDK 2.000 and then migrates to SDK 2.100 or later, assume that `sceLocationOpen()` will be blocked when used.

Note
When using the automatic obtainment of permission to use location information by liblocation on SDK 2.100 or later, too, `sceLocationOpen()` will be executed as in the past, without processing equivalent to `sceLocationConfirm()` being carried out within `sceLocationOpen()`.

Display Conditions for the Dialog for Obtaining Permission to Use Location Information when Location Information is Used

Output is restricted in the following cases when location information obtainment is attempted through liblocation. The behavior when `sceLocationConfirm()` is executed is described for each case.

- When there are restrictions per country/region
  A special dialog will be displayed, and the restrictions cannot be canceled
- When restriction is caused by parental control in system settings
  The dialog indicating parental control is just displayed, and parental control cannot be canceled
- When location information is not obtained following installation of the target application
  The [Yes] and [No] buttons are displayed and whether or not to use location information is selected
- When the location data items in the system settings are off
  The [Yes] and [No] buttons are displayed and whether or not to display the location data items in the system settings is selected
- When the location data usage permission for individual applications in the system settings is off
  The [Yes] and [No] buttons are displayed and whether or not to use location information is selected

The priority level of these restrictions follows that of the items listed above, with the highest restriction priority for the highest item, and likewise for the dialog display priority.

Measures during Development

Because, at application startup from the development host computer or from APP_HOME, the application is not registered to the system and the param file (param.sfo) does not exist or is not invalid, the usage permission status cannot be saved to the system. Therefore, location information cannot be obtained under these conditions, and during debugging, the various usage conditions must be freely reproduced.

In cases such as those described above or when **Settings -> ★ Debug Settings -> Location Data -> Emulate Permission** has been set to **On**, in liblocation the user usage permission status can be specified by the value of **Settings -> ★ Debug Settings -> Location Data -> Permission Status**, thereby enabling

development and debugging work. For example, to forcibly use location information, set **Emulate Permission** to **On** and **Permission Status** to **Allow** (Additionally, setting **Parental Controls -> Features -> Location Data** to **Not Blocked** and selecting the **Settings -> Location Data -> Use Location Data** check box are required).

Emulate Permission

When this setting is set to **On**, the usage permission status is emulated using the value of Permission Status. When it is **Off**, the behavior is the same as in the product phase. Regardless of this setting, when the application has been started up from the development host computer or from APP_HOME, the behavior is the same as when the setting is **On**.

Permission Status

When the application has been started up from the development host computer or from APP_HOME, or **Emulate Permission** is set to **On**, **Permission Status** functions and the following four statuses can be set. The status may change when `sceLocationConfirm()` is executed.

- Disable
  This status is set for applications that do not obtain location information. In this status, location information cannot be obtained regardless of system setting values.
- Not Yet Accessed
  This status is set when, following the installation of an application, that application has not obtained location information even once.
- Deny
  This status is set when the user selects **No** for location data items for each application in the system settings or the dialog displayed by `sceLocationConfirm()` of liblocation.
- Allow
  This status is set when the user selects **Yes** for the location data item for each application in system settings or the dialog displayed by `sceLocationConfirm()` of liblocation.

Restrictions

If **Emulate Permission** is set to **On**, behavior when the check boxes for individual applications under **Settings -> Location Data** are operated will not be guaranteed.

Relationship with Display of Other Dialogs

If another dialog was already displayed before the usage permission dialog is displayed, the response of the system differs according to the type of dialog.

If the permission dialog of liblocation is displayed while the Message Dialog (executed by the application) is already displayed, the permission dialog is displayed overlapping the already-displayed Message Dialog. Also, control of the touch service, etc., is also transferred to the permission dialog.

If display of the usage permission dialog of liblocation is attempted while the Wi-Fi network connection dialog or a dialog such as IME (displayed by the system) is already being displayed, the dialog that was displayed first is closed and the usage permission dialog is displayed. Thereafter, the dialog that was originally displayed will be displayed again.

4 GPS Emulation Function

Overview

In normal development environments, there are many cases where location calculation is not possible owing to equipment conditions, or where the calculation results are difficult to control, so emulation using calculation result history files is provided for development. Each history file is saved as a GPS emulation file in a memory card, and the file contents are periodically output when liblocation is implemented in the GPS emulation mode. The GPS emulation file can be generated using the support tool (Track Creator) included in the SDK. For details, refer to the "Track Creator User's Guide" document.

GPS Emulation Setting

To set liblocation to the GPS emulation mode, specify `SCE_LOCATION_LMETHOD_GPS_EMULATION` as the argument of `sceLocationOpen()` in *`lmethod`* or before executing `sceLocationOpen()` set **Setting -> ★ Debug Settings -> Location Data -> Device Model** to **Emulation Model**.

GPS Emulation File Selection

GPS emulation starts up when the file selected with the `sceLocationSetGpsEmulationFile()` function of liblocation does exist when `sceLocationOpen()` is executed. If no file was selected, ux0:/data/gpsdata/gpsdata.nma is checked. To change the emulation file while emulation is in progress, after having all the applications that use liblocation execute `sceLocationClose()` to stop that calculation, select the desired file with `sceLocationSetGpsEmulationFile()`. This operation is common for the entire system, so that when an emulation file is selected for another application such as the sample program `set_emulation_file`, that selection is reflected to the local application. gpsdata.nma is located in the sample directory sample_code/input_output_devices/api_liblocation/get_location_callback. The GPS emulation function can be verified by copying this file to the directory ux0:/data/gpsdata/ of a memory card and then executing sample get_location_callback. If gpsdata.nma is located at host0:, file copying can be realized with "`psp2ctrl cp gpsdata.nma ux0:/data/gpsdata/`" after system startup.

GPS Emulation File Format

Each GPS emulation file forms one location information unit per message group consisting of $GPGGA, $GPGSA, $GPGSV, $GPRMC, and $GPVTG, according to the following format. Each message is expressed as one line in ASCII format and includes multiple data items. Each message begins with the Message ID, followed by the multiple data items delimited by a comma, and is closed by a final checksum.

The detailed message format is described below.

Format

- ASCII format
- message delimiter
  Line break (CR+LF). A line break is required also for the last message.
- message's initial character string
  Message ID (5-character character string beginning with $)
- message's terminating character string
  *Checksum (2-character HEX value prefixed by *. $ and * are not included in the checksum calculation)
- data delimiter
  CSV without space

Messages

The order of the messages cannot be changed. Items that are not needed for emulation have been omitted (even if described, they are ignored during the actual operation), but the number of commas prescribed by the format must be used.

The $GPGGA message is required as a minimum. In this case, altitude, accuracy, vaccuracy, direction and speed are not output.

The underlined parts below are required items.

$GPGGA, UTC Time, Latitude, N/S, Longitude, E/W, Position Fix Indicator, Satellites Used, HDOP, MSL Altitude, , Geoid Separation, , , * Checksum

- UTC Time
  The UTC time is denoted as hhmmss.sss, where hh equals hours, mm equals minutes, ss equals seconds, and .sss equals milliseconds.
- Latitude
  - The latitude expresses the north latitude/south latitude. The latitude format is ddmm.mmmm, where dd equals degrees, mm equal minutes, and .mmmm is decimal minutes. The geographical coordinate system is WGS 84.
  - Conversion from the degree notation is done as: Integer part x 100 + fractional part x 60.
- N/S
  This indicates the north latitude or the south latitude. A single character is used, N for North latitude and S for South latitude.
- Longitude
  - The longitude expresses the east longitude/west longitude. The longitude format is ddmm.mmmm, where dd equals degrees, mm equal minutes, and .mmmm is decimal minutes. The geographical coordinate system is WGS 84.
  - Conversion from the degree notation is done as: Integer part x 100 + fractional part x 60.
- E/W
  This indicates the east longitude or the west longitude. A single character is used, E for East longitude, and W for West longitude.
- Position Fix Indicator
  This indicates the GPS location calculation method.

Value	Description
0	Invalid
1	GPS SPS mode
2	DGPS SPS mode

- Satellites Used
  This indicates the number of satellites obtained. The range is from 0 to 12.
- HDOP
  Horizontal dilution of precision
- MSL Altitude
  Elevation above sea level
- Geoid Separation
  Height difference with the geoid height

$GPGSA, , Mode, Satellite Used1(S1), S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, , , * Checksum

- Mode
  This indicates the GPS location calculation method.

Value	Description
0	Invalid
1	2D mode (in case of 3 or fewer satellites)
2	3D mode (in case of 4 or more satellites)

- Satellite Used
  This indicates the ID of satellites obtained. The notation is fixed to twelve areas, including satellites that have not been obtained.

$GPGSV, Number of Message, Message Number, Satellite in View, Satellite ID, Elevation, Azimuth, SNR, ... * Checksum

One $GPGGA message contains the information of up to four satellites. Since three messages can be described for $GPGGA for one unit of location information, the information of up to twelve satellites can be stored in total. The information of one satellite consists of four data, namely Satellite ID, Elevation, Azimuth, and SNR, and up to four such groups can be stored, but if the number of satellites is less than four, the data area itself is deleted. Therefore, excluding Message ID and Checksum, from seven to nineteen areas are available as data areas of $GPGSV.

- Number of Message
  This indicates the number of $GPGSV messages.
- Message Number
  This indicates which number this message is among all $GPGSV messages.
- Satellite in View
  Number of satellites from which data is available (this is the total number of satellites in $GPGSV, not just those in the applicable messages)
- Satellite ID
  Satellite ID. ID described in Satellite Used of $GPGSA. The ID range is 1 to 32.
- Elevation
  Satellite elevation angle, in degrees. The range is 0 to 90.
- Azimuth
  Satellite azimuth, in degrees. The range is 0 to 359.
- SNR
  Signal-to-noise ratio of signal received from satellite, in dB. The range is 0 to 99.

$GPRMC, UTC Time, , , , , , Speed Over Ground, Course Over Ground, Date, , , * Checksum

- UTC Time
  The UTC time is denoted as hhmmss.sss, where hh equals hours, mm equals minutes, ss equals seconds, and .sss equals milliseconds.
- Speed Over Ground
  Travel speed, in knots. This is used if $GPVTG does not contain Speed. Normally this item is left blank.
- Course Over Ground
  Travel direction, in degrees. This is used if $GPVTG does not contain Course. Normally this item is left blank.
- Date
  Calculation date, expressed as ddmmyy.

$GPVTG, Course, , , , , , Speed, , , * Checksum

- Course
  Travel direction, in degrees.
- Speed
  Travel speed, in km/h.

Other items, such as GLL, VTG, and ZDA, are ignored.

Description Example

Required Minimum Description (Accuracy, Speed, Course, etc., Are Missing)

In this emulation function, the calculation information can be output if there is at least one $GPGGA message. If there is only one $GPGGA message, the calculation time does not increment and the same information is continually output at 1-second intervals.

```
$GPGGA,061837.000,3540.3017,N,13943.3386,E,1,0,,,,,,*7C
```

Adequate Description

In the following example, two calculation information items are described at 1-second intervals. Messages and data not described in the above specification are included, but these have no influence on the actual operation and need not be included.

```
$GPGGA,061837.000,3540.3017,N,13943.3386,E,1,04,6.4,-34.2,M,39.4,M,,0000*42
$GPGLL,3540.3017,N,13943.3386,E,061837.000,A,A*57
$GPGSA,A,3,24,07,17,28,,,,,,,,,6.5,6.4,1.0*3F
$GPGSV,3,1,11,24,62,314,32,20,62,057,,17,60,323,33,07,45,314,24*7E
$GPGSV,3,2,11,28,45,219,25,11,31,070,,04,25,273,,23,23,127,17*7C
$GPGSV,3,3,11,01,09,044,11,13,08,158,11,19,03,122,*4D
$GPRMC,061837.000,A,3540.3017,N,13943.3386,E,0.64,96.74,120106,,,A*5A
$GPVTG,96.74,T,,M,0.64,N,1.2,K,A*30
$GPGGA,061838.000,3540.3017,N,13943.3387,E,1,04,6.4,-34.6,M,39.4,M,,0000*48
$GPGLL,3540.3017,N,13943.3387,E,061838.000,A,A*59
$GPGSA,A,3,24,07,17,28,,,,,,,,,6.5,6.4,1.0*3F
$GPGSV,3,1,11,24,62,314,32,20,62,057,,17,60,323,33,07,45,314,24*7E
$GPGSV,3,2,11,28,45,219,25,11,31,070,,04,25,273,,23,23,127,18*73
$GPGSV,3,3,11,01,09,044,11,13,08,158,12,19,03,122,15*4A
$GPRMC,061838.000,A,3540.3017,N,13943.3387,E,0.57,100.59,120106,,,A*65
$GPVTG,100.59,T,,M,0.57,N,1.0,K,A*03
```

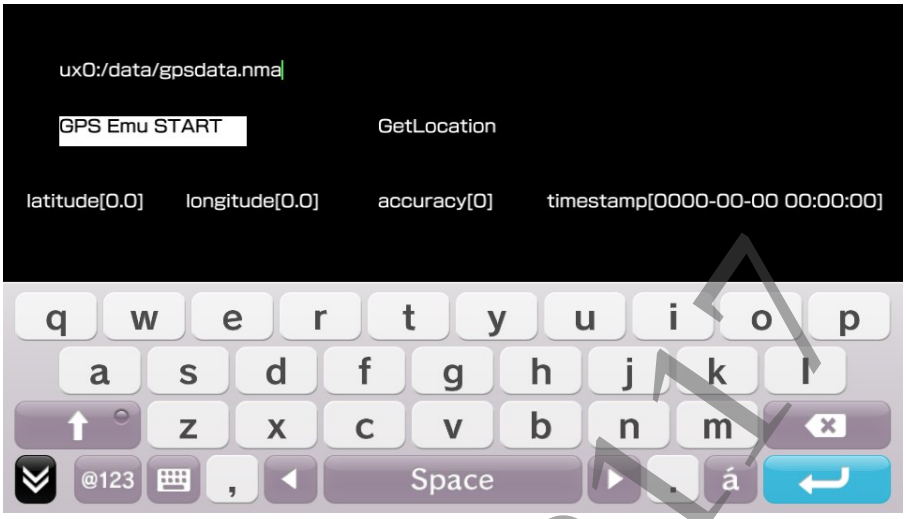
Setting the Emulation Function Using a Sample Program

Through execution of the `set_emulation_file` sample program of liblocation, it is possible to create a minimal emulation file indicating just one geographic point, set liblocation to the GPS emulation mode, and continually output the specified point to other applications.

The sample program usage method is described below.

(1) Set the file name

When the sample program starts up, IME starts up with the focus to the text box for inputting the file name. By default, ux0:/data/gpsdata/gpsdata.nma is input and basically need not be changed. Once input is completed, quit IME, for example by tapping the **Enter** button.



(2) Enter the latitude and longitude

Tap latitude: 0.0 and input the latitude.

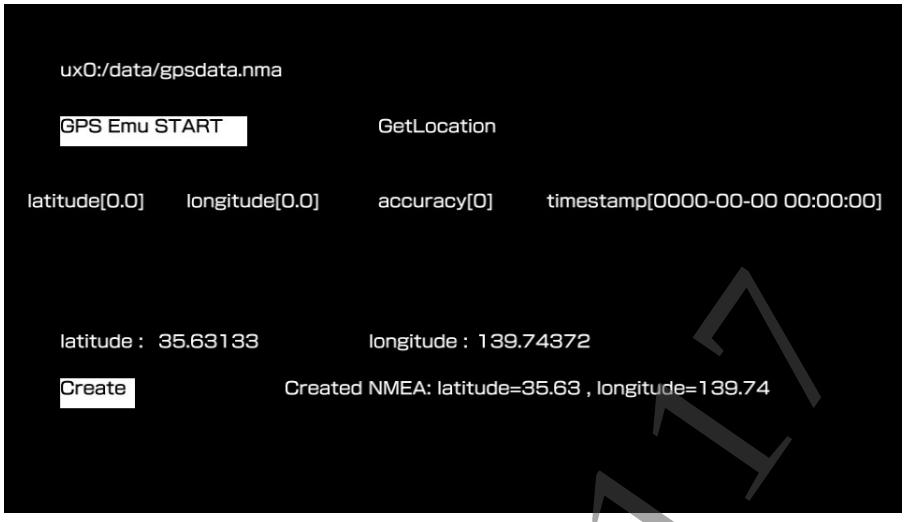
Once the specified latitude has been input from IME, quit IME, for example by tapping the **Enter** button. Input the longitude in the same manner.

(3) Create the GPS emulation file

Tap the **Create** button to create the file specified in step (1). When this is done, the created contents are displayed to the right of the **Create** button.

(4) Transition to the GPS emulation mode

Tap the **GPS Emu START** button to transition to the GPS emulation mode. The **GPS Emu START** button changes to the **GPS Emu STOP** button.



Press the PS button to suspend `set_emulation_file` sample program.

5 Notes on Location Calculation

As mentioned in the "1: Library Overview" chapter, location calculation by liblocation uses devices with respective communication modes (GPS, Wi-Fi, etc.,) and there are pros and cons for each. The GPS emulation feature provided during development outputs ideal states; in real location calculations, note the following

Coverage

Although GPS can be accessed from most areas, location calculation is difficult indoors and where the sky is obstructed. Wi-Fi and mobile communication do not depend so much on being indoors/outdoors; however, location calculation itself is not possible in areas where the calculation elements are not registered to the location database. Thus, there are areas and states in which location cannot be calculated by any device. With Wi-Fi and mobile communication, especially, supported areas are limited. Do not carry out implementation under the assumption that it will always be possible to carry out location calculation.

Location Calculation Time

The time required for location calculation drastically varies from a few seconds to several minutes according to network connection/disconnection, whether support files for GPS are downloaded when using GPS (whether a connection can be made to a network and the files can be downloaded), etc. Location calculation may succeed when retrying the process for several tens of seconds.

When executing one-shot location calculation, for example, by using `sceLocationGetLocationWithTimeout()`, set the calculation time for as long as possible and implement a feature to enable retries. In the system application's Photos application and the "near" application, the first timeout is set at 20 seconds, and the timeout of a retry is set at 10 minutes.

Accuracy

The accuracy of location calculation will vary greatly according to environment and conditions. liblocation represents this location calculation accuracy as `SceLocationLocationInfo.`*`accuracy`* (horizontal error); normally, calculation accuracy will vary within a range of several meters to several hundreds of meters. Horizontal error is provided for reference purposes and it does not guarantee that the actual calculation result falls within this error range. Thus, in some conditions, the actual error may reach several kilometers.

Moreover, because the accuracy of information registered to the location database is not guaranteed for Wi-Fi and mobile communication, a completely wrong location may be output when, for example, the user registers the wrong data. Although this location error will be gradually corrected automatically as it is compared to the surrounding locations on the database, it cannot be completely eliminated. Prevent an implementation that must guarantee the accuracy of location calculations.

6 Location Calculation Support Messages

Overview

Location calculation using a Wi-Fi model is heavily limited compared to a 3G/Wi-Fi model. Thus, the response to be taken by a user when a location cannot be calculated differs as well. However, the application is not able to distinguish between these two models and it is difficult to display the most appropriate information to the user. This SDK provides Message Dialogs to display to the user when location calculation fails. Application developers can display these dialogs to present the most appropriate information to users without having to worry about different model types or states.

Dialog Displays

There are three Message Dialog formats (see below) regarding Message Dialogs for location calculation to be input to `SceMsgDialogSystemMessageParam.`*`sysMsgType`*.

- `SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE`
  A dialog in which only the OK button exists to terminate the dialog
- `SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE_RETRY`
  A dialog in which the cancel and retry buttons exist and the retry button can be pushed to attempt a retry
- `SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_OBTAINING`
  A dialog to be specified for display while location calculation is being processed

When `sceLocationGetLocation()` or `sceLocationGetLocationWithTimeout()` is executed and the return value is `SCE_LOCATION_INFO_UNDETERMINED_LOCATION` or `SCE_LOCATION_INFO_DISABLE_DEVICE`, it is appropriate to display a dialog with the option to retry.

When executing `sceLocationStartLocationCallback()` to continually carry out location calculation and the process exceeds the timeout set by the application thereby making it impossible to carry out the process (for example), it is appropriate to display the message dialog with just the OK button.

Message appropriate to each state is as follows.

- 3G/Wi-Fi model
  Encourage the user to connect to a network or go outside
- Wi-Fi model
  Encourage the user to connect to a network or announce that location calculation cannot be carried out because the user is out of range from where Wi-Fi is available
- Wi-Fi model (when application setting for Wi-Fi is off)
  Encourage the user to set the Wi-Fi setting to on

These dialogs are not required to be displayed when location calculation fails. However, their usage is recommended as a set of consistent messages to help the user understand states of the location calculation processing.

These dialogs are only usable after executing `sceLocationInit()` and before executing `sceLocationTerm()`.

For details, refer to the "SceMsgDialogSystemMessageParam" section of the "Message Dialog Reference" document and `set_emulation_file` sample program.

7 Restrictions per Country/Region

Overview

liblocation may limit location calculation based on the restrictions that apply to the country/region in which it is being used. In this case, the `SCE_LOCATION_INFO_DENIED_BY_USER` error code will return when executing an API related to location calculation, and a dialog relating to the actual restriction will be displayed when `sceLocationConfirm()` is subsequently executed.

Restrictions per country/region can be confirmed by changing specifications with **Settings -> ★Debug Settings -> System -> Region Settings**. In a country/region where age is a restriction, the age registered upon Sony Entertainment Network signup or the age set upon running the system for the first time will be checked.

These restrictions are not all permanent; they may be changed in the future by the judgment of the applicable country/region. For the current restrictions, refer to the "Countries/Regions with Restrictions" section.

Countries/Regions with Restrictions

- Oceania/UK/Europe/Russia
  In these regions, location calculation cannot be executed if age is twelve or younger.
- Korea
  Location calculation is currently not available for this country.

8 Magnetometer Sensor Correction Message

Overview

The direction calculation data of liblocation is obtained from the built-in magnetometer sensor. The magnetometer sensor being affected by the magnetic field of the external environment, calibration is required even during magnetometer sensor operation. Normally, the developer or user need not be aware of this calibration because it is done by the automatic calibration function of the system. However, depending on the environment, calibration through the automatic calibration function may be difficult, so the user may have to assist the automatic calibration function by move the device in a particular pattern in order to ensure magnetometer sensor accuracy.

Using Message Dialog

In this system, a magnetometer sensor correction message dialog is provided to prompt the user to assist the automatic calibration function. By checking the `stability` variable of the `SceLocationHeadingInfo` structure during direction information obtainment from liblocation, the application developer can display the above-mentioned message dialog when the value is `SCE_LOCATION_HEADING_STABILITY_UNSTABLE` to prompt the user to take a specific action to the device so as to enable effective calibration of the magnetometer sensor.

Regarding the implementation of the magnetometer sensor correction message dialog, refer to the "Appendix D: Implementation Guidelines for Magnetometer Sensor Calibration Message Dialog" chapter of the "Programming Startup Guide" document, "Magnetic Field Stability" chapter of the "libmotion Overview" document, and the `heading_calibration` sample program.

9 Notes for PlayStation®TV

Behavior when liblocation is Used with PlayStation®TV

When calling the following functions with PlayStation®TV, the `SCE_LOCATION_INFO_DENIED_BY_USER` error always returns and location information cannot be obtained.

- `sceLocationGetLocation()`
- `sceLocationGetLocationWithTimeout()`
- `sceLocationGetLocationWifiHistory()`
- `sceLocationQueryLocationWifiHistory()`
- `SceLocationLocationInfoCallback()`: the above error will be stored in the *`result`* argument of the callback function

When calling the following functions with PlayStation®TV, the `SCE_LOCATION_INFO_UNDETERMINED_HEADING` error always returns and directional information cannot be obtained.

- `sceLocationGetHeading()`
- `SceLocationHeadingInfoCallback()`: the above error will be stored in the *`result`* argument of the callback function

Note
Please assume that applications for PlayStation®Vita will also be executed on PlayStation®TV. As mentioned above, location information and directional information cannot be obtained with PlayStation®TV; please avoid a routine in which an application continues to wait until location information or directional information can be obtained.