File Path Setting Guide

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Document serial number: 000004892117

Document serial number: 000004892117

Table of Contents

1 Overview

Purpose and Features

The file path setting file is a setting file for specifying the path of the working directory and the overlay settings during program execution. The data folder to be accessed can be switched to a different directory, only the latest updated files can be loaded from the development host computer, and so on, without changing programs.

The following settings can be made in the file path setting file.

- Working directory (app0:)
- Save data directory (savedata0:)
- Additional contents directory (addcont0:)
- Directory of other applications
- Overlays (up to 8)

Reference Materials

For the application file configuration, save data, add data, and patches, refer to the following document.

- Application Development Process Overview

2 Usage

Creation of Setting File

Create a setting file. The host tools treat the file with the ".psp2path " extension as the file path setting file.

A sample setting file can be found in the sample program of the file path setting file.

target\samples\sample_code\developer_tools\api_mappingfile\basic\app\configuration.psp2path

Setting File Specification Method

If the program is to be started from Neighborhood for PlayStation®Vita (Neighborhood), psp2run or the Visual Studio debugger, an arbitrary setting file can be specified. If no setting file is specified and the "configuration.psp2path" file exists in the directory that contains the executable file, that file is loaded at program launch as the file path setting file.

If the setting file contains errors, program loading fails. The error's cause, location, and other relevant information are output to the console.

The setting file can be specified with Neighborhood, psp2run, or Visual Studio.

Specifying with Neighborhood

(1) Select **Load Executable**

(2) Specify the setting file to **Mapping Path** at the lower part of the **Load Executable** window

Specifying with psp2run

- Specify the setting file to the /mappingFile:<file> option

Specifying with the Debugger of Visual Studio

(1) Select **Properties** of the target project

(2) Select **Debugging** from **Configuration Properties**

(3) Specify the setting file to **Mapping File**

Note
When the **Release Check Mode** of the Development Kit is set to **Release Mode**, program launch fails if the "configuration.psp2path" setting file exists in the directory of that executable file.

Working Directory

The working directory is the root directory accessed by the program with "app0:" during program execution. The working directory can be specified with Neighborhood, psp2run or the Visual Studio debugger, but if the working directory is specified in the file path setting file, the path that is specified in the setting file is mounted as the working directory. If the working directory is not specified, the location of the executable file is mounted as the working directory.

Directories under ux0:data/userdata cannot be specified as the working directory. Likewise, it is also not possible to specify the Windows drive root directory (such as D:) as the working directory.

The working directory can be specified with Neighborhood, psp2run, or Visual Studio.

Specifying with Neighborhood

(1) Select **Load Executable**

(2) Specify the working directory to **Working Directory** in the lower section of the **Load Executable** window

Specifying with psp2run

- Specify the working directory to the /workingDirectory:<dir> option

Specifying with the Debugger of Visual Studio

(1) Select **Properties** of the target project

(2) Select **Debugging** from **Configuration Properties**

(3) Specify the working directory to **Working Directory**

Save Data Directory

The save data directory is the data save area accessed by the program with "savedata0:".

The path specified in the setting file is mounted as the save data directory. If no path is specified in the setting file, the "host0:savedata" directory is mounted if it exists, but if it does not, "savedata0:" is not mounted.

The path specified for "savedata0:" must be a path starting with "host0:" or with "launch:". Directories under ux0:data/userdata cannot be specified as savedata0:. Likewise, it is also not possible to specify the Windows drive root directory (such as D:) as the save data directory.

Overlay

Up to eight overlays can be set to the setting file.

Overlay settings with a higher overlay order are applied later. In the case of overlays with the same order, the overlay that was set later is applied later. Because overlays are applied in order from the one written at the beginning of the setting file, when two overlays have the same order, the overlay written later in the setting file will be applied later.

Among the eight overlay settings, if there is an open slot in between overlays, that slot will be skipped, but if subsequent overlays are set after that overlay, these overlays are executed.

For details on overlays, refer to the "libfios2 Overview" document.

3 Format

The file path setting file is a text file described in UTF-8. Line breaks are CR+LF. Each line consists of the item name, the "=" separator, and the value (parameter).

```
Item + "=" + Parameter + CRLF
```

If no value is set, do not write anything for the parameter, like in the following example.

```
app0=
```

All the setting items must be aligned in the order specified in this document.

Version

The format version specified in this document is "1".

File Path

If specifying the path of the file system on the development host computer, specify a Windows style path with the absolute path.

Example: host0:c:\data\video

The file path character string must be 291 bytes or less, including the device name, such as "host0:". If the file path character string exceeds this size, program launch will fail.

The path of the redirecting destination of an overlay must always start with "app0:".

The "..\" notation indicating the parent directory relatively cannot be used to describe file paths.

Note
When creating packages, there are separate file system restrictions concerning path length and depth. For details, refer to the "Package Generator User's Guide"

Limitations

In cases when the same file in the host file system is accessed with a different path using an environmental variable of Windows, etc., an unforeseen problem such as data inconsistency may occur.

Virtual device "launch:"

For greater portability of the setting file, the virtual device "launch:" may be used inside the setting file. The path where the executable file exists will be assigned to "launch:". For example, if the executable file is "host0:c:\project\game.self", "launch:\data" is mapped to "host0:c:\project\data".

The size limitation on the file path is applied after the virtual device "launch:" is mapped.

In the case of a package with a file configuration such as the following, using this virtual device makes it possible to use the package without modifying the setting file, regardless of where the package is placed on the development host computer.

Package configuration example:

```
package
├──configuration.psp2path
├──data
│    └──monkey.txt
├──game.self (executable file)
├──patch1
│    └──monkey2.txt
└──patch2
     └──monkey.txt
```

Overlay setting example of setting file (configuration.psp2path):

```
version=1
app0=
 :
overlay1type=
overlay1order=
overlay1src=launch:\patch1
overlay1dst=app0:data
overlay2type=
overlay2order=
overlay2src=launch:\patch2
overlay2dst=app0:data
 :
```

Setting Items

Table 1 Setting Items

Line	Item	Description
1	version	The setting file format version. The value is "1".
2	app0	Working directory Paths beginning with "app0:" cannot be specified.
3	savedata0	Save data directory
4	addcont0	Additional contents directory (not supported)
5	otherApps	Root directory of other applications (not supported)
6	otherAppsPatches	Root directory of patches of other applications (not supported)
7	otherAppsSavedata	Root directory of save data of other applications (not supported)
8	otherAppsContents	Root directory of additional contents of other applications (not supported)

9	overlay1type	Type of overlay 1. Either one of the following character strings:
		`SCE_FIOS_OVERLAY_TYPE_OPAQUE`
		`SCE_FIOS_OVERLAY_TYPE_TRANSLUCENT`
		`SCE_FIOS_OVERLAY_TYPE_NEWER`
		`SCE_FIOS_OVERLAY_TYPE_WRITABLE`
		The default is `SCE_FIOS_OVERLAY_TYPE_TRANSLUCENT`

10	overlay1order	Search order of overlay 1. Any decimal number from 0 to 127. The default is "0".
11	overlay1src	Source path of overlay 1. If this item is not set, do not set overlay1dst either.
12	overlay1dst	Destination path (redirecting destination path) of overlay 1. If this item is not set, do not set overlay1src either.
Subsequently, same overlay settings as lines 9 - 12(overlay*type, overlay*order, overlay*src, overlay*dst) will continue until overlay8.		

Example

```
version=1
app0=host0:c:\game
savedata0=app0:savedata
addcont0=host0:c:\data\contents
otherApps=
otherAppsPatches=
otherAppsSavedata=
otherAppsContents=
overlay1type=SCE_FIOS_OVERLAY_TYPE_TRANSLUCENT
overlay1order=63
overlay1src=host0:c:\data\music
overlay1dst=app0:music
overlay2type=
overlay2order=
overlay2src=launch:\data\stage1
overlay2dst=app0:data/stage
   :
overlay8type=
overlay8order=
overlay8src=
overlay8dst=
```