SmileBASIC file formats

File formats used in the application series "SmileBASIC"

Table Of Contents

Note: This document is not complete. The following content will be added soon:

- Petit Computer / PTC mkII formats
  - File types
  - QR Codes format

SmileBASIC 3 & 4

File Location and Naming Convention

TODO:	• Find raw location for SmileBASIC 4 • SmileBASIC 4 has PNG file support, according to the #GUIDE project, however, they're marked as GRP in FILES(). Are they borrowing the G prefix as well? • What's the difference between the B/G prefixes, if both can be loaded/saved interchangably; or is it just for user preference?

The file location depends on the respective platform for which a SmileBASIC version is used on:

3DS (SmileBASIC 3):
Raw location: `SD:/Nintendo 3DS/[ID0]/[ID1]/extdata/00000000/[TID]` *1
- The [DEFAULT] folder is named ###

Wii U (Petit Computer BIG):
Raw location: `mlc:/usr/save/00050000/101dff00/user/[UID]/` *2
- The [DEFAULT] folder is named ###

Switch (SmileBASIC 4):
Raw location: ??? (not known, will be investigated)

The files have a prefix to simplify lookup:

T	Text files (i.e. strings or whole programs)
B	Binaries (DAT files)
P	Project files *3
G	GRP files — SmileBASIC 4-only
J	JPG files — SmileBASIC 4-only

However, the only exception to the prefix is the META file (see "Project Files → Project Metadata")

Note: To simplify notes and mentions to specific SmileBASIC versions, I will treat "SmileBASIC 3 (3DS)" and "Petit Computer BIG (Wii U)" as "SmileBASIC 3", unless specifically noted.

Important Notice: Any value in the format descriptions are little-endian.

*1 00001172 for Japan, 000016DE for USA, 00001A1C for EUR.

*2 The user ID depends on the account used.

*3 Project files are never normally seen outside of SmileBASIC server communication. SmileBASIC packs and unpacks the content on-the-fly when uploading/downloading.

Common Header

The common header is present on all files in SmileBASIC 3, and 4. It contains the most important information for the parser to know, what type of content it awaits, as well as time and creator information.

Offset	Type (Size)	Name	Note
0x00	Int16 (2)	Source version	The SmileBASIC version that this file was created in (0-3 = SB3*1 ; 4 = SB4)
0x02	Int16 (2)	File Type	For SmileBASIC 3: 0=TXT, 1=DAT, 2=PRJ For SmileBASIC 4: 0=TXT, 1=DAT, 2=GRP, 3=PRJ, 4=META
0x04	Int16 (2)	File Options Bitmask	bit0 – zLib Compression bit1 – Read-out protection *2
0x06	Int16 (2)	File Icon	The icon to show in SB3 file browser. *3 0=TXT/DAT, 1=PRG, 2=GRP
0x08	Int32 (4)	Data Size	File size, excluding the common header and the footer
0x0C	Int16 (2)	Modification Date (Year)	Full year (i.e. 2022)
0x0E	Int8 (1)	Modification Date (Month)	Month (1-12)
0x0F	Int8 (1)	Modification Date (Day)	Day (1-31)
0x10	Int8 (1)	Modification Date (Hour)	Hour (0-23)
0x11	Int8 (1)	Modification Date (Minute)	Minute (0-59)
0x12	Int8 (1)	Modification Date (Second)	Second (0-59)
0x13	Int8 (1)	Modification Date (Week day)	0=Sunday, 1=Monday, ... , 6=Saturday *4

*1 The version number depends on the subversion, however, it isn't clear and has to be investigated. Usually, the version is set to 1 on SmileBASIC 3.

*2 The read-out protection flag prevents any form of reading data from a DAT container. DAT files simply fail to load and GRP's cannot be read from, even after manipulating the GRP (or even clearing out with `GCLS`). That internal flag can only be removed using `ACLS`. The flag in the file appears to be SmileBASIC 3-exclusive.

*3 This icon controls how the file acts when loaded through the file browser. TXT/PRG are attempted to be loaded in the active program slot, whereas DAT/GRP just close the browser.

*4 This value is unused, but still written by SmileBASIC 3. In SmileBASIC 4, it's also unused but additionally overwritten with 0xFF (-1).

SmileBASIC 3 and 4 also have creator information, however, the header continues differently for both versions.

SmileBASIC 3:

Offset	Type (Size)	Name	Note
0x14	String (18)	Creator's NNID (ASCII)	
0x26	String (18)	Editor's NNID (ASCII)	This NNID is shown in the file browser.
0x38	Uint32 (4)	Creator's user ID*4	This ID is used for the blacklist.
0x3C	Uint32 (4)	Editor's user ID*4	
0x40	Int64 (8)	Creator's Upload ID*5	
0x48	Int64 (8)	Editor's Upload ID*5	

SmileBASIC 4:

Offset	Type (Size)	Name	Note
0x14	String (32)	Creator's account name (UTF-8)	
0x34	String (32)	Editor's account name (UTF-8)	This name is shown in the file browser.
0x54	Uint32 (4)	Creator's user ID*4	This ID is used for the blacklist.
0x58	Uint32 (4)	Editor's user ID*4	
0x5C	Int64 (8)	Creator's Upload ID*5	
0x64	Int64 (8)	Editor's Upload ID*5	
0x6C	Uint32 (4)	Padding	To keep the preceding data 0x10-byte-aligned

*4 The user ID is tied to the NNID / Nintendo Account.

*5 The Upload ID appears to be a counter as to which upload the file/project was created/reuploaded in.

Note: These values do not update when creating/overwriting files, these only get updated when sent to and received from the SmileBASIC server.

The common header is 0x50 bytes long for SB3, and 0x70 bytes for SB4.

The Footer

The footer is used for verification, whereas it is a HMAC-SHA1 hash of the headers and the data section.

The HMAC key used is:

```
nqmby+e9S?{%U*-V]51n%^xZMk8>b{?x]&?(NmmV[,g85:%6Sqd"'U")/8u77UL2
```

Should a file not pass the hash check, will it not be able to be loaded or uploaded. This will also cause a download to fail, if you somehow managed to upload a corrupted file.

Secondary header for DAT/GRP

While TXT/PRG store the text (UTF-8-encoded) right away after the common header, DAT/GRP need more information, so they can be properly parsed. That is why they have the secondary header.

Note: The offsets shown are relative to after the header, which depends on the SmileBASIC version, found in the common header.

Offset	Type (Size)	Name	Note
0x00	String (8)	Magic "PCBN000X" (Petit Computer BiNary)	X is 1 in SB3, and 4 in SB4
0x08	Int16 (2)	Content Type	Unused: 0=Int8, 1=Uint8, 2=Int16 Used: 3=Uint16, 4=Int32, 5=double, 6=string *1 *2
0x0A	Int16 (2)	Dimension Count	Redundant; simplies array creation (Range: 1-4)
0x0C	Int32 (4)	Size of first dimension	
0x10	Int32 (4)	Size of second dimension	If dimension count is 1, a garbage value may be read
0x14	Int32 (4)	Size of third dimension	If dimension count is 1/2, a garbage value may be read
0x18	Int32 (4)	Size of fourth dimension	If dimension count is 1-3, a garbage value may be read

The data is stored in Row-major order. The secondary DAT header is 0x1C bytes long, which is counted in the Common Header's file size value.

*1 For DAT files, only type 4 and 5 are used.
Type 3 is only used by SB3 GRP files, as SB3 uses a 16bit pixel format internally (RGBA5551)
An unused type cannot be saved from SB3/SB4 itself, the file has to be made externally.
If overwriting such a file, types 0 through 3 will be set to type 4 and the content reformatted to Int32 (so content will be zero-padded).

For GRP:
In SmileBASIC 3, if the content type isn't type 3 (Uint16), it will not plot the content.
In SmileBASIC 4, the content type has to be 3 (for RGBA5551) or 4 (for ARGB8), otherwise the content will not be plotted.

*2 The *string* Content Type is only avilable on SmileBASIC 4. Elements have no fixed width but preceded with a u32 denoting the element's length. Although it appears there's a restriction on how long an element can be. This needs some more investigation.

Project Files

These files are only encountered when communicating with the SmileBASIC servers, i.e. when uploading/downloading projects, however, the file is never written to the save data, as it's packed and unpacked during transmission. You can obtain such a file however when manually communicating with the server.

The file format was upgraded in SmileBASIC 4, whereas it introduced project metadata.

The format's secondary header for SmileBASIC 3:

Offset	Type (Size)	Name	Note
0x00	Int32 (4)	Project size	This value holds the total size of all files embedded below this header.
0x04	Int32 (4)	File Count	The amount of files embedded in this project file.
Repeating the following struct for the file count specified (offset is relative to the above)			
0x00	Int32 (4)	File n's size	
0x04	String (16)	File n's name (ASCII)	SB3 file names are limited to 14 characters, the other 2 are the file prefix and a NULL-terminator for the string

The format's secondary header for SmileBASIC 4:

Offset	Type (Size)	Name	Note
0x0000	Metadata (0x2B20 *1)	Project Metadata	See ("Project Files → Project Metadata" for its format)
0x2B20	Int32 (4)	Project size	This value holds the total size of all files embedded below this header.
0x2B24	Int32 (4)	File Count	The amount of files embedded in this project file.
Repeating the following struct for the file count specified (offset is relative to the above)			
0x00	Int32 (4)	File n's size	
0x04	String (36)	File n's name (ASCII)	SB4 file names are limited to 32 characters, although here, the prefix and a NULL-terminator are also used. This buffer length is likely chosen for byte alignment

Following the header are the full contents of the files in a project, including their respective Common Headers and Footers.

The project file itself has another HMAC footer, just in case.

*1 The Metadata format appears to have intentions of being flexible in size, however, it's always expected to be 0x2B20 bytes long.

Project Metadata

This format is exclusive to SmileBASIC 4.

This file format is used inside a project file's header and as a standalone file inside a project folder.

The metadata in the project file header is used by the Play Menu when looking around in the "Published Works" section.

The metadata as a standalone is used by the Play Menu and the file browser for the projects you have stored locally.

Fun fact: META files are the only files that don't have the HMAC footer, likely for its embedded version inside project files, which those have actually have the footer.

Offset	Type (Size)	Name	Note
0x0000	String (8)	Magic "PCPM0005" (Petit Computer Project Metadata)	
0x0008	UCS-2 (48)	Project name (24 char)	These strings are encoded in UCS-2-LE, whereas any character takes 2 bytes each, therefore halving the actual text length
0x0038	UCS-2 (4576)	Description (2288 char)	
0x1218	Int32 (4)	Icon size (n)	Under normal circumstances, it's always set to 40 (0x28) This value defines width and height.
0x121C	Uint32 array (n * n * 4)	Icon data (ARGB8-encoded)	Also stored in row-major order. For this format description, let's keep it at default (40) The size would be 6400 bytes (0x1900)
0x2B1C	Int32 (4)	Padding (?)	This might be a dynamic padding that could change if the icon size was changed in a valid way.

Sources / Credits

MasterR3C0RD — Format page on Old SBS, SBAPI File Parser

SmileBoom Co. Ltd. — Creator/Developer of the SmileBASIC series