

# Bundle File

**\*Originally written by Simon W.**

## Bundle Database (BLB)

The Bundle Database file, also known as “bundle\_db.blb” and “all.blb”, defines all file entries with their hashed path, hashed extension, language, and a unique ID. Special things to note: “idstring\_lookup.idstring\_lookup” has an ID that is equal to the count of file entries.

## File structure of the .blb format

### Header

```
uint32 languages_tag //Payday 2 = "8C F2 18 00" Payday: The Heist = "9C 2B F2 00"
uint32 languages_count
uint32 languages_count //should be same value as previous uint32
uint32 languages_offset //offset to the beginning of languages section
uint32 unknown
uint32 unknown
uint32 unknown
uint32 file_entries_count
uint32 file_entries_count //should be same value as previous uint32
uint32 file_entries_offset //offset to the beginning of file entries section
uint32 unknown
uint32 unknown
uint32 unknown
uint32 unknown
```

### Languages section

```
FOREACH( languages_count )
uint64 language_hash
uint32 language_representation
uint32 unknown
END FOREACH
```

## File entries section

```
FOREACH( file_entries_count )
uint64 file_entry_extension_hash
uint64 file_entry_path_hash
uint32 file_entry_language //one of the language representations
uint32 unknown
uint32 file_entry_ID
uint32 unknown
END FOREACH
```

# Packages (bundles)

Bundles are diesel's packages containing files with corresponding IDs. The package name can be looked up with Bundle Modder by using the hash converter on the bundle name with “use hex” and “swap endianness” (this does not work with all\_x bundles).

Which then can be used in lua to load packages. Bundles are split into two files, header and data. The header files end with “\*\_h.bundle” while data files end with “\*.bundle”.

## File structure of the \_h.bundle (Header)

### Header section

```
uint32 section_size
uint32 section_tag //Since update 70, this tag is present in every file. Payday: The Heist is present only if bundle
```

```

is all_x
uint32 file_entries_count
uint32 file_entries_count //should be same value as previous uint32
uint32 file_entries_offset //if section_tag == "00 00 00 00" then file_entries_offset += 4

uint32 file_entries_tag //Payday 2 = "E8 EB 18 00" Payday: The Heist = "88 EE 18 00"
FOREACH( file_entries_count )
    uint32 file_entry_ID
    uint32 file_entry_address //address within the *.bundle file
    IF bundle is all_x then
        uint32 file_entry_length //the length of this file entry
    END IF
END FOREACH

```

## Footer section

Please note that footer items correspond to the header file\_entries in same order.

```

uint32 section_tag //Payday 2 and Payday: The Heist = "F8 C5 FC EB"
uint32 section_size
uint32 section_item_count
uint32 unknown
uint32 unknown
uint32 unknown //tag?
FOREACH( section_item_count )
    uint64 item_extension_hash
    uint64 item_path_hash
    uint32 unknown //end tag?
END FOREACH
uint32 zero //end

```

## File structure of the h.bundle (Data)

This file consists of a collection of files. Each file entry starts at a specified address in header with a specified length, if length is not specified then it is calculated as a difference between current entry

address and next.

```
FOREACH( file_entries_count )  
    byte[] file_entry  
END FOREACH
```

---

Revision #1

Created 25 October 2019 16:42:58 by Luffy

Updated 2 July 2021 16:43:16 by Luffy