ADVENTURE SYSTEM

# ADVENTURE_IO

## I/O format and libraries for ADVENTURE mudules

## Document 入出力関数リスト

February 17, 2006

**ADVENTURE Project**

# 目 次

# 1 AdvFile の Open/Close

- `AdvDocFile* adv_dio_file_open (const char* filename, const char* mode)`
  Adv ファイルを開く.

| | | |
|---|---|---|
| **戻り値** | | AdvDocFile のポインタ |
| **引数** | `filename`: | AdvDocFile に格納するファイルの名前 |
| | `mode`: | オープンする目的を表すモード<br>'r' for read(default), 'c' for create,<br>'a' for append |

- `void adv_dio_file_close(AdvDocFile* dfile)`
  Adv ファイル `dfile` をクローズする.

| | | |
|---|---|---|
| **戻り値** | | なし |
| **引数** | `dfile`: | クローズする AdvDocFile のポインタ |

- `const char* adv_dio_file_get_locator(AdvDocFile* dfile)`
  dfile が指している Advfile の絶対パスを返す.

| | | |
|---|---|---|
| **戻り値** | | 絶対パス |
| **引数** | `dfile`: | AdvDocFile のポインタ |

## 2 AdvDocument の Open/Close

- `AdvDocument* adv_dio_create (AdvDocFile* dfile, const char* did)`
  dfile の中に新規に Document をオープンする.DocumentID `did` は、`adv_dio_make_documentid`(4 ページ) を用いて作成する.

| | | |
|---|---|---|
| **戻り値** | | 開く AdvDocument のポインタ |
| **引数** | `dfile`: | AdvDocFile ポインタ |
| | `did`: | AdvDocument に付ける DocumentID |

- `AdvDocument* adv_dio_open_by_documentid(AdvDocFile* dfile, const char* did)`
  dfile の中から指定した DocumentID を持つ Document をオープンする.

| | | |
|---|---|---|
| **戻り値** | | 該当する AdvDocument のポインタ |
| **引数** | `dfile`: | AdvDocFile のポインタ (検索元) |
| | `did`: | DocumentID |

- `AdvDocument* adv_dio_open_nth (AdvDocFile* dfile, int n)`
  `dfile` 中の `n` 番目の Document をオープンする.

| | | |
|---|---|---|
| **戻り値** | | 該当する AdvDocument のポインタ |
| **引数** | `dfile`: | AdvDocFile のポインタ (検索元) |
| | `n`: | 整数 |

- `AdvDocument* adv_dio_open_by_property (AdvDocFile* dfile, void * prev, ..., NULL)`
  dfile 中から指定した property を含む Document をオープンする."..." には検索する property の key と 値 を順に文字列として並べる. `prev` が NULL の場合は一番最初にマッチしたドキュメントを返す. `prev` を以前マッチした Document のポインタとした場合は、 `prev` の指す Document の次にマッチする Document が得られる. 検索指定の最後には必ず `NULL` を付け加える。

| | | |
|---|---|---|
| **戻り値** | | 該当する AdvDocument のポインタ |
| **引数** | `dfile`: | 検索先の AdvDocFile |
| | `prev`: | "..."の条件にマッチする Document |
| | `"..."`: | 検索条件 |

**使用例：**

```
doc = adv_dio_open_by_property
 ( dfile, NULL, "content_type", "FEGenericAttribute",
                "label", "Load", NULL);
/* dfile から "content_type=FEGenericAddribute" と "label=Load"
   の property を持つ Document を一つ開く。*/
```

- `AdvDocument* adv_dio_open_by_locator (const char* locator)`
  locator (Document が収録されているファイルへのパスと DocumentID の組合せ：4 ページ参照) に対応する Document を開く.

| | | |
|---|---|---|
| **戻り値** | | 該当する Document のポインタ |
| **引数** | `locator`: | ファイルへのパスと DocumentID を文字 ”?” でつなげた文字列 |

- `void adv_dio_close (AdvDocument* doc)`
  Document をクローズする.

| | | |
|---|---|---|
| **戻り値** | | なし |
| **引数** | `doc`: | 閉じる Document のポインタ |

## 3 AdvDocument の Status

- `const char* adv_dio_make_documentid(const char* str)`
  `str` を基に DocumentID を作る。

| | | |
|---|---|---|
| **戻り値** | | 作った DocumentID |
| **引数** | `str`: | DocumentID の基になる文字列<br>(例えば label@content_type) |

- `const char* adv_dio_get_documentid (AdvDocument* doc)`
  `doc` の指している Document の DocumentID を返す。

| | | |
|---|---|---|
| **戻り値** | | DocumentID (文字列) |
| **引数** | `doc`: | Document のポインタ |

- `adv_off_t adv_dio_get_size (AdvDocument* doc)`
  `doc` の指している Document のマスデータの size を返す。

| | | |
|---|---|---|
| **戻り値** | | Document の size (整数) |
| **引数** | `doc`: | Document のポインタ |

- `const char* adv_dio_get_locator (AdvDocument* doc)`
  Document `doc` の locator を取得する。locator は Document をユニークに指し示すための文字列であり、file の絶対パスと `doc` の DocumentID より file の絶対パスと DocumentID を文字 "?" でつなげた文字列で表わされる。

| | | |
|---|---|---|
| **戻り値** | | locator 文字列 |
| **引数** | `doc`: | AdvDocument |

**使用例**：

```
/*********************************
 example.c
*********************************/
AdvDatabox *dbox;
AdvDocument *docin;

dbox = adv_dbox_new();
adv_dbox_add(dbox, argv[1]);
```

```
docin = adv_dbox_find_by_property(
        dbox,NULL, "label","Displacement",NULL);
/* "label=Displacement" の Property を持つ Document を docin にいれる */
fprintf(stdout,"%s\n",adv_dio_get_locator(docin));
```

**出力結果**：

```
% example Disp.adv
/home/Disp.adv?6B8B4567:Displacement@HDDM_
FEGA:1988:39E5AB59
  ( → Disp.adv のパスと docin の DocumentID が表示された)
%
```

## 4 property の読み書き

- `void adv_dio_set_property(AdvDocument* doc, const char* key, const char* val)`

  Document `doc` に char 型の値を持つ property をセットする。

| | | |
|---|---|---|
| **戻り値** | | なし |
| **引数** | `doc`: | セットする AdvDocument |
| | `key`: | property の項目 |
| | `val`: | 値としてセットする `char` 型データ |

- `void adv_dio_set_property_int32(AdvDocument* doc, const char* key, int32 val)`

  Document `doc` に int32 型の値を持つ property をセットする。

| | | |
|---|---|---|
| **戻り値** | | なし |
| **引数** | `doc`: | セットする AdvDocument |
| | `key`: | property の項目 |
| | `val`: | 値としてセットする `int32` 型データ |

- `void adv_dio_set_property_float64(AdvDocument* doc, const char* key, float64 val)`

  Document `doc` の property に float64 型の値を持つ property をセットする

| | | |
|---|---|---|
| **戻り値** | | なし |
| **引数** | `doc`: | セットする AdvDocument |
| | `key`: | property の項目 |
| | `val`: | 値としてセットする `float64` 型データ |

- `const char* adv_dio_get_property(AdvDocument* doc, const char* key)`

  Document `doc` から、char 型の値を持つ property を読む。

| | | |
|---|---|---|
| **戻り値** | | `key` に対応する property の値 |
| **引数** | `doc`: | 読む AdvDocument |
| | `key`: | property の項目 |

- `bool adv_dio_get_property_int32(AdvDocument* doc, const char* key, int32* val)`

Document doc から、int32 型の値を持つ property を読む。

| | | |
|---|---|---|
| **戻り値** | | 正常時：0以外の値、エラー時：値0 |
| **引数** | doc: | 読む AdvDocument |
| | key: | property の項目 |
| | val: | データ代入先ポインタ (int32 型) |

- bool adv_dio_get_property_float64(AdvDocument* doc, const char* key, float64* val)

  Document doc から、foeat64 型の値を持つ property を読む。

| | | |
|---|---|---|
| **戻り値** | | 正常時：0以外の値、エラー時：値0 |
| **引数** | doc: | 読む AdvDocument |
| | key: | property の項目 |
| | val : | データ代入先ポインタ (float64 型) |

- bool adv_dio_get_nth_property(AdvDocument* doc, int n, char* key, int keysize, char* val, int valsize)

  Document doc の n 番目の property の項目と値をそれぞれ key,val にいれる。keysize,valsize はそれぞれ key, val の読み込み最大文字数であり、key, val はあらかじめそれより大きな領域を確保しておく必要がある。

| | | |
|---|---|---|
| **戻り値** | | 正常時：0以外の値、エラー時：値0 |
| **引数** | doc: | 読み込み先の AdvDocument |
| | n: | 整数 |
| | key: | property の項目名を表すポインタ |
| | keysize: | key に格納できる最大の文字数 |
| | val: | 変数のポインタ |
| | valsize: | val に格納できる最大の文字数 |

**使用例**

```
AdvDocument *doc
int n = 0;
char key[1024];
char val[1024];
while (adv_dio_get_nth_property
            (doc, n, key, sizeof(key), val, sizeof(val))) n++;
        /* 全ての property を読み込む */
```

- `void adv_dio_unset_nth_property (AdvDocument* doc, int n)`
  `doc` に格納された Document の `n` 番目の property をメモリから消去する。

| **戻り値** | | なし |
|---|---|---|
| **引数** | `doc`: | property を消去する AdvDocument ポインタ |
| | `n`: | 整数 |

# 5 マスデータの読み込み

- `int32 adv_dio_read_octet(AdvDocument* doc, adv_off_t offset, int32 len, octet* buf)`

  AdvDocument `doc` のマスデータ[1] を `offset` の位置から octet 型 (8 ビット型) データとして、`len` 個を読む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | `doc`: | 読み取る Document |
| | `offset`: | マスデータ内の読み込み位置 |
| | `len`: | 読み取る octet data の個数 |
| | `buf`: | 読み取った octet data の格納先アドレス |

- `int32 adv_dio_read_string_length(AdvDocument* doc,adv_off_t offset)`

  AdvDocument `doc` のマスデータの `offset` の位置にある文字列データの文字数を数える。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | `doc`: | 読み取る Document |
| | `offset`: | マスデータ内の読み込み位置 |

- `int32 adv_dio_read_string(AdvDocument* doc, adv_off_t offset, char* buf)`

  AdvDocument `doc` のマスデータの `offset` の位置にある文字列データを読み込んで、`buf` に格納する。`buf` のサイズは文字列データの文字数以上確保しておく必要がある。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | `doc`: | 読み取る Document |
| | `offset`: | マスデータ内の読み込み位置 |
| | `buf`: | 読み込んだ文字列データの格納先アドレス |

- `int32 adv_dio_read_int8(AdvDocument* doc, adv_off_t offset, int8* val)`

  AdvDocument `doc` のマスデータを `offset` の位置から 8 ビット int 型として読み込む。

[1] Raw Data とも呼ばれる

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | doc: | 読み取る Document |
| | offset: | マスデータ内の読み込み位置 |
| | val: | 読み込んだデータの格納先アドレス |

- int32 adv_dio_read_int8v(AdvDocument* doc, adv_off_t offset, int num, int8* val)

  AdvDocument doc のマスデータを offset の位置から 8 ビット int 型として num 個読み込む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | doc: | 読み取る Document |
| | offset: | マスデータ内の読み込み位置 |
| | num: | 読み取るデータの個数 |
| | val: | 読み込んだデータの格納先アドレス |

- int32 adv_dio_read_int16(AdvDocument* doc, adv_off_t offset, int16* val)

  AdvDocument doc のマスデータを offset の位置から 16 ビット int 型として読み込む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | doc: | 読み取る Document |
| | offset: | マスデータ内の読み込み位置 |
| | val: | 読み込んだデータの格納先アドレス |

- int32 adv_dio_read_int16v(AdvDocument* doc, adv_off_t offset, int num, int16* val)

  AdvDocument doc のマスデータを offset から 16 ビット int 型として num 個読み込む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | doc: | 読み取る Document |
| | offset: | マスデータ内の読み込み位置 |
| | num: | 読み取るデータの個数 |
| | val: | 読み込んだデータの格納先アドレス |

- int32 adv_dio_read_int32(AdvDocument* doc, adv_off_t offset, int32* val)

  AdvDocument doc のマスデータを offset から 32 ビット int 型として読み込む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | doc: | 読み取る Document |
| | offset: | マスデータ内の読み込み位置 |
| | val: | 読み込んだデータの格納先アドレス |

- `int32 adv_dio_read_int32v(AdvDocument* doc, adv_off_t offset, int num, int32* val)`

  AdvDocument `doc` のマスデータを `offset` から 32 ビット int 型として `num` 個読み込む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだ型データのサイズ |
| **引数** | doc: | 読み取る Document |
| | offset: | マスデータ内の読み込み位置 |
| | num: | 読み取るデータの個数 |
| | val: | 読み込んだデータの格納先アドレス |

- `int32 adv_dio_read_int64(AdvDocument* doc, adv_off_t offset, int64* val)`

  AdvDocument `doc` のマスデータを `offset` から 64 ビット int 型として読み込む。64 ビット int 型が用意されていない環境では `val` のデータ型には `int32*` を用いる。この場合、32 ビット分のみが返される。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | doc: | 読み取る Document |
| | offset: | マスデータ内の読み込み位置 |
| | val: | 読み込んだデータの格納先アドレス |

- `int32 adv_dio_read_int64v(AdvDocument* doc, adv_off_t offset, int num, int64* val)`

  AdvDocument `doc` のマスデータを `offset` から 64 ビット int 型として `num` 個読み込む。64 ビット int 型が用意されていない環境では `val` のデータ型には `int32*` を用いる。この場合、32 ビット分のみ返される。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | doc: | 読み取る Document |
| | offset: | マスデータ内の読み込み位置 |
| | num: | 読み取るデータの 個数 |
| | val: | 読み込んだデータの格納先アドレス |

- `int32 adv_dio_read_float32(AdvDocument* doc, adv_off_t offset, float32* val)`

  AdvDocument `doc` のマスデータを `offset` から 32 ビット float 型として読み込む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | `doc`: | 読み取る Document |
| | `offset`: | マスデータ内の読み込み位置 |
| | `val`: | 読み込んだデータの格納先アドレス |

- `int32 adv_dio_read_float32v(AdvDocument* doc, adv_off_t offset, int num, float32* val)`

  AdvDocument `doc` のマスデータを `offset` から 32 ビット float 型として `num` 個読み込む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | `doc`: | 読み取る Document |
| | `offset`: | マスデータ内の読み込み位置 |
| | `num`: | 読み取るデータの個数 |
| | `val`: | 読み込んだデータの格納先アドレス |

- `int32 adv_dio_read_float64(AdvDocument* doc, adv_off_t offset, float64* val)`

  AdvDocument `doc` のマスデータを `offset` から 64 ビット float 型として読み込む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | `doc`: | 読み取る Document |
| | `offset`: | マスデータ内の読み込み位置 |
| | `val`: | 読み込んだデータの格納先アドレス |

- `int32 adv_dio_read_float64v(AdvDocument* doc, adv_off_t offset, int num, float64* val)`

  AdvDocument `doc` のマスデータを `offset` から 64 ビット float 型として `num` 個読み込む。

| | | |
|---|---|---|
| **戻り値** | | 読み込んだデータのサイズ |
| **引数** | doc: | 読み取る Document |
| | offset: | マスデータ内の読み込み位置 |
| | num: | 読み取るデータの個数 |
| | val: | 読み込んだデータの格納先アドレス |

# 6 マスデータの書き込み

- `int32 adv_dio_write_octet (AdvDocument* doc, adv_off_t offset, int32 length, const octet* buf)`

  Document `doc` の `offset` の位置に octet 配列 `buf` を書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータサイズ |
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `len`: | 書き込む Octet Data の長さ |
| | `buf`: | 書き込む Octet Data |

- `int32 adv_dio_write_string (AdvDocument* doc, adv_off_t offset, const char* buf)`

  Document `doc` の `offset` の位置に文字列 `buf` を書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータサイズ |
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `buf`: | 書き込む文字列データ |

- `int32 adv_dio_write_int8 (AdvDocument* doc, adv_off_t offset, int8 val)`

  Document `doc` の `offset` の位置に 8 ビット int 型のデータを書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込むデータのサイズ |
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `val`: | 書きこむデータ |

- `int32 adv_dio_write_int8v (AdvDocument* doc, adv_off_t offset, int num, const int8* val)`

  Document `doc` の `offset` の位置に `num` 個の 8 ビット int 型データを書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータのサイズ |
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `num`: | データの個数 |
| | `buf`: | 書き込むデータ |

- `int32 adv_dio_write_int16 (AdvDocument* doc, adv_off_t offset, int16 val)`
  Document `doc` の `offset` の位置に 16 ビット int 型データを書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータのサイズ |
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `val`: | 書き込むデータ |

- `int32 adv_dio_write_int16v (AdvDocument* doc, adv_off_t offset, int num, const int16* val)`
  Document `doc` の `offset` の位置に `num` 個の 16 ビット int 型データを書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータのサイズ |
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `num`: | データの個数 |
| | `val`: | 書き込むデータ |

- `int32 adv_dio_write_int32 (AdvDocument* doc, adv_off_t offset, int32 val)`
  Document `doc` の `offset` の位置に 32 ビット int 型データを書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータサイズ |
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `val`: | 書き込むデータ |

- `int32 adv_dio_write_int32v (AdvDocument* doc, adv_off_t offset, int num, const int32* val)`
  Document `doc` の `offset` の位置に `num` 個の 32 ビット int 型データを書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータのサイズ |
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `num`: | データの個数 |
| | `val`: | 書き込むデータ |

- `int32 adv_dio_write_int64 (AdvDocument* doc, adv_off_t offset, int64 val)`
  Document `doc` の `offset` の位置に 64 ビット int 型データを書き込む。64 ビッ

ト int 型が用意されていない環境では val のデータ型には int32 を用いる。この場合、不足の 32 ビット分に 0 が埋められて出力される。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータのサイズ |
| **引数** | doc: | 書き込む Document |
| | offset: | マスデータ内の書き込み位置 |
| | val: | 書き込むデータ |

- `int32 adv_dio_write_int64v (AdvDocument* doc, adv_off_t offset, int num, const int64* val)`

  Document `doc` の `offset` の位置に `num` 個の 64 ビット int 型データを書き込む。64 ビット int 型が用意されていない環境では val のデータ型には int32* を用いる。この場合、不足の 32 ビット分に 0 が埋められて出力される。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータのサイズ |
| **引数** | doc: | 書き込む Document |
| | offset: | マスデータ内の書き込み位置 |
| | num: | データの個数 |
| | val: | 書き込むデータ |

- `int32 adv_dio_write_float32 (AdvDocument* doc, adv_off_t offset, float32 val)`

  Document `doc` の `offset` の位置に 32 ビット float 型データを書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータのサイズ |
| **引数** | doc: | 書き込む Documnet |
| | offset: | マスデータ内の書き込み位置 |
| | val: | 書き込むデータ |

- `int32 adv_dio_write_float32v (AdvDocument* doc, adv_off_t offset, int num, const float32* val)`

  Document `doc` の `offset` の位置に `num` 個の 32 ビット float 型データを書き込む。

| | | |
|---|---|---|
| **戻り値** | | 書き込んだデータのサイズ |
| **引数** | doc: | 書き込む Document |
| | offset: | マスデータ内の書き込み位置 |
| | num: | データの個数 |
| | val: | 書き込むデータ |

- `int32 adv_dio_write_float64 (AdvDocument* doc, adv_off_t offset, float64 val)`

  Document `doc` の `offset` の位置に 64 ビット float 型のデータを書き込む。

| **戻り値** | | 書き込んだデータのサイズ |
|---|---|---|
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `val`: | 書き込むデータ |

- `int32 adv_dio_write_float64v (AdvDocument* doc, adv_off_t offset, int num, const float64* val)`

  Document `doc` の `offset` の位置に `num` 個の 64 ビット float 型データを書き込む。

| **戻り値** | | 書き込んだデータのサイズ |
|---|---|---|
| **引数** | `doc`: | 書き込む Document |
| | `offset`: | マスデータ内の書き込み位置 |
| | `num`: | データの個数 |
| | `val`: | 書き込むデータ |

# 7 Databox 関連

- `AdvDatabox* adv_dbox_new(void)`
  AdvDocment をいれるデータボックスを開く。

| | | |
|---|---|---|
| **戻り値** | | 開いた AdvDatabox のポインタ |
| **引数** | | なし |

- `bool adv_dbox_add(AdvDatabox* adb, const char* locator)`
  `locator` が示すファイル内の Document をデータボックス `adb` に格納する。

| | | |
|---|---|---|
| **戻り値** | | 正常時：0 以外の値、エラー時：値 0 |
| **引数** | `adb`: | AdvDocument を格納する AdvDatabox |
| | `locator`: | AdvDocument を含んだファイル名 |

- `void adv_dbox_close(AdvDatabox* adb)`
  AdvDatabox `adb` をクローズする。

| | | |
|---|---|---|
| **戻り値** | | なし |
| **引数** | `adb`: | 閉じる AdvDatabox |

- `AdvDocument* adv_dbox_find_by_documentid(AdvDatabox* adb, const char* docid)`
  AdvDatabox `adb` から DocumentID が `docid` の Document をオープンする。

| | | |
|---|---|---|
| **戻り値** | | 該当する AdvDocument のポインタ |
| **引数** | `adb`: | AdvDatabox |
| | `docid`: | Document ID を表す文字列 |

- `AdvDocument* adv_dbox_find_by_property(AdvDatabox* adb,AdvDocument* prev, ...)`
  AdvDatabox `adb` の中から、指定した property にマッチする Document を検索する。`prev` と `...` は `adv_dio_open_by_property`(2 ページ) に同じ。

| | | |
|---|---|---|
| **戻り値** | | 該当する AdvDocument のポインタ |
| **引数** | `adb`: | 検索先の AdvDatabox |
| | `prev`: | `...` の条件にマッチする Document |
| | `...`: | 検索条件 |

- `int adv_dbox_count_by_property(AdvDatabox* adb, ...)`
  AdvDatabox adb 中の指定した property にマッチする Document の数を数える。

| | | |
|---|---|---|
| **戻り値** | | 該当する Document の個数 |
| **引数** | adb: | 検索先の AdvDatabox |
| | ...: | 検索条件<br>指定の仕方は<br>adv_dio_open_by_property(2 ページ) に同じ |

- `AdvDocument* adv_dbox_open_nth(AdvDatabox* adb, int n)`
  AdvDatabox adb の中で n 番目に記録されている Document を開く。

| | | |
|---|---|---|
| **戻り値** | | 該当する AdvDocument のポインタ |
| **引数** | adb: | 検索先の AdvDatabox |
| | n: | 整数 |

**使用例**：
adb に格納されている全ての Document の Document-ID を表示する。

```
void main(int argc,char* argv[]){

 int i=0;
 AdvDatabox *adb;
 AdvDocument *doc;

 adb = adv_dbox_new();
 adv_dbox_add(adb, "test.adv");
 while( (doc = adv_dbox_open_nth(adb, i++)) != NULL )
  fprintf(stderr,"%s\n",adv_dio_get_documentid(doc));
       /* adv_dio_get_documentid(doc) :
                          doc の Document-ID を返す */
```

**出力結果：**

```
6B8B4567:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
643C9869:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
74B0DC51:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
2AE8944A:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
238E1F29:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
3D1B58BA:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
```

```
2EB141F2:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
79E2A9E3:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
515F007C:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
12200854:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
216231B:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
1190CDE7:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
140E0F76:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
109CF92E:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
7FDCC233:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
41A7C4C9:HDDM_FEGA@HDDM_Part[0]:190F:39B4FB9B
4E6AFB66:DocumentList@HDDM_Part[0]:190F:39B4FB9B
```

(adb の全ての Document の ID が表示された)

# 8 その他

- `void adv_dio_copy_to_file (AdvDocFile* dfile, AdvDocument* doc)`
  `doc` の指す Document を `dfile` の指す AdvFile にコピーする。

| | | |
|---|---|---|
| **戻り値** | | なし |
| **引数** | `dfile`: | コピー先 AdvFile |
| | `doc`: | コピー元 Document |

- `int adv_format_get_size(const char* format)`
  文字列 `format` で示されるフォーマットにより使用される データサイズを返す。`format` はマスデータのフォーマットを表すために `property` に記述される文字列であり、 `i1`、`i2`、`i4`、`i8`、`f4`、`f8` の組合せで表される。

| | | |
|---|---|---|
| **戻り値** | | `format` により表されるデータのサイズ<br>(文字列 `format` がマスデータのフォーマットを表す<br>正しい文字列で無い場合は -1 をかえす) |
| **引数** | `format`: | マスデータ の フォーマットを表す文字列 |

**使用例**：

```
int bytes1,bytes2;

bytes1 = adv_format_get_size("i4f8f8");
bytes2 = adv_format_get_size("int32_gata");
printf("size of format = %d\n",bytes1);
printf("size of int32_gata = %d\n",bytes2);
```

**出力結果**：

```
size of i4f8f8 = 20
size of int32_gata = -1
```

("int32_gata" はマスデータのフォーマットを表す正しい文字列ではないので -1 が返された。)

- `bool adv_format_pack(octet* buf, const char* format, ...)`
  フォーマット `format` により表されるデータを octet 型データ配列 `buf` にパックする。

| | | |
|---|---|---|
| **戻り値** | | 正常時：0 以外の値、エラー時：値 0 |
| **引数** | buf: | データ配列の格納先 (オクテットデータ配列) |
| | format: | Raw Data のフォーマットを表す文字列 |
| | ...: | データの並び |

- `bool adv_format_pack_v(octet* buf, const char* format, va_list va)`
  可変引数リスト va 中のデータを format にあわせて octet 型データ配列 buf にパックする。

| | | |
|---|---|---|
| **戻り値** | | 正常時：0 以外の値、エラー時：値 0 |
| **引数** | buf: | データ配列の格納先 (オクテットデータ配列) |
| | format: | マスデータのフォーマットを表す文字列 |
| | va: | データを示す可変引数リスト |

- `bool adv_format_unpack(octet* buf, const char* format, ...)`
  パックされた octet 型データ配列 buf から format にしたがってデータをアンパックする。

| | | |
|---|---|---|
| **戻り値** | | 正常時：0 以外の値、エラー時：値 0 |
| **引数** | buf: | パックされたデータ配列 |
| | format: | フォーマットを表す文字列 |
| | ...: | データを格納する変数のアドレスのならび |

# 索 引