＜＜ 構造体記述仕様　＞＞

◆値

- 0　　: 整数
- 0x1F, 0X1F　　: 16 進数
- b101, B101　　: 2 進数

◆式サポート演算子

+, - ,*, /, %, ==, !=, <, <=, >,>=, ? : (三項演算子)

◆sizeof 演算子

その式より前で定義された別のメンバーのサイズを参照することができます。

sizeof(メンバー名)

◆文字エンコード

文字列のエンコード指定識別子は以下の 6 種類です。

- ASCII
- SJIS
- JIS
- EUC
- UTF8
- Unicode

◆基本型

- int8 : 8 ビットの符号あり整数
- uint8 : 8 ビットの符号なし整数
- int16 : 16 ビットの符号あり整数
- uint16 : 16 ビットの符号なし整数
- int32 : 32 ビットの符号あり整数
- uint32 : 32 ビットの符号なし整数
- int64 : 64 ビットの符号あり整数
- uint64 : 64 ビットの符号なし整数
- float : 32 ビットの実数
- double : 64 ビットの実数
- string(head8, 文字エンコード) : 8 ビットの文字バイト数ヘッダ付き文字列
  文字エンコードは省略できます。
- string(head16, 文字エンコード) : 16 ビットの文字バイト数ヘッダ付き文字列
  文字エンコードは省略できます。
- string(head32, 文字エンコード) : 32 ビットの文字バイト数ヘッダ付き文字列
  文字エンコードは省略できます。
- string(null, 文字エンコード) : Null 文字ターミネイト文字列
  文字エンコードは省略できます。

string(fix(バイト数式), 文字エンコード) : 固定バイト数 Null 文字ターミネイト文字列
固定バイト数には式やそのメンバーより前で定義された別のメンバーを参照することができます。
文字エンコードは省略できます。

- function(スクリプト関数名) : スクリプト関数コール型(後述)
- script(スクリプト関数名) : スクリプト処理型(後述)

◆構造体

C 言語と同じように以下のように記述します。
構造体はネスとして定義できます。

```
struct (構造体名) {
        (メンバー);
        (メンバー);
        struct (構造体名) {
                (メンバー);
        };
};
```

◆構造体メンバー

これも C 言語と同じように記述します。
配列要素数には式やそのメンバーより前で定義された別のメンバーを参照することができます。
ビットフィールドにも対応しています。
前述しましたが構造体もメンバーに含めることができ、構造体の配列の記述も可能です。

```
(基本型)        (識別子);
(基本型)        (識別子)[(配列要素数式)];
(基本型)        (識別子) : (ビット数);
struct (構造体名) {
        (メンバー);
};
struct (構造体名) {
        (メンバー);
}[(配列要素数式)];
```

◆スクリプト型

スクリプト型は２種類ありこれらを駆使することによって、より複雑な解析が可能となっています。
スクリプトの使用については別途「スクリプト仕様書」を参照してください。

- **function(スクリプト関数名)**

関数の引数にはそのメンバーのアドレスが入ります。
戻り値として配列を返します。
配列の０要素目は構造体表に表示する文字列、１要素目はそのメンバーのサイズを返します。

- **script(スクリプト関数名)**

関数の引数にはそのメンバーのアドレスが入ります。
戻り値としてそのメンバーのサイズを返します。
この型はスクリプト内で構造体表に表示する項目を関数コールによって実装します。
構造体表に表示する項目の完全なカスタマイズが可能となっています。

◆特殊コマンド

- **%script ～ %%**

「%script」と「%%」の間にスクリプトを記述します。

- **構造体内記述コマンド**
  - **defaultEncoding　　（文字エンコード）**
    文字列型で文字エンコードを省略したときに使用されるデフォルトの文字エンコードを指定します。

  以下のコマンドはバイナリファイルが不正な値なっていた場合、アプリケーションがクラッシュしないためのセキュリティ設定です。
  - **maxArrayCount　　（数値式）**
    構造体以外の配列の最大個数を指定します。
    これを超えるとエラーになります。
  - **maxArrayShowCount　　（数値式）**
    構造体以外の配列の表示最大個数を指定します。
    これを超えると表示が省略されます。
  - **maxStructArrayCount　　（数値式）**
    構造体の配列の最大個数を指定します。
    これを超えるとエラーになります。
  - **maxStringLength　　（数値式）**
    文字列の最大バイト数を指定します。
    これを超えるとエラーを返します。

◆typedef 構文

構造体外部で記述し、既存の型の別名を定義することができます。

typedef　（基本型）（型別名）;

◆（構造体例）

```
typedef int8 char;
typedef uint8 uchar;
typedef string(fix(16):UTF8) stringf16;

%script

function Test(addr) {
        var bytes = getBytes(addr, 3);
        var v = bytes[0] | (bytes[1] << 8) | (bytes[2] << 16);
        return [v, 3];
}

function Test2(addr) {
        var bytes = getBytes(addr, 4);
        var v = bytes[0] | (bytes[1] << 8) | (bytes[2] << 16);
        addItem("AA", (string)v, addr, 3);
        addItem("BB", (string)bytes[3], addr + 3, 1);
        addItem("Size", (string)fileSize, addr, 4);
        return 4;
}

%%

struct Main {

        maxArrayCount           1024;
        maxArrayShowCount       8;
        maxStructArrayCount     10;
        maxStringLength         256;

        defaultEncoding         SJIS;
```

```
        char    A;
        int8    Array[7];
        int16   B;
        int32   C;
        int64   D;
        float   E;
        double  F;

        int16   Count;
        struct Inner {

                string(null)            StringA;
                stringf16               StringB[2];

                uint16                  B;
                struct Inner2 {
                        string(head16:UTF8)     StringC;
                        uchar                   [32 - sizeof(StringC) - 2)];
                }[B];

                uint8                   Bit1 : 1;
                uint8                   Bit2 : 2;
                uint8                   Bit3 : 3;
                uint8                   Bit4 : 5;

                function(Test)          Func;
                script(Test2);

        }[Count];

};
```