Read Write

Read Write

ファイルを読んだり
ファイルを書いたり

Read

ファイルを読む。
三種類紹介。
上２つは数値、下は文字列を扱う。

```
BinaryReader r = new BinaryReader(File.OpenRead(dir));
int a = r.ReadInt32();
r.Close();


FileStream reader = new FileStream(dir, FileMode.Open, FileAccess.Read);
 byte a=reader.ReadByte();
reader.Close();

StreamReader sr = new StreamReader(dir, System.Text.Encoding.GetEncoding("SHIFT-JIS"), false);
String a = sr.ReadLine();
sr.Close();
```

Read ①

```
BinaryReader r = new BinaryReader(File.OpenRead(dir));
int a = r.ReadInt32();
r.Close();
```

```
BinaryReader r = new BinaryReader(File.OpenRead(dir));

int a = r.ReadInt32();

r.Close();
```

任意の名前

ファイルの場所（string型）

読んだデータを入れる変数。型は任意。
r.ReadByte(); → byte
r.ReadInt16(); → 2バイト整数
r.ReadInt32(); → 4バイト整数（int型）
r.ReadSingle(); → float型　　など。

説明していないところは呪文

Read ①

```
BinaryReader r = new BinaryReader(File.OpenRead(dir));
int a = r.ReadInt32();
r.Close();
```

Sample code

```
BinaryReader r = new BinaryReader(File.OpenRead(dir));
for(int i = 0;i<100;i++)
{
    a[i] = r.ReadInt16();
}
r.Close();
```

dir はstring型でファイルの場所を指している。
2バイト読んで a にいれる。
必ず close する。

Read ②

```
FileStream reader = new FileStream(dir, FileMode.Open, FileAccess.Read);
 byte a=reader.ReadByte();
reader.Close();
```

```
FileStream reader = new FileStream(dir, FileMode.Open, FileAccess.Read);

byte a=reader.ReadByte();

reader.Close();
```

任意の名前

ファイルの場所（string型）

読んだデータを入れる変数。
型はbyte。

説明していないところは呪文

Read ②

```
FileStream reader = new FileStream(dir, FileMode.Open, FileAccess.Read);
 byte a=reader.ReadByte();
reader.Close();
```

Sample code

```
FileStream reader = new FileStream(ofd.FileName, FileMode.Open, FileAccess.Read);
for (frame = 1; frame <= maxframe; frame++)
{
    for (k = 0; k < 64; k++)
    {
        for (j = 0; j < 64; j++)
        {
            for (i = 0; i < 64; i++)
            {
                byte c = reader.ReadByte();
                byte d = reader.ReadByte();
                xx[i, k, j, frame] = c * 256 + d;
                if (xx[i, k, j, frame] > max0) max0 = xx[i, k, j, frame];
            }
        }
    }
}
reader.Close();
```

ofd.FileName はstring型でファイルの場所を指している。
他はC++ bone プログラムと見比べればわかりやすいかも。
必ず close する。

Read ③

```
StreamReader sr = new StreamReader(dir, System.Text.Encoding.GetEncoding("SHIFT-JIS"), false);
String a = sr.ReadLine();
sr.Close();
```

StreamReader sr = new StreamReader(dir, System.Text.Encoding.GetEncoding("SHIFT-JIS"), false);

String a = sr.ReadLine();

sr.Close();

任意の名前

ファイルの場所（string型）

読んだデータを入れる変数。型はstring型。
ReadLine は一行をstring型で読む。

説明していないところは呪文

Read ③

```
StreamReader sr = new StreamReader(dir, System.Text.Encoding.GetEncoding("SHIFT-JIS"), false);
String a = sr.ReadLine();
sr.Close();
```

Sample code

```
xt = new int[maxframe + 1];

StreamReader sr = new StreamReader("C:¥¥HeartAce¥¥PETT21.prn", System.Text.Encoding.GetEncoding("SHIFT-JIS"), false);
for (frame = 1; frame <= maxframe; frame++)
{
    xt[frame] = int.Parse(sr.ReadLine());
}
sr.Close();
```

int.Parse() は、string型をint型に（キャスト）する。
一行読んで xt にいれる。
結果
xt[1] = 5,
xt[2] = 15, …
と入っていく。
必ずclose

読んだファイル

	A	B
1	5	
2	15	
3	25	
4	35	
5	45	
6	55	
7	65	
8	75	
9	85	
10	95	
11	130	
12	210	
13	310	
14	410	
15	510	
16	610	
17	750	
18	930	
19	1110	
20	1350	
21	1650	
22		

Write

ファイルを書く。
三種類紹介。
上2つは数値、下は文字列を扱う。

```
BinaryWriter w = new BinaryWriter(File.Create(dir));
w.Write(data);
w.Close();
```

```
FileStream writer = new FileStream(dir, FileMode.Create, FileAccess.Write);
byte a = data;
writer.WriteByte(a);
writer.Close();
```

```
StreamWriter sw = new StreamWriter(dir, false, System.Text.Encoding.GetEncoding("SHIFT-JIS"));
sw.WriteLine(data);
sw.Close();
```

Write ①

```
BinaryWriter w = new BinaryWriter(File.Create(dir));
w.Write(data);
w.Close();
```

```
BinaryWriter w = new BinaryWriter(File.Create(dir));

w.Write(data);

w.Close();
```

任意の名前

ファイルの場所（string型）

書きこみたいデータ。型は任意。
読み取るときは同じ型で。

説明していないところは呪文

Write ①

```
BinaryWriter w = new BinaryWriter(File.Create(dir));
w.Write(data);
w.Close();
```

Sample code

```
BinaryWriter w = new BinaryWriter(File.Create(dir));
for (int i = 0; i < 100; i++)
{
    w.Write(x[i]);
}
w.Close();
```

dir（保存したい場所と名前。string型）にファイルを作る、もしくは上書きする。

Write ②

```
FileStream writer = new FileStream(dir, FileMode.Create, FileAccess.Write);
byte a = data;
writer.WriteByte(a);
writer.Close();
```

```
FileStream writer = new FileStream(dir, FileMode.Create, FileAccess.Write);

byte a = (byte)data;

writer.WriteByte(a);

writer.Close();
```

任意の名前

ファイルの場所（string型）

書きこみたいデータ。型はバイト。
バイト型で扱っていなかったものはキャストする必要がある。
たぶんwriter.WriteByte((byte)data);でも可。

説明していないところは呪文

Write ②

FileStream writer = new FileStream(dir, FileMode.Create, FileAccess.Write);
byte a = data;
writer.WriteByte(a);
writer.Close();

Sample code

```
FileStream writer = new FileStream(acedir + "ACEsubCO", FileMode.Create, FileAccess.Write);
for (k = 0; k < 64; k++)
{
    for (j = 0; j < 64; j++)
    {
        for (i = 0; i < 64; i++)
        {
            byte c = (byte)(ACEsubCO[i, j, k] / 256);
            byte d = (byte)(ACEsubCO[i, j, k] - c * 256);
            writer.WriteByte(c);
            writer.WriteByte(d);
        }
    }
}
```

指定した場所にファイルを作る、もしくは上書きする。
bone をみながらだとわかりやすいかも。

Write ③

```
StreamWriter sw = new StreamWriter(dir, false, System.Text.Encoding.GetEncoding("SHIFT-JIS"));
sw.WriteLine(data);
sw.Close();
```

```
StreamWriter sw = new StreamWriter(dir, false, System.Text.Encoding.GetEncoding("SHIFT-JIS"));

sw.WriteLine(data);

sw.Close();
```

任意の名前

ファイルの場所（string型）

書きこみたいデータ。型はstring。
WriteLineで一行書きこんで改行する。

説明していないところは呪文

Write ③

```
StreamWriter sw = new StreamWriter(dir, false, System.Text.Encoding.GetEncoding("SHIFT-JIS"));
sw.WriteLine(data);
sw.Close();
```

Sample code

```
StreamWriter sw = new StreamWriter(dir+"¥¥sensor16.prn", false, System.Text.Encoding.GetEncoding("SHIFT-JIS"));
for (i = 0; i <= 100; i++)
{
    buf += "¥t" + (i + 1).ToString();
}
sw.WriteLine(buf);
for (j = 0; j < 3 * 16; j++)
{
    buf = "";
    int k = j / 3;
    int jj = j - k * 3;
    if (jj == 0) buf = "x" + (k+1).ToString();
    if (jj == 1) buf = "y" + (k+1).ToString();
    if (jj == 2) buf = "z" + (k+1).ToString();
    for (i = 0; i <= 100; i++)
    {
        switch (jj)
        {
            case 0:
                buf += "¥t" + xp16[i, k].ToString();
                break;

            case 1:
                buf += "¥t" + yp16[i, k].ToString();
                break;

            case 2:
                buf += "¥t" + zp16[i, k].ToString();
                break;
        }
    }
    sw.WriteLine(buf);
}
sw.Close();
```

人に見せるためのエクセルデータ作成などに重宝すると思う。
.Tostring();によりstring型にキャストさせる。
¥t はタブ。
色々書いてあるが、上の6行
sw.WriteLine(buf);までで下の1行目ができる。

	A	B	C	D	E	F	G	H	I	J	K	L
1		1	2	3	4	5	6	7	8	9	10	11
2	x1	-0.11341	-0.10202	-0.08439	-0.12074	-0.16074	-0.11511	-0.10137	-0.13627	-0.11479	-0.10882	-0.10096
3	y1	-0.03287	-0.05955	-0.08254	-0.00557	-0.00477	-0.03689	-0.06579	-0.08529	-0.00363	-0.03655	0.005546
4	z1	-0.02637	-0.02637	-0.02637	0.006072	0.005472	0.006072	0.006072	0.005472	0.038153	0.038153	0.066419
5	x2	-0.11201	-0.10074	-0.08331	-0.12067	-0.16067	-0.11504	-0.10131	-0.13621	-0.1161	-0.11008	-0.10348
6	y2	-0.03256	-0.05891	-0.08161	-0.00557	-0.00477	-0.03688	-0.06576	-0.08526	-0.00358	-0.03688	0.005877
7	z2	-0.03128	-0.03128	-0.03128	0.000953	0.000353	0.000953	0.000953	0.000353	0.033203	0.033203	0.061971
8	x3	-0.11061	-0.09946	-0.08222	-0.1206	-0.1606	-0.11497	-0.10124	-0.13614	-0.11741	-0.11135	-0.10599
9	y3	-0.03225	-0.05826	-0.08067	-0.00557	-0.00477	-0.03686	-0.06572	-0.08522	-0.00354	-0.03722	0.006208

動かしていないコードもあるので間違いあるかも。

調べれば色々のっているので探してみてください。

お気づきかと思いますが、FileStreamはいらないかもです。

おわり