

## NAME

FingerprintsSDFFileIO

## SYNOPSIS

```
use FileIO::FingerprintsSDFFileIO;

use FileIO::FingerprintsSDFFileIO qw(:all);
```

## DESCRIPTION

FingerprintsSDFFileIO class provides the following methods:

new, GetCompoundString, GetFingerprints, GetFingerprintsString, IsFingerprintsDataValid, IsFingerprintsFileDataValid, IsFingerprintsSDFFile, Next, Read, SetBitStringFormat, SetBitsOrder, SetCompoundIDMode, SetCompoundString, SetDetailLevel, SetFingerprints, SetFingerprintsString, SetFingerprintsStringMode, SetVectorStringFormat, WriteFingerprints, WriteFingerprintsString

The following methods can also be used as functions:

IsFingerprintsSDFFile

FingerprintsSDFFileIO class is derived from *FileIO* class and uses its methods to support generic file related functionality.

The fingerprints SD file format with .sdf or .sd file extensions supports two types of fingerprints string data: fingerprints bit-vectors and fingerprints vector strings. The fingerprints string data is treated as value of a fingerprints data field label in a SD file.

Example of SD file format containing fingerprints string data:

```
... ..
... ..
$$$$
... ..
... ..
... ..
41 44 0 0 0 0 0 0 0 0 0999 V2000
-3.3652 1.4499 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0
... ..
2 3 1 0 0 0 0
... ..
M END
> <CmpdID>
Test

> <PathLengthFingerprints>
FingerprintsBitVector;PathLengthBits:AtomicInvariantsAtomTypes:MinLength:MaxLength8;1024;HexadecimalString;Ascending;9c8460989ec8a49913991a6603130b0a19e8051c89184414953800cc2151082844a201042800130860308e8204d402800831048940e44281c00060449a5000ac80c894114e006321264401600846c05016446208190410805000304a10205b0100e04c0038ba0fad0209c0ca8b1200012268b61c0026a
aa0660a11014a011d46

$$$$
... ..
... ..
```

The current release of MayaChemTools supports the following types of fingerprint bit-vector and vector strings:

```
FingerprintsVector;AtomNeighborhoods:AtomicInvariantsAtomTypes:MinRadius:MaxRadius2;41;AlphaNumericalValues;ValuesString;NR0-C.X1.BO1.H3-ATC1:NR1-C.X3.BO3.H1-ATC1:NR2-C.X1.BO1.H3-ATC1:NR2-C.X3.BO4-ATC1 NR0-C.X1.BO1.H3-ATC1:NR1-C.X3.BO3.H1-ATC1:NR2-C.X1.BO1.H3-ATC1:NR2-C.X3.BO4-ATC1 NR0-C.X2.BO2.H2-ATC1:NR1-C.X2.BO2.H2-ATC1:NR1-C.X3.BO3.H1-ATC1:NR2-C.X2.BO2.H2-ATC1:NR2-N.X3.BO3-ATC1:NR2-O.X1.BO1.H1-ATC1 NR0-C.X2.B...
```

```
FingerprintsVector;AtomTypesCount:AtomicInvariantsAtomTypes:ArbitrarySize;10;NumericalValues;IDsAndValuesString;C.X1.BO1.H3 C.X2.BO2.H2 C.X2.BO3.H1 C.X3.BO3.H1 C.X3.BO4 F.X1.BO1 N.X2.BO2.H1 N.X3.BO3 O.X1.BO1.H1
```



```
FingerprintsBitVector;MACCSKeyBits;322;BinaryString;Ascending;11101011
1110011111100101111111000111101100110000000000000011100010000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
```

```
FingerprintsVector;MACCSKeyCount;166;OrderedNumericalValues;ValuesStri
ng;0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 0 3 0 0 0 0 4 0 0 2 0 0 0 0 0 0 0 2 0 0 2 0 0 0 0
0 0 0 0 1 1 8 0 0 0 1 0 0 1 0 1 0 3 1 3 1 0 0 0 1 2 0 11 1 0 0 0
5 0 0 1 2 0 1 1 0 0 0 0 0 1 1 0 1 1 1 1 0 4 0 0 1 1 0 4 6 1 1 1 2 1 1
3 5 2 2 0 5 3 5 1 1 2 5 1 2 1 2 4 8 3 5 5 2 2 0 3 5 4 1
```

```
FingerprintsVector;MACCSKeyCount;322;OrderedNumericalValues;ValuesStri
ng;14 8 2 0 2 0 4 4 2 1 4 0 0 2 5 10 5 2 1 0 0 2 0 5 13 3 28 5 5 3 0 0
0 4 2 1 1 0 1 1 0 0 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 22 5 3 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 11 0 2 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...
```

```
FingerprintsBitVector;PathLengthBits:AtomicInvariantsAtomTypes:MinLeng
th1:MaxLength8;1024;BinaryString;Ascending;001000010011010101011000110
0100010101011000101001011100110001000010001001101000001001001001000
0010110100000111001001000001001010100100100000000011000000101001011100
0010000001000101010100000100111100110111011011011000000010110111001101
010110001100000001000100001100001010001110110000100000100010000000...
```

```
FingerprintsVector;PathLengthCount:AtomicInvariantsAtomTypes:MinLength
1:MaxLength8;432;NumericalValues;IDsAndValuesPairsString;C.X1.BO1.H3 2
C.X2.BO2.H2 4 C.X2.BO3.H1 14 C.X3.BO3.H1 3 C.X3.BO4 10 F.X1.BO1 1 N.X
2.BO2.H1 1 N.X3.BO3 1 O.X1.BO1.H1 3 O.X1.BO2 2 C.X1.BO1.H3C.X3.BO3.H1
2 C.X2.BO2.H2C.X2.BO2.H2 1 C.X2.BO2.H2C.X3.BO3.H1 4 C.X2.BO2.H2C.X3.BO
4 1 C.X2.BO2.H2N.X3.BO3 1 C.X2.BO3.H1:C.X2.BO3.H1 10 C.X2.BO3.H1:C...
```

```
FingerprintsVector;PathLengthCount:MMFF94AtomTypes:MinLength1:MaxLengt
h8;463;NumericalValues;IDsAndValuesPairsString;C5A 2 C5B 2 C=ON 1 CB 1
8 COO 1 CR 9 F 1 N5 1 NC=O 1 O=CN 1 O=CO 1 OC=O 1 OR 2 C5A:C5B 2 C5A:N
5 2 C5ACB 1 C5ACR 1 C5B:C5B 1 C5BC=ON 1 C5BCB 1 C=ON=O=CN 1 C=ONNC=O 1
CB:CB 18 CBF 1 CBNC=O 1 COO=O=CO 1 COOCR 1 COOOC=O 1 CRCR 7 CRN5 1 CR
OR 2 C5A:C5B:C5B 2 C5A:C5BC=ON 1 C5A:C5BCB 1 C5A:N5:C5A 1 C5A:N5CR ...
```

```
FingerprintsVector;TopologicalAtomPairs:AtomicInvariantsAtomTypes:MinDi
stancel:MaxDistance10;223;NumericalValues;IDsAndValuesString;C.X1.BO1
.H3-D1-C.X3.BO3.H1 C.X2.BO2.H2-D1-C.X2.BO2.H2 C.X2.BO2.H2-D1-C.X3.BO3.
H1 C.X2.BO2.H2-D1-C.X3.BO4 C.X2.BO2.H2-D1-N.X3.BO3 C.X2.BO3.H1-D1-...;
2 1 4 1 1 10 8 1 2 6 1 2 2 1 2 1 2 2 1 2 1 5 1 10 12 2 2 1 2 1 9 1 3 1
1 1 2 2 1 3 6 1 6 14 2 2 2 3 1 3 1 8 2 2 1 3 2 6 1 2 2 5 1 3 1 23 1...
```

```
FingerprintsVector;TopologicalAtomPairs:FunctionalClassAtomTypes:MinDi
stancel:MaxDistance10;144;NumericalValues;IDsAndValuesString;Ar-D1-Ar
Ar-D1-Ar.HBA Ar-D1-HBD Ar-D1-Hal Ar-D1-None Ar.HBA-D1-None HBA-D1-NI H
BA-D1-None HBA.HBD-D1-NI HBA.HBD-D1-None HBD-D1-None NI-D1-None No...;
23 2 1 1 2 1 1 1 2 1 1 7 28 3 1 3 2 8 2 1 1 1 5 1 5 24 3 3 4 2 13 4
1 1 4 1 5 22 4 4 3 1 19 1 1 1 1 1 2 2 3 1 1 8 25 4 5 2 3 1 26 1 4 1 ...
```

```
FingerprintsVector;TopologicalAtomTorsions:AtomicInvariantsAtomTypes;3
3;NumericalValues;IDsAndValuesString;C.X1.BO1.H3-C.X3.BO3.H1-C.X3.BO4-
C.X3.BO4 C.X1.BO1.H3-C.X3.BO3.H1-C.X3.BO4-N.X3.BO3 C.X2.BO2.H2-C.X2.BO
2.H2-C.X3.BO3.H1-C.X2.BO2.H2 C.X2.BO2.H2-C.X2.BO2.H2-C.X3.BO3.H1-O...;
2 2 1 1 2 2 1 1 3 4 4 8 4 2 2 6 2 2 1 2 1 1 2 1 1 2 6 2 4 2 1 3 1
```

```
FingerprintsVector;TopologicalAtomTorsions:EStateAtomTypes;36;Numerica
lValues;IDsAndValuesString;aaCH-aaCH-aaCH-aaCH aaCH-aaCH-aaCH-aasC aaC
H-aaCH-aasC-aaCH aaCH-aaCH-aasC-aasC aaCH-aaCH-aasC-sF aaCH-aaCH-aasC-
```

```
ssNH aaCH-aasC-aasC-aasC aaCH-aasC-aasC-aasN aaCH-aasC-ssNH-dssC a...;
4 4 8 4 2 2 6 2 2 2 4 3 2 1 3 3 2 2 2 1 2 1 1 1 2 1 1 1 1 1 1 1 2 1 1 2
```

```
FingerprintsVector;TopologicalAtomTriplets:AtomicInvariantsAtomTypes:MinDistance1:MaxDistance10;3096;NumericalValues;IDsAndValuesString;C.X1.B01.H3-D1-C.X1.B01.H3-D1-C.X3.B03.H1-D2 C.X1.B01.H3-D1-C.X2.B02.H2-D10-C.X3.B04-D9 C.X1.B01.H3-D1-C.X2.B02.H2-D3-N.X3.B03-D4 C.X1.B01.H3-D1-C.X2.B02.H2-D4-C.X2.B02.H2-D5 C.X1.B01.H3-D1-C.X2.B02.H2-D6-C.X3...;
1 2 2 2 2 2 2 8 8 4 8 4 4 2 2 2 4 2 2 2 4 2 2 2 1 2 2 4 4 4 2 2
2 4 4 4 8 4 4 2 4 4 4 2 4 4 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 8...
```

```
FingerprintsVector;TopologicalAtomTriplets:SYBYLAtomTypes:MinDistance1:MaxDistance10;2332;NumericalValues;IDsAndValuesString;C.2-D1-C.2-D9-C.3-D10 C.2-D1-C.2-D9-C.ar-D10 C.2-D1-C.3-D1-C.3-D2 C.2-D1-C.3-D10-C.3-D9 C.2-D1-C.3-D2-C.3-D3 C.2-D1-C.3-D2-C.ar-D3 C.2-D1-C.3-D3-C.3-D4 C.2-D1-C.3-D3-N.ar-D4 C.2-D1-C.3-D3-O.3-D2 C.2-D1-C.3-D4-C.3-D5 C.2-D1-C.3-D5-C.3-D6 C.2-D1-C.3-D5-O.3-D4 C.2-D1-C.3-D6-C.3-D7 C.2-D1-C.3-D7...
```

```
FingerprintsVector;TopologicalPharmacophoreAtomPairs:ArbitrarySize:MinDistance1:MaxDistance10;54;NumericalValues;IDsAndValuesString;H-D1-H H-D1-NI HBA-D1-NI HBD-D1-NI H-D2-H H-D2-HBA H-D2-HBD HBA-D2-HBA HBA-D2-HBD H-D3-H H-D3-HBA H-D3-HBD H-D3-NI HBA-D3-NI HBD-D3-NI H-D4-H H-D4-HBA H-D4-HBD HBA-D4-HBA HBA-D4-HBD HBD-D4-HBD H-D5-H H-D5-HBA H-D5-...;
18 1 2 1 22 12 8 1 2 18 6 3 1 1 1 22 13 6 5 7 2 28 9 5 1 1 1 36 16 10
3 4 1 37 10 8 1 35 10 9 3 3 1 28 7 7 4 18 16 12 5 1 2 1
```

```
FingerprintsVector;TopologicalPharmacophoreAtomPairs:FixedSize:MinDistance1:MaxDistance10;150;OrderedNumericalValues;ValuesString;18 0 0 1 0
0 0 2 0 0 1 0 0 0 0 22 12 8 0 0 1 2 0 0 0 0 0 0 0 0 18 6 3 1 0 0 0 1
0 0 1 0 0 0 0 22 13 6 0 0 5 7 0 0 2 0 0 0 0 0 28 9 5 1 0 0 0 1 0 0 1 0
0 0 0 36 16 10 0 0 3 4 0 0 1 0 0 0 0 0 37 10 8 0 0 0 0 1 0 0 0 0 0 0
0 35 10 9 0 0 3 3 0 0 1 0 0 0 0 0 28 7 7 4 0 0 0 0 0 0 0 0 0 0 0 18...
```

```
FingerprintsVector;TopologicalPharmacophoreAtomTriplets:ArbitrarySize:MinDistance1:MaxDistance10;696;NumericalValues;IDsAndValuesString;Ar1-Ar1-Ar1 Ar1-Ar1-H1 Ar1-Ar1-HBA1 Ar1-Ar1-HBD1 Ar1-H1-H1 Ar1-H1-HBA1 Ar1-H1-HBD1 Ar1-HBA1-HBD1 H1-H1-H1 H1-H1-HBA1 H1-H1-HBD1 H1-HBA1-HBA1 H1-HBA1-HBD1 H1-HBA1-NI1 H1-HBD1-NI1 HBA1-HBA1-NI1 HBA1-HBD1-NI1 Ar1-...;
46 106 8 3 83 11 4 1 21 5 3 1 2 2 1 1 1 100 101 18 11 145 132 26 14 23
28 3 3 5 4 61 45 10 4 16 20 7 5 1 3 4 5 3 1 1 1 5 4 2 1 2 2 2 1 1 1
119 123 24 15 185 202 41 25 22 17 3 5 85 95 18 11 23 17 3 1 1 6 4 ...
```

```
FingerprintsVector;TopologicalPharmacophoreAtomTriplets:FixedSize:MinDistance1:MaxDistance10;2692;OrderedNumericalValues;ValuesString;46 106
8 3 0 0 83 11 4 0 0 0 1 0 0 0 0 0 0 0 0 21 5 3 0 0 1 2 2 0 0 1 0 0 0
0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 100 101 18 11 0 0 145 132 26
14 0 0 23 28 3 3 0 0 5 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 61 45 10 4 0
0 16 20 7 5 1 0 3 4 5 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 5 ...
```

## METHODS

new

```
$NewFingerprintsSDFFileIO = new FileIO::FingerprintsSDFFileIO(%IOParameters);
```

Using specified *IOParameters* names and values hash, new method creates a new object and returns a reference to a newly created FingerprintsSDFFileIO object. By default, the following properties are initialized during *Read* mode:

```
Name = '';
Mode = 'Read';
Status = 0;
FingerprintsStringMode = 'AutoDetect';
FingerprintsFieldLabel = 'AutoDetect';
CompoundIDMode = 'LabelPrefix';
CompoundIDFieldLabel = undef;
CompoundIDPrefix = 'Cmpd';
ValidateData = 1;
```

```
DetailLevel = 1;
```

During *Write* mode, the following properties get initialize by default:

```
FingerprintsStringMode = undef;
```

```
BitStringFormat = HexadecimalString;
```

```
BitsOrder = Ascending;
```

```
VectorStringFormat = NumericalValuesString or ValuesString;
```

Examples:

```
$NewFingerprintsSDFFileIO = new FileIO::FingerprintsSDFFileIO(
    'Name' => 'Sample.sdf',
    'Mode' => 'Read');
```

```
$NewFingerprintsSDFFileIO = new FileIO::FingerprintsSDFFileIO(
    'Name' => 'Sample.sdf',
    'Mode' => 'Read',
    'FingerprintsStringMode' =>
        'AutoDetect',
    'FingerprintsFieldLabel' =>
        'Fingerprints',
    'CompoundIDMode' =>
        'DataField',
    'CompoundIDFieldLabel' =>
        'CompoundID');
```

```
$NewFingerprintsSDFFileIO = new FileIO::FingerprintsSDFFileIO(
    'Name' => 'Sample.sdf',
    'Mode' => 'Write',
    'FingerprintsStringMode' =>
        'FingerprintsBitVectorString',
    'Overwrite' => 1,
    'BitStringFormat' => 'HexadecimalString',
    'BitsOrder' => 'Ascending');
```

```
$NewFingerprintsSDFFileIO = new FileIO::FingerprintsSDFFileIO(
    'Name' => 'Sample.sd',
    'Mode' => 'Write',
    'FingerprintsStringMode' =>
        'FingerprintsVectorString',
    'Overwrite' => 1,
    'VectorStringFormat' => 'IDsAndValuesString',
    'FingerprintsLabel' => 'Fingerprints');
```

#### GetCompoundString

```
$CompoundString = $FingerprintsSDFFileIO->GetCompoundString();
```

Returns CompoundString for current compound.

#### GetFingerprints

```
$FingerprintsObject = $FingerprintsSDFFileIO->GetFingerprints();
```

Returns FingerprintsObject generated for current compound using fingerprints bit-vector or vector string data. The fingerprints object corresponds to any of the supported fingerprints such as PathLengthFingerprints, ExtendedConnectivity, and so on.

#### GetFingerprintsString

```
$FingerprintsString = $FingerprintsSDFFileIO->GetFingerprintsString();
```

Returns FingerprintsString for current compound.

#### IsFingerprintsDataValid

```
$Status = $FingerprintsSDFFileIO->IsFingerprintsDataValid();
```

Returns 1 or 0 based on whether FingerprintsObject is valid.

**IsFingerprintsFileDataValid**

```
$Status = $FingerprintsSDFFileIO->IsFingerprintsFileDataValid();
```

Returns 1 or 0 based on whether fingerprints file contains valid fingerprints data.

**IsFingerprintsSDFFile**

```
$Status = $FingerprintsSDFFileIO->IsFingerprintsSDFFile($FileName);  
$Status = FileIO::FingerprintsSDFFileIO::IsFingerprintsSDFFile($FileName);
```

Returns 1 or 0 based on whether *FileName* is a SD file.

**Next or Read**

```
$FingerprintsSDFFileIO = $FingerprintsSDFFileIO->Next();  
$FingerprintsSDFFileIO = $FingerprintsSDFFileIO->Read();
```

Reads next available compound fingerprints in SD file, processes the data, generates appropriate fingerprints object, and returns FingerprintsSDFFileIO. The generated fingerprints object is available using method GetFingerprints.

**SetBitStringFormat**

```
$FingerprintsSDFFileIO->SetBitStringFormat($Format);
```

Sets bit string *Format* for fingerprints bit-vector string data in a SD file and returns FingerprintsSDFFileIO. Possible values for BitStringFormat: *BinaryString* or *HexadecimalString*.

**SetBitsOrder**

```
$FingerprintsSDFFileIO->SetBitsOrder($BitsOrder);
```

Sets *BitsOrder* for fingerprints bit-vector string data in SD file and returns FingerprintsSDFFileIO. Possible values for BitsOrder: *Ascending* or *Descending*.

**SetCompoundIDMode**

```
$FingerprintsSDFFileIO->SetCompoundIDMode($Mode);
```

Sets compound ID *Mode* for fingerprints bit-vector string data in a SD file and returns FingerprintsSDFFileIO. Possible values for CompoundIDMode: *DataField*, *MolName*, *LabelPrefix*, or *MolNameOrLabelPrefix*.

**SetCompoundString**

```
$FingerprintsSDFFileIO->SetCompoundString($CompoundString);
```

Sets *CompoundString* and returns FingerprintsSDFFileIO.

**SetDetailLevel**

```
$FingerprintsSDFFileIO->SetDetailLevel($Level);
```

Sets details *Level* for generating diagnostics messages during SD file processing and returns FingerprintsSDFFileIO. Possible values: *Positive integers*.

**SetFingerprints**

```
$FingerprintsSDFFileIO->SetFingerprints($FingerprintsObject);
```

Sets *FingerprintsObject* for current data line and returns FingerprintsSDFFileIO.

**SetFingerprintsString**

```
$FingerprintsSDFFileIO->SetFingerprintsString($FingerprintsString);
```

Sets *FingerprintsString* for current data line and returns FingerprintsSDFFileIO.

**SetFingerprintsStringMode**

```
$FingerprintsSDFFileIO->SetFingerprintsStringMode($Mode);
```

Sets *FingerprintsStringMode* for SD file and returns FingerprintsFPFileIO. Possible values: *AutoDetect*, *FingerprintsBitVectorString*, or *FingerprintsVectorString*

**SetVectorStringFormat**

```
$FingerprintsSDFFileIO->SetVectorStringFormat($Format);
```

Sets *VectorStringFormat* for SD file and returns FingerprintsFPFileIO. Possible values: *IDsAndValuesString*, *IDsAndValuesPairsString*, *ValuesAndIDsString*, *ValuesAndIDsPairsString*.

#### WriteFingerprints

```
$FingerprintsFPFileIO->WriteFingerprints($FingerprintsObject,  
                                         $CompoundID);
```

Writes fingerprints string generated from *FingerprintsObject* object and other data including *CompoundID* to SD file and returns FingerprintsSDFFileIO.

#### WriteFingerprintsString

```
$FingerprintsSDFFileIO->WriteFingerprints($FingerprintsString,  
                                         $CompoundID);
```

Writes *FingerprintsString* and other data including *CompoundID* to SD file and returns FingerprintsSDFFileIO.

#### Caveats:

- o *FingerprintsStringMode*, *BitStringFormat*, *BitsOrder*, *VectorStringFormat* values are ignored during writing of fingerprints and it's written to the file as it is.
- o *CompoundString* is not checked to remove any existing fingerprints data

#### AUTHOR

Manish Sud <msud@san.rr.com>

#### SEE ALSO

FingerprintsTextFileIO.pm, FingerprintsFPFileIO.pm, SDFFileIO.pm

#### COPYRIGHT

Copyright (C) 2024 Manish Sud. All rights reserved.

This file is part of MayaChemTools.

MayaChemTools is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.