

## 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 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:MinRadius0: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 O.X1.BO2;2 4 14 3 10 1 1 1 3 2
```

```
FingerprintsVector;AtomTypesCount:SLogPAtomTypes:ArbitrarySize;16;NumericalValues;IDsAndValuesString;C1 C10 C11 C14 C18 C20 C21 C22 C5 CS F N11 N4 O10 O2 O9;5 1 1 1 14 4 2 1 2 2 1 1 1 1 3 1
```

```
FingerprintsVector;AtomTypesCount:SLogPAtomTypes:FixedSize;67;OrderedNumericalValues;IDsAndValuesString;C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 C11 C12 C13 C14 C15 C16 C17 C18 C19 C20 C21 C22 C23 C24 C25 C26 C27 CS N1 N
```

```
FingerprintsVector;MACCSKeyCount;322;OrderedNumericalValues;ValuesString;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 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
```



```
FingerprintsVector;TopologicalPharmacophoreAtomPairs:FixedSize:MinDistance: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 1 0
0 0 0 0 36 16 10 0 0 3 4 0 0 1 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 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:MinDistance: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 0 1 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 *IOPParameters* 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.

#### IsFingerprintsSDFile

```
$Status = $FingerprintsSDFileIO->IsFingerprintsSDFile($FileName);
$Status = FileIO::FingerprintsSDFileIO::IsFingerprintsSDFile($FileName);
```

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

#### Next or Read

```
$FingerprintsSDFileIO = $FingerprintsSDFileIO->Next();
$FingerprintsSDFileIO = $FingerprintsSDFileIO->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

```
$FingerprintsSDFileIO->SetBitStringFormat($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

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

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

#### SetCompoundIDMode

```
$FingerprintsSDFileIO->SetCompoundIDMode($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

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

Sets *CompoundString* and returns FingerprintsSDFFileIO.

#### SetDetailLevel

```
$FingerprintsSDFileIO->SetDetailLevel($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) 2018 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.