

## NAME

Fingerprints - Fingerprints class

## SYNOPSIS

```
use Fingerprints;

use Fingerprints qw(:all);
```

## DESCRIPTION

Fingerprints class provides the following methods:

`new`, `FoldFingerprintsByBitDensity`, `FoldFingerprintsBySize`, `GetFingerprintBitsAsBinaryString`, `GetFingerprintBitsAsHexadecimalString`, `GetFingerprintBitsAsRawBinaryString`, `GetFingerprintsVectorValueIds`, `GetFingerprintsVectorValues`, `IsFingerprintsGenerationSuccessful`, `SetFingerprintsBitVector`, `SetFingerprintsVector`, `SetFingerprintsVectorType`, `SetMolecule`, `SetSize`, `SetType`, `SetVectorType`

Fingerprints class is used as a base class for various specific fingerprint classes such as `AtomNeighborhoodsFingerprints`, `AtomTypesFingerprints`, `EStateIndicesFingerprints`, `PathLengthFingerprints`, `ExtendedConnectivityFingerprints`, `MACCSKeys` and so on. It implements functionality common to fingerprint classes.

Fingerprints class is derived from `ObjectProperty` base class which provides methods not explicitly defined in `Fingerprints` or `ObjectProperty` classes using Perl's `AUTOLOAD` functionality. These methods are generated on-the-fly for a specified object property:

```
Set<PropertyName>(<PropertyValue>);
$PropertyValue = Get<PropertyName>();
Delete<PropertyName>();
```

Fingerprints class uses `FingerprintsBitVector` class to provide bits manipulation functionality.

## METHODS

`new`

```
$NewFingerprints = new Fingerprints(%NamesAndValues);
```

Using specified *Fingerprints* property names and values hash, `new` method creates a new object and returns a reference to newly created `Fingerprints` object. By default, following properties are initialized:

```
Molecule = '';
Type = '';
VectorType = '';
Size = '';
MinSize = '';
MaxSize = '';
FingerprintsBitVector = '';
FingerprintsVectorType = '';
FingerprintsVector = '';
```

`FoldFingerprintsByBitDensity`

```
$Fingerprints->FoldFingerprintsByBitDensity($BitDensity);
```

Folds fingerprints by recursively reducing its size by half until bit density is greater than or equal to specified *BitDensity* and returns *Fingerprints*.

`FoldFingerprintsBySize`

```
$Fingerprints->FoldFingerprintsBySize($Size, [$CheckSizeValue]);
```

Fold fingerprints by recursively reducing its size by half until size is less than or equal to specified *Size* and returns *Fingerprints*. By default, value *Size* is checked to make sure it's:

```
>= MinSize and < Size and IsPowerOfTwo
```

`GetFingerprintBitsAsBinaryString`

```
$BinaryASCIIString =
  $Fingerprints->GetFingerprintBitsAsBinaryString();
```

Returns fingerprints as a binary ASCII string containing 0s and 1s.

`GetFingerprintBitsAsHexadecimalString`

```
$HexadecimalString =
  $Fingerprints->GetFingerprintBitsAsHexadecimalString();
```

Returns fingerprints as a hexadecimal string

`GetFingerprintBitsAsRawBinaryString`

```
$RawBinaryString =
```

```
$Fingerprints->GetFingerprintBitsAsRawBinaryString();
```

Returns fingerprints as a raw binary string containing packed bit values for each byte.

#### GetFingerprintsVectorValueIDs

```
$ValueIDsRef = $Fingerprints->GetFingerprintsVectorValueIDs();
@ValueIDs = $Fingerprints->GetFingerprintsVectorValueIDs();
```

Returns fingerprints vector value IDs as an array or reference to an array.

#### GetFingerprintsVectorValues

```
$ValuesRef = $Fingerprints->GetFingerprintsVectorValues();
@Values = $Fingerprints->GetFingerprintsVectorValues();
```

Returns fingerprints vector values as an array or reference to an array.

#### IsFingerprintsGenerationSuccessful

```
$Return = $Fingerprints->IsFingerprintsGenerationSuccessful();
```

Returns 1 or 0 based on whether fingerprints were successfully generated.

#### SetFingerprintsBitVector

```
$Fingerprints->SetFingerprintsBitVector($FingerprintsBitVector);
```

Sets *FingerprintsBitVector* object for *Fingerprints* and returns *Fingerprints*.

#### SetFingerprintsVector

```
$Fingerprints->SetFingerprintsVector();
```

Sets *FingerprintsVector* object for *Fingerprints* and returns *Fingerprints*.

#### SetFingerprintsVectorType

```
$Fingerprints->SetFingerprintsVectorType($VectorType);
```

Sets *FingerprintsVector* type for *Fingerprints* and returns *Fingerprints*. Possible *VectorType* values: *OrderedNumericalValues*, *NumericalValues* or *AlphaNumericalValues*.

#### SetMolecule

```
$Fingerprints->SetMolecule($Molecule);
```

Sets *Molecule* object for *Fingerprints* and returns *Fingerprints*.

#### SetSize

```
$Fingerprints->SetSize($Size);
```

Sets *Size* of fingerprints and returns *Fingerprints*.

#### SetType

```
$Fingerprints->SetType($Type);
```

Sets *Type* of fingerprints and returns *Fingerprints*.

#### SetVectorType

```
$Fingerprints->SetVectorType($Type);
```

Sets *Type* of fingerprints vector and returns *Fingerprints*. Possible *Type* values: *FingerprintsBitVector* or *FingerprintsVector*.

#### AUTHOR

Manish Sud <msud@san.rr.com>

#### SEE ALSO

FingerprintsStringUtil.pm, AtomNeighborhoodsFingerprints.pm, AtomTypesFingerprints.pm, EStateIndicesFingerprints.pm, ExtendedConnectivityFingerprints.pm, MACCSKeys.pm, PathLengthFingerprints.pm, TopologicalAtomPairsFingerprints.pm, TopologicalAtomTripletsFingerprints.pm, TopologicalAtomTorsionsFingerprints.pm, TopologicalPharmacophoreAtomPairsFingerprints.pm, TopologicalPharmacophoreAtomTripletsFingerprints.pm

#### COPYRIGHT

Copyright (C) 2004-2012 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.