

1213692591 1222032501 1224517934 1235687794 1244268533 1528120700 162
 9595024 1856308891 1978806036 2001865095 2096549435 172675415 18344...

METHODS

new

```
$NewExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    %NamesAndValues);
```

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

```
Molecule = ''
Type = 'ExtendedConnectivity'
NeighborhoodRadius = 2
AtomIdentifierType = ''
AtomicInvariantsToUse = ['AS', 'X', 'BO', 'H', 'FC', 'MN']
FunctionalClassesToUse = ['HBD', 'HBA', 'PI', 'NI', 'Ar', 'Hal']
```

Examples:

```
$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Molecule' => $Molecule,
    'AtomIdentifierType' =>
        'AtomicInvariantsAtomTypes');

$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Type' => 'ExtendedConnectivityCount',
    'Molecule' => $Molecule,
    'AtomIdentifierType' =>
        'AtomicInvariantsAtomTypes');

$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Type' => 'ExtendedConnectivityBits',
    'Molecule' => $Molecule,
    'Size' => 1024,
    'AtomIdentifierType' =>
        'AtomicInvariantsAtomTypes');

$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Type' => 'ExtendedConnectivity',
    'Molecule' => $Molecule,
    'NeighborhoodRadius' => 2,
    'AtomIdentifierType' =>
        'AtomicInvariantsAtomTypes',
    'AtomicInvariantsToUse' =>
        ['AS', 'X', 'BO', 'H', 'FC', 'MN'] );

$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Type' => 'ExtendedConnectivity',
    'Molecule' => $Molecule,
    'NeighborhoodRadius' => 2,
    'AtomIdentifierType' =>
        'FunctionalClassAtomTypes',
    'FunctionalClassesToUse' =>
        ['HBD', 'HBA', 'PI', 'NI', 'Ar', 'Hal'] );

$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Type' => 'ExtendedConnectivity',
    'Molecule' => $Molecule,;
    'AtomIdentifierType' =>
        'MMFF94AtomTypes');

$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Type' => 'ExtendedConnectivityCount',
    'Molecule' => $Molecule,;
    'AtomIdentifierType' =>
        'MMFF94AtomTypes');

$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Type' => 'ExtendedConnectivityCount',
    'Molecule' => $Molecule,;
    'AtomIdentifierType' =>
        'SLogPAtomTypes');

$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Type' => 'ExtendedConnectivity',
    'Molecule' => $Molecule,;
```

```

'AtomIdentifierType' =>
    'SLogPAtomTypes' );

$ExtendedConnectivityFingerprints = new ExtendedConnectivityFingerprints(
    'Type' => 'ExtendedConnectivity',
    'Molecule' => $Molecule,
    'AtomIdentifierType' =>
        'SYBYLAtomTypes' );

$ExtendedConnectivityFingerprints->GenerateFingerprints();
print "$ExtendedConnectivityFingerprints\n";

```

GenerateFingerprints

```
$ExtendedConnectivityFingerprints->GenerateFingerprints();
```

Generates extended connectivity fingerprints and returns *ExtendedConnectivityFingerprints*.

GetDescription

```
$Description = $ExtendedConnectivityFingerprints->GetDescription();
```

Returns a string containing description of extended connectivity fingerprints fingerprints.

SetAtomIdentifierType

```
$ExtendedConnectivityFingerprints->SetAtomIdentifierType($IdentifierType);
```

Sets atom *IdentifierType* to use during extended connectivity fingerprints generation and returns *ExtendedConnectivityFingerprints*.

Possible values: *AtomicInvariantsAtomTypes*, *DREIDINGAtomTypes*, *EStateAtomTypes*, *FunctionalClassAtomTypes*, *MMFF94AtomTypes*, *SLogPAtomTypes*, *SYBYLAtomTypes*, *TPSAAAtomTypes*, *UFFAtomTypes*.

SetAtomicInvariantsToUse

```
$ExtendedConnectivityFingerprints->SetAtomicInvariantsToUse($ValuesRef);
$ExtendedConnectivityFingerprints->SetAtomicInvariantsToUse(@Values);
```

Sets atomic invariants to use during *AtomicInvariantsAtomTypes* value of *AtomIdentifierType* for extended connectivity fingerprints generation and returns *ExtendedConnectivityFingerprints*.

Possible values for atomic invariants are: *AS*, *X*, *BO*, *LBO*, *SB*, *DB*, *TB*, *H*, *Ar*, *RA*, *FC*, *MN*, *SM*. Default value [Ref 24]: *AS,X,BO,H,FC,MN*.

The atomic invariants abbreviations correspond to:

AS = Atom symbol corresponding to element symbol

```

X<n>   = Number of non-hydrogen atom neighbors or heavy atoms
BO<n>  = Sum of bond orders to non-hydrogen atom neighbors or heavy atoms
LBO<n> = Largest bond order of non-hydrogen atom neighbors or heavy atoms
SB<n>  = Number of single bonds to non-hydrogen atom neighbors or heavy atoms
DB<n>  = Number of double bonds to non-hydrogen atom neighbors or heavy atoms
TB<n>  = Number of triple bonds to non-hydrogen atom neighbors or heavy atoms
H<n>   = Number of implicit and explicit hydrogens for atom
Ar     = Aromatic annotation indicating whether atom is aromatic
RA     = Ring atom annotation indicating whether atom is a ring
FC<+n/-n> = Formal charge assigned to atom
MN<n>  = Mass number indicating isotope other than most abundant isotope
SM<n>  = Spin multiplicity of atom. Possible values: 1 (singlet), 2 (doublet) or
        3 (triplet)

```

Atom type generated by *AtomTypes::AtomicInvariantsAtomTypes* class corresponds to:

```
AS.X<n>.BO<n>.LBO<n>.<SB><n>.<DB><n>.<TB><n>.H<n>.Ar.RA.FC<+n/-n>.MN<n>.SM<n>
```

Except for *AS* which is a required atomic invariant in atom types, all other atomic invariants are optional. Atom type specification doesn't include atomic invariants with zero or undefined values.

In addition to usage of abbreviations for specifying atomic invariants, the following descriptive words are also allowed:

```

X : NumOfNonHydrogenAtomNeighbors or NumOfHeavyAtomNeighbors
BO : SumOfBondOrdersToNonHydrogenAtoms or SumOfBondOrdersToHeavyAtoms
LBO : LargestBondOrderToNonHydrogenAtoms or LargestBondOrderToHeavyAtoms
SB : NumOfSingleBondsToNonHydrogenAtoms or NumOfSingleBondsToHeavyAtoms
DB : NumOfDoubleBondsToNonHydrogenAtoms or NumOfDoubleBondsToHeavyAtoms
TB : NumOfTripleBondsToNonHydrogenAtoms or NumOfTripleBondsToHeavyAtoms
H : NumOfImplicitAndExplicitHydrogens
Ar : Aromatic
RA : RingAtom
FC : FormalCharge
MN : MassNumber
SM : SpinMultiplicity

```

AtomTypes::AtomicInvariantsAtomTypes module is used to assign atomic invariant atom types.

SetFunctionalClassesToUse

```
$ExtendedConnectivityFingerprints->SetFunctionalClassesToUse($ValuesRef);
$ExtendedConnectivityFingerprints->SetFunctionalClassesToUse(@Values);
```

Sets functional classes invariants to use during *FunctionalClassAtomTypes* value of *AtomIdentifierType* for extended connectivity fingerprints generation and returns *ExtendedConnectivityFingerprints*.

Possible values for atom functional classes are: *Ar, CA, H, HBA, HBD, Hal, NI, PI, RA*. Default value [Ref 24]: *HBD,HBA,PI,NI,Ar,Hal*.

The functional class abbreviations correspond to:

```
HBD: HydrogenBondDonor
HBA: HydrogenBondAcceptor
PI : PositivelyIonizable
NI : NegativelyIonizable
Ar : Aromatic
Hal : Halogen
H : Hydrophobic
RA : RingAtom
CA : ChainAtom
```

Functional class atom type specification for an atom corresponds to:

```
Ar.CA.H.HBA.HBD.Hal.NI.PI.RA or None
```

AtomTypes::FunctionalClassAtomTypes module is used to assign functional class atom types. It uses following definitions [Ref 60-61, Ref 65-66]:

```
HydrogenBondDonor: NH, NH2, OH
HydrogenBondAcceptor: N[!H], O
PositivelyIonizable: +, NH2
NegativelyIonizable: -, C(=O)OH, S(=O)OH, P(=O)OH
```

SetNeighborhoodRadius

```
$ExtendedConnectivityFingerprints->SetNeighborhoodRadius($Radius);
```

Sets neighborhood radius to use during extended connectivity fingerprints generation and returns *ExtendedConnectivityFingerprints*.

StringifyExtendedConnectivityFingerprints

```
$String = $Fingerprints->StringifyExtendedConnectivityFingerprints();
```

Returns a string containing information about *ExtendedConnectivityFingerprints* object.

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

Fingerprints.pm, FingerprintsStringUtil.pm, AtomNeighborhoodsFingerprints.pm, AtomTypesFingerprints.pm, EStateIndicesFingerprints.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.