NAME

AtomicInvariantsAtomTypes

SYNOPSIS

use AtomTypes::AtomicInvariantsAtomTypes;

use AtomTypes::AtomicInvariantsAtomTypes qw(:all);

DESCRIPTION

AtomicInvariantsAtomTypes class provides the following methods:

new, AssignAtomTypes, GetAtomicInvariantsOrder, GetAvailableAtomicInvariants, IsAtomicInvariantAvailable, SetAtomicInvariantsToUse, StringifyAtomicInvariantsAtomTypes

The following functions are available:

GetAvailableAtomicInvariants, IsAtomicInvariantAvailable

AtomicInvariantsAtomTypes is derived from AtomTypes class which in turn is derived from ObjectProperty base class that provides methods not explicitly defined in AtomicInvariantsAtomTypes, AtomTypes 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>();
```

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

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::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
```

```
 Notes:

    . AtomicInvariants with zero or undefined values are not shown.
    . LBO with value of 1 is not shown. And absence of LBO in AtomTypes
      implies the largest bond order value is one.
    . SB, DB and TB with values of zero are not shown.
    . The difference in BO and X values corresponds to numbed of pi electrons [ Ref 57 ].
```

Examples of atomic invariant atom types:

```
    . O.X1.BO1.H1 - Hydroxyl oxygen in carboxylate with attached hydrogen
      and no explicit charge
    . O.X1.BO1.FC-1 - Hydroxyl ozygen in carboxylate with explicit negative
      charge
    . O.X1.BO2 - Carbonyl oxygen in carboxylate with double bond to carbon
    . O.X2.BO2 - Hydroxyl ozygen in carboxylate attached to carbonyl carbon
      and another heavy atom
    . C.X2.BO3.H1.Ar - Aromatic carbon
```

METHODS

new

```
$NewAtomicInvariantsAtomTypes = new AtomTypes::AtomicInvariantsAtomTypes(
                                               %NamesAndValues);
```

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

```
Molecule = ''
Type = 'AtomicInvariants'
IgnoreHydrogens = 0
AtomicInvariantsToUse = AS,X,BO,H,FC
```

Examples:

```
$AtomicInvariantsAtomTypes = new AtomTypes::AtomicInvariantsAtomTypes(
                          'Molecule' => $Molecule,
                          'IgnoreHydrogens' => 0,
                          'AtomicInvariantsToUse' =>
                                     ['AS', 'X', 'BO', 'H', 'FC']);
```

AssignAtomTypes

```
$AtomicInvariantsAtomTypes->AssignAtomTypes();
```

Assigns atomic invariant atom types to all the atoms in a molecule and returns *AtomicInvariantsAtomTypes*.

GetAtomicInvariantsOrder

```
@AtomicInvariantsOrder = $AtomicInvariantsAtomTypes->
                         GetAtomicInvariantsOrder();
```

Returns an array obtaining order of atomic invariants used to generate atom types.

GetAvailableAtomicInvariants

```
%AvailableAtomicInvariants = $AtomicInvariantsAtomTypes->
                             GetAvailableAtomicInvariants();
```

Returns available atomic invariants as a hash containing available atomic invariants and their description as key/value pairs.

IsAtomTypesAssignmentSuccessful

```
$Status = $AtomTypes->IsAtomTypesAssignmentSuccessful();
```

Returns 1 or 0 based on whether atom types assignment was successfully performed. This method overrides the same method available in the base class AtomTypes.pm used to derived this class.

IsAtomicInvariantAvailable

```
    $Status = $AtomicInvariantsAtomTypes->
              IsAtomicInvariantAvailable($AtomicInvariant);
    $Status = AtomTypes::AtomicInvariantsAtomTypes::
              IsAtomicInvariantAvailable($AtomicInvariant);
```

Returns 1 or 0 based on whether *AtomicInvariant* is valid.

SetAtomicInvariantsToUse

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

Sets atomic invariants to use for generating and assigning atom types and returns *AtomicInvariantsAtomTypes*.

StringifyAtomicInvariantsAtomTypes

```
    $String = $AtomicInvariantsAtomTypes->StringifyAtomicInvariantsAtomTypes();
```

Returns a string containing information about *AtomicInvariantsAtomTypes* object.

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

AtomTypes.pm, DREIDINGAtomTypes.pm, EStateAtomTypes.pm, FunctionalClassAtomTypes.pm, MMFF94AtomTypes.pm, SLogPAtomTypes.pm, SYBYLAtomTypes.pm, TPSAAtomTypes.pm, UFFAtomTypes.pm

COPYRIGHT