NAME

TorsionLibraryUtil

SYNOPSIS

import TorsionLibraryUtil

DESCRIPTION

TorsionLibraryUtil module provides the following functions:

CalculateTorsionAngleDifference, DoesSMARTSContainValidSubClassMappedAtoms, DoesSMARTSContainValidTorsionRuleMappedAtoms, DoesSMARTSContainsMappedAtoms, GetGenericHierarchyClassElementNode, IdentifyRotatableBondsForTorsionLibraryMatch, IsSpecificHierarchyClass, ListTorsionLibraryInfo, RemoveLastHierarchyClassElementNodeFromTracking, RemoveLastHierarchySubClassElementNodeFromTracking, RetrieveTorsionLibraryInfo, SetupHierarchyClassAndSubClassNamesForRotatableBond, SetupHierarchySubClassElementPatternMol, SetupTorsionLibraryInfoForMatchingRotatableBonds, SetupTorsionRuleAnglesInfo, SetupTorsionRuleElementPatternMol, TrackHierarchyClassElementNode, TrackHierarchySubClassElementNode

FUNCTIONS

CalculateTorsionAngleDifference

CalculateTorsionAngleDifference(TorsionAngle1, TorsionAngle2)

Calculate torsion angle difference in the range from 0 to 180.

Arguments:

TorsionAngle1 (float): First torsion angle.
TorsionAngle2 (float): Second torsion angle.

Returns:

float: Difference between first and second torsion angle.

DoesSMARTSContainValidSubClassMappedAtoms

DoesSMARTSContainValidSubClassMappedAtoms(SMARTS)

Check for the presence of two central mapped atoms in SMARTS pattern. A valid SMARTS pattern must contain only two mapped atoms corresponding to map atom numbers ':2' and ':3'.

Arguments:

SMARTS (str): SMARTS pattern for sub class in torsion library XML tree.

Returns:

bool: True – A valid pattern; Otherwise, false.

DoesSMARTSContainValidTorsionRuleMappedAtoms

DoesSMARTSContainValidTorsionRuleMappedAtoms(SMARTS)

Check for the presence of four mapped atoms in a SMARTS pattern. A valid SMARTS pattern must contain only four mapped atoms corresponding to map atom numbers ':1', ':2', ':3' and ':4'.

Arguments:

SMARTS (str): SMARTS pattern for torsion rule in torsion library XML
    tree.

Returns:

bool: True – A valid pattern; Otherwise, false.

DoesSMARTSContainsMappedAtoms

DoesSMARTSContainsMappedAtoms(SMARTS, MappedAtomNumsList)

Check for the presence of specified mapped atoms in SMARTS pattern. The mapped atom numbers in the list are specified as ':1', ':2', ':3' etc.

Arguments:

```
SMARTS (str): SMARTS pattern in torsion library XML tree.
MappedAtoms (list): Mapped atom numbers as ":1", ":2" etc.
```

Returns:

```
bool: True - All mapped atoms present in pattern; Otherwise, false.
```

GetGenericHierarchyClassElementNode

```
GetGenericHierarchyClassElementNode(TorsionLibraryInfo)
```

Get generic hierarchy class element node.

Arguments:

```
TorsionLibraryInfo (dict): A dictionary containing information for
    matching rotatable bonds.
```

Returns:

```
object: Generic hierarchy class element node in torsion library XML
    tree.
```

IdentifyRotatableBondsForTorsionLibraryMatch

```
IdentifyRotatableBondsForTorsionLibraryMatch(TorsionLibraryInfo, Mol,
RotBondsPatternMol)
```

Identify rotatable bonds in a molecule for torsion library match.

Arguments:

```
TorsionLibraryInfo (dict): A dictionary containing information for
    matching rotatable bonds.
Mol (object): RDKit molecule object.
RotBondsPatternMol (object): RDKit molecule object for SMARTS pattern
    corresponding to rotatable bonds.
```

Returns:

```
bool: True - Rotatable bonds present in molecule; Otherwise, false.
None or dict: None - For no rotatable bonds in molecule; otherwise, a
    dictionary containing the following informations for rotatable bonds
    matched to RotBondsPatternMol:

        RotBondsInfo["IDs"] = []
        RotBondsInfo["AtomIndices"] = {}
        RotBondsInfo["HierarchyClass"] = {}
```

IsSpecificHierarchyClass

```
IsSpecificHierarchyClass(TorsionLibraryInfo, HierarchyClass)
```

Check whether it's a specific hierarchy class.

Arguments:

```
TorsionLibraryInfo (dict): A dictionary containing information for
    matching rotatable bonds.
HierarchyClass (str): Hierarchy class name.
```

Returns:

```
bool: True - A valid hierarchy class name; Otherwise, false.
```

ListTorsionLibraryInfo

```
ListTorsionLibraryInfo(TorsionLibElementTree)
```

List torsion library information using XML tree object. The following information is listed:

Summary:

```
Total number of HierarchyClass nodes: <Number>
Total number of HierarchyClassSubClass nodes: <Number
Total number of TorsionRule nodes: <Number
```

Details:

```
HierarchyClass: <Name>; HierarchySubClass nodes: <Number>;
    TorsionRule nodes: <SMARTS>
 ... ... ...
```

Arguments:

```
TorsionLibElementTree (object): XML tree object.
```

Returns:

```
Nothing.
```

RemoveLastHierarchyClassElementNodeFromTracking

```
RemoveLastHierarchyClassElementNodeFromTracking(TorsionLibraryInfo)
```

Remove last hierarchy class element node from tracking by removing it from a stack.

Arguments:

```
TorsionLibraryInfo (dict): A dictionary containing information for
    matching rotatable bonds.
```

Returns:

```
Nothing. The torsion library info is updated.
```

RemoveLastHierarchySubClassElementNodeFromTracking

```
RemoveLastHierarchySubClassElementNodeFromTracking(TorsionLibraryInfo)
```

Remove last hierarchy sub class element node from tracking by removing it from a stack.

Arguments:

```
TorsionLibraryInfo (dict): A dictionary containing information for
    matching rotatable bonds.
```

Returns:

```
Nothing. The torsion library info is updated.
```

RetrieveTorsionLibraryInfo

```
RetrieveTorsionLibraryInfo(TorsionLibraryFilePath, Quiet = True)
```

Retrieve torsion library information.

Arguments:

```
TorsionLibraryFilePath (str):  Torsion library XML file path.
```

Returns:

```
object: An object returned by xml.etree.ElementTree.parse function.
```

The XML file is parsed using xml.etree.ElementTree.parse function and object created by the parse function is simply returned.

SetupHierarchyClassAndSubClassNamesForRotatableBond

```
SetupHierarchyClassAndSubClassNamesForRotatableBond(TorsionLibraryInfo)
```

Setup hierarchy class and subclass names for a rotatable bond matched to a torsion rule element node.

Returns:

> TorsionLibraryInfo (dict): A dictionary containing information for matching rotatable bonds.
>
> str: A back slash delimited string containing hierarchy class names at the level of torsion rule element node.
> str: A back slash delimited string containing hierarchy sub class names at the level of torsion rule element node.

SetupHierarchySubClassElementPatternMol

```
SetupHierarchySubClassElementPatternMol(TorsionLibraryInfo, ElementNode)
```

Setup pattern molecule for SMARTS pattern in hierarchy subclass element.

Arguments:

> TorsionLibraryInfo (dict): A dictionary containing information for matching rotatable bonds.
> ElementNode (object): A hierarchy sub class element node being matched in torsion library XML tree.

Returns:

> object: RDKit molecule object corresponding to SMARTS pattern for hierarchy sub class element node.

SetupTorsionLibraryInfoForMatchingRotatableBonds

```
SetupTorsionLibraryInfoForMatchingRotatableBonds(TorsionLibraryInfo)
```

Setup torsion library information for matching rotatable bonds. The following information is initialized and updated in torsion library dictionary for matching rotatable bonds:

```
TorsionLibraryInfo["GenericClass"] = None
TorsionLibraryInfo["GenericClassElementNode"] = None

TorsionLibraryInfo["SpecificClasses"] = {}
TorsionLibraryInfo["SpecificClasses"]["Names"] = []
TorsionLibraryInfo["SpecificClasses"]["ElementNode"] = {}

TorsionLibraryInfo["HierarchyClassNodes"] = []
TorsionLibraryInfo["HierarchySubClassNodes"] = []

TorsionLibraryInfo["DataCache"] = {}
TorsionLibraryInfo["DataCache"]["SubClassPatternMol"] = {}

TorsionLibraryInfo["DataCache"]["TorsionRulePatternMol"] = {}
TorsionLibraryInfo["DataCache"]["TorsionRuleAnglesInfo"] = {}
```

Arguments:

> TorsionLibraryInfo (dict): A dictionary containing root node for torsion library element tree.

Returns:

> Nonthing. The torsion library information dictionary is updated.

SetupTorsionRuleAnglesInfo

```
SetupTorsionRuleAnglesInfo(TorsionLibraryInfo, TorsionRuleElementNode)
```

Setup torsion angles and energy info for matching a torsion rule.

Arguments:

> TorsionLibraryInfo (dict): A dictionary containing information for
>   matching rotatable bonds.
> TorsionRuleElementNode (object): A torsion rule element node being
>   matched in torsion library XML tree.

Returns:

> dict: A dictionary containing the following information for torsion rule
>   being matched to a rotatable bond:
>
>   RuleAnglesInfo = {}
>
>   RuleAnglesInfo["IDs"] = []
>   RuleAnglesInfo["Value"] = {}
>   RuleAnglesInfo["Score"] = {}
>   RuleAnglesInfo["Tolerance1"] = {}
>   RuleAnglesInfo["Tolerance2"] = {}
>
>   RuleAnglesInfo["ValuesList"] = []
>   RuleAnglesInfo["ValuesIn360RangeList"] = []
>   RuleAnglesInfo["Tolerances1List"] = []
>   RuleAnglesInfo["Tolerances2List"] = []
>
>   # Strain energy calculations...
>   RuleAnglesInfo["EnergyMethod"] = None
>   RuleAnglesInfo["EnergyMethodExact"] = None
>   RuleAnglesInfo["EnergyMethodApproximate"] = None
>
>   # For approximate strain energy calculation...
>   RuleAnglesInfo["Beta1"] = {}
>   RuleAnglesInfo["Beta2"] = {}
>   RuleAnglesInfo["Theta0"] = {}
>
>   # For exact strain energy calculation...
>   RuleAnglesInfo["HistogramEnergy"] = []
>   RuleAnglesInfo["HistogramEnergyLowerBound"] = []
>   RuleAnglesInfo["HistogramEnergyUpperBound"] = []

SetupTorsionRuleElementPatternMol

> SetupTorsionRuleElementPatternMol(TorsionLibraryInfo, ElementNode, TorsionRuleNodeID,
> TorsionSMARTSPattern)

Setup pattern molecule for SMARTS pattern in torsion rule element.

Arguments:

> TorsionLibraryInfo (dict): A dictionary containing information for
>   matching rotatable bonds.
> ElementNode (object): A torsion rule element node being matched in
>   torsion library XML tree.
> TorsionRuleNodeID (int): Torsion rule element node ID.
> TorsionSMARTSPattern (str): SMARTS pattern for torsion rule element node.

Returns:

> object: RDKit molecule object corresponding to SMARTS pattern for
>   torsion rule element node.

TrackHierarchyClassElementNode

> TrackHierarchyClassElementNode(TorsionLibraryInfo, ElementNode)

Track hierarchy class element node using a stack.

Arguments:

```
TorsionLibraryInfo (dict): A dictionary containing information for
    matching rotatable bonds.
ElementNode (object): Hierarchy class element node in torsion library
    XML tree.
```

Returns:

```
Nothing. The torsion library info is updated.
```

TrackHierarchySubClassElementNode

```
TrackHierarchySubClassElementNode(TorsionLibraryInfo, ElementNode)
```

Track hierarchy sub class element node using a stack.

Arguments:

```
TorsionLibraryInfo (dict): A dictionary containing information for
    matching rotatable bonds.
ElementNode (object): Hierarchy sub class element node in torsion
    library XML tree.
```

Returns:

```
Nothing. The torsion library info is updated.
```

AUTHOR

Manish Sud <msud@san.rr.com>

COPYRIGHT

Copyright (C) 2022 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.