NAME

FingerprintsVector

SYNOPSIS

```
use Fingerprints::FingerprintsVector;

use Fingerprints::FingerprintsVector qw(:all);
```

DESCRIPTION

FingerprintsVector class provides the following methods:

new, AddValueIDs, AddValues, CityBlockDistanceCoefficient, CosineSimilarityCoefficient, CzekanowskiSimilarityCoefficient, DiceSimilarityCoefficient, EuclideanDistanceCoefficient, GetDescription, GetFingerprintsVectorString, GetID, GetIDsAndValuesPairsString, GetIDsAndValuesString, GetNumOfNonZeroValues, GetNumOfValueIDs, GetNumOfValues, GetSupportedDistanceAndSimilarityCoefficients, GetSupportedDistanceCoefficients, GetSupportedSimilarityCoefficients, GetType, GetValue, GetValueID, GetValueIDs, GetValueIDsString, GetValues, GetValuesAndIDsPairsString, GetValuesAndIDsString, GetValuesString, GetVectorType, HammingDistanceCoefficient, IsFingerprintsVector, JaccardSimilarityCoefficient, ManhattanDistanceCoefficient, NewFromIDsAndValuesPairsString, NewFromIDsAndValuesString, NewFromValuesAndIDsPairsString, NewFromValuesAndIDsString, NewFromValuesString, OchiaiSimilarityCoefficient, SetDescription, SetID, SetType, SetValue, SetValueID, SetValueIDs, SetValues, SetVectorType, SoergelDistanceCoefficient, SorensonSimilarityCoefficient, StringifyFingerprintsVector, TanimotoSimilarityCoefficient

The methods available to create fingerprints vector from strings and to calculate similarity and distance coefficients between two vectors can also be invoked as class functions.

FingerprintsVector class provides support to perform comparison between vectors containing three different types of values:

Type I: OrderedNumericalValues

- Size of two vectors are same
- Vectors contain real values in a specific order. For example: MACCS keys count, Topological pharmacophore atom pairs and so on.

Type II: UnorderedNumericalValues

- Size of two vectors might not be same
- Vectors contain unordered real value identified by value IDs. For example: Topological atom pairs, Topological atom torsions and so on

Type III: AlphaNumericalValues

- Size of two vectors might not be same
- Vectors contain unordered alphanumerical values. For example: Extended connectivity fingerprints, atom neighborhood fingerprints.

Before performing similarity or distance calculations between vectors containing UnorderedNumericalValues or AlphaNumericalValues, the vectors are transformed into vectors containing unique OrderedNumericalValues using value IDs for UnorderedNumericalValues and values itself for AlphaNumericalValues.

Three forms of similarity and distance calculation between two vectors, specified using CalculationMode option, are supported: *AlgebraicForm, BinaryForm or SetTheoreticForm*.

For *BinaryForm*, the ordered list of processed final vector values containing the value or count of each unique value type is simply converted into a binary vector containing 1s and 0s corresponding to presence or absence of values before calculating similarity or distance between two vectors.

For two fingerprint vectors A and B of same size containing OrderedNumericalValues, let:

```
N = Number values in A or B
```

```
    Xa = Values of vector A
    Xb = Values of vector B

    Xai = Value of ith element in A
    Xbi = Value of ith element in B

   SUM = Sum of i over N values
```

For SetTheoreticForm of calculation between two vectors, let:

```
    SetIntersectionXaXb = SUM ( MIN ( Xai, Xbi ) )
    SetDifferenceXaXb = SUM ( Xai ) + SUM ( Xbi ) - SUM ( MIN ( Xai, Xbi ) )
```

For BinaryForm of calculation between two vectors, let:

```
    Na = Number of bits set to "1" in A = SUM ( Xai )
    Nb = Number of bits set to "1" in B = SUM ( Xbi )
    Nc = Number of bits set to "1" in both A and B = SUM ( Xai * Xbi )
    Nd = Number of bits set to "0" in both A and B
       = SUM ( 1 - Xai - Xbi + Xai * Xbi)

    N = Number of bits set to "1" or "0" in A or B = Size of A or B = Na + Nb - Nc + Nd
```

Additionally, for BinaryForm various values also correspond to:

```
    Na = | Xa |
    Nb = | Xb |
    Nc = | SetIntersectionXaXb |
    Nd = N - | SetDifferenceXaXb |

    | SetDifferenceXaXb | = N - Nd = Na + Nb - Nc + Nd - Nd = Na + Nb - Nc
                          =  | Xa | + | Xb | - | SetIntersectionXaXb |
```

Various similarity and distance coefficients [Ref 40, Ref 62, Ref 64] for a pair of vectors A and B in *AlgebraicForm, BinaryForm and SetTheoreticForm* are defined as follows:

CityBlockDistance: (same as HammingDistance and ManhattanDistance)

AlgebraicForm: SUM (ABS (Xai - Xbi))

BinaryForm: (Na - Nc) + (Nb - Nc) = Na + Nb - 2 * Nc

SetTheoreticForm: | SetDifferenceXaXb | - | SetIntersectionXaXb | = SUM (Xai) + SUM (Xbi) - 2 * (SUM (MIN (Xai, Xbi)))

CosineSimilarity: (same as OchiaiSimilarityCoefficient)

AlgebraicForm: SUM (Xai * Xbi) / SQRT (SUM (Xai ** 2) * SUM (Xbi ** 2))

BinaryForm: Nc / SQRT (Na * Nb)

SetTheoreticForm: | SetIntersectionXaXb | / SQRT (|Xa| * |Xb|) = SUM (MIN (Xai, Xbi)) / SQRT (SUM (Xai) * SUM (Xbi))

CzekanowskiSimilarity: (same as DiceSimilarity and SorensonSimilarity)

AlgebraicForm: (2 * (SUM (Xai * Xbi))) / (SUM (Xai ** 2) + SUM (Xbi **2))

BinaryForm: 2 * Nc / (Na + Nb)

SetTheoreticForm: 2 * | SetIntersectionXaXb | / (|Xa| + |Xb|) = 2 * (SUM (MIN (Xai, Xbi))) / (SUM (Xai) + SUM (Xbi))

DiceSimilarity: (same as CzekanowskiSimilarity and SorensonSimilarity)

AlgebraicForm: (2 * (SUM (Xai * Xbi))) / (SUM (Xai ** 2) + SUM (Xbi **2))

BinaryForm: 2 * Nc / (Na + Nb)

SetTheoreticForm: 2 * | SetIntersectionXaXb | / (|Xa| + |Xb|) = 2 * (SUM (MIN (Xai, Xbi))) / (SUM (Xai) + SUM (Xbi))

EuclideanDistance:

AlgebraicForm: SQRT (SUM (((Xai - Xbi) ** 2)))

BinaryForm: SQRT ((Na - Nc) + (Nb - Nc)) = SQRT (Na + Nb - 2 * Nc)

SetTheoreticForm: SQRT (| SetDifferenceXaXb | - | SetIntersectionXaXb |) = SQRT (SUM (Xai) + SUM (Xbi) - 2 * (SUM (MIN (Xai, Xbi))))

HammingDistance: (same as CityBlockDistance and ManhattanDistance)

AlgebraicForm: SUM (ABS (Xai - Xbi))

BinaryForm: (Na - Nc) + (Nb - Nc) = Na + Nb - 2 * Nc

SetTheoreticForm: | SetDifferenceXaXb | - | SetIntersectionXaXb | = SUM (Xai) + SUM (Xbi) - 2 * (SUM (MIN (Xai, Xbi)))

JaccardSimilarity: (same as TanimotoSimilarity)

AlgebraicForm: SUM (Xai * Xbi) / (SUM (Xai ** 2) + SUM (Xbi ** 2) - SUM (Xai * Xbi))

BinaryForm: Nc / ((Na - Nc) + (Nb - Nc) + Nc) = Nc / (Na + Nb - Nc)

SetTheoreticForm: | SetIntersectionXaXb | / | SetDifferenceXaXb | = SUM (MIN (Xai, Xbi)) / (SUM (Xai) + SUM (Xbi) - SUM (MIN (Xai, Xbi)))

ManhattanDistance: (same as CityBlockDistance and HammingDistance)

AlgebraicForm: SUM (ABS (Xai - Xbi))

BinaryForm: (Na - Nc) + (Nb - Nc) = Na + Nb - 2 * Nc

SetTheoreticForm: | SetDifferenceXaXb | - | SetIntersectionXaXb | = SUM (Xai) + SUM (Xbi) - 2 * (SUM (MIN (Xai, Xbi)))

OchiaiSimilarity: (same as CosineSimilarity)

AlgebraicForm: SUM (Xai * Xbi) / SQRT (SUM (Xai ** 2) * SUM (Xbi ** 2))

BinaryForm: Nc / SQRT (Na * Nb)

SetTheoreticForm: | SetIntersectionXaXb | / SQRT (|Xa| * |Xb|) = SUM (MIN (Xai, Xbi)) / SQRT (SUM (Xai) * SUM (Xbi))

SorensonSimilarity: (same as CzekanowskiSimilarity and DiceSimilarity)

AlgebraicForm: (2 * (SUM (Xai * Xbi))) / (SUM (Xai ** 2) + SUM (Xbi **2))

BinaryForm: 2 * Nc / (Na + Nb)

SetTheoreticForm: 2 * | SetIntersectionXaXb | / (|Xa| + |Xb|) = 2 * (SUM (MIN (Xai, Xbi))) / (SUM (Xai) + SUM (Xbi))

SoergelDistance:

AlgebraicForm: SUM (ABS (Xai - Xbi)) / SUM (MAX (Xai, Xbi))

BinaryForm: 1 - Nc / (Na + Nb - Nc) = (Na + Nb - 2 * Nc) / (Na + Nb - Nc)

SetTheoreticForm: (| SetDifferenceXaXb | - | SetIntersectionXaXb |) / | SetDifferenceXaXb | = (SUM (Xai) + SUM (Xbi) - 2 * (SUM (MIN (Xai, Xbi)))) / (SUM (Xai) + SUM (Xbi) - SUM (MIN (Xai, Xbi)))

TanimotoSimilarity: (same as JaccardSimilarity)

AlgebraicForm: SUM (Xai * Xbi) / (SUM (Xai ** 2) + SUM (Xbi ** 2) - SUM (Xai * Xbi))

BinaryForm: Nc / ((Na - Nc) + (Nb - Nc) + Nc) = Nc / (Na + Nb - Nc)

SetTheoreticForm: | SetIntersectionXaXb | / | SetDifferenceXaXb | = SUM (MIN (Xai, Xbi)) / (SUM (Xai) + SUM (Xbi) - SUM (MIN (Xai, Xbi)))

METHODS

new

```
$FPVector = new Fingerprints::FingerprintsVector(%NamesAndValues);
```

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

```
Type = ''
@{Values} = ()
@{ValuesIDs} = ()
```

Examples:

```
$FPVector = new Fingerprints::FingerprintsVector('Type' => 'OrderedNumericalValues',
                                    'Values' => [1, 2, 3, 4]);
$FPVector = new Fingerprints::FingerprintsVector('Type' => 'NumericalValues',
                                    'Values' => [10, 22, 33, 44],
                                    'ValueIDs' => ['ID1', 'ID2', 'ID3', 'ID4']);
$FPVector = new Fingerprints::FingerprintsVector('Type' => 'AlphaNumericalValues',
                                    'Values' => ['a1', 2, 'a3', 4]);
```

AddValueIDs

```
$FingerprintsVector->AddValueIDs($ValueIDsRef);
$FingerprintsVector->AddValueIDs(@ValueIDs);
```

Adds specified *ValueIDs* to *FingerprintsVector* and returns *FingerprintsVector*.

AddValues

```
$FingerprintsVector->AddValues($ValuesRef);
$FingerprintsVector->AddValues(@Values);
$FingerprintsVector->AddValues($Vector);
```

Adds specified *Values* to *FingerprintsVector* and returns *FingerprintsVector*.

CityBlockDistanceCoefficient

```
$Value = $FingerprintsVector->CityBlockDistanceCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::CityBlockDistanceCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *CityBlock* distance coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm*, *BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

CosineSimilarityCoefficient

```
$Value = $FingerprintsVector->CosineSimilarityCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::CosineSimilarityCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Cosine* similarity coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm*, *BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

CzekanowskiSimilarityCoefficient

```
$Value = $FingerprintsVector->CzekanowskiSimilarityCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::CzekanowskiSimilarityCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Czekanowski* similarity coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm*, *BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

DiceSimilarityCoefficient

```
$Value = $FingerprintsVector->DiceSimilarityCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::DiceSimilarityCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Dice* similarity coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm*, *BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

EuclideanDistanceCoefficient

```
$Value = $FingerprintsVector->EuclideanDistanceCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::EuclideanDistanceCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Euclidean* distance coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm*, *BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

GetDescription

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

Returns a string containing description of fingerprints vector.

GetFingerprintsVectorString

```
$FPString = $FingerprintsVector->GetFingerprintsVectorString($Format);
```

Returns a FingerprintsString containing vector values and/or IDs in *FingerprintsVector* corresponding to specified *Format*.

Possible *Format* values: *IDsAndValuesString, IDsAndValues*, *IDsAndValuesPairsString, IDsAndValuesPairs*, *ValuesAndIDsString, ValuesAndIDs*, *ValuesAndIDsPairsString, ValuesAndIDsPairs, ValueIDsString, ValueIDs*, *ValuesString, or Values*.

GetID

```
$ID = $FingerprintsVector->GetID();
```

Returns *ID* of *FingerprintsVector*.

GetVectorType

```
$VectorType = $FingerprintsVector->GetVectorType();
```

Returns *VectorType* of *FingerprintsVector*.

GetIDsAndValuesPairsString

```
$IDsValuesPairsString = $FingerprintsVector->GetIDsAndValuesPairsString();
```

Returns *FingerprintsVector* value IDs and values as space delimited ID/value pair string.

GetIDsAndValuesString

```
$IDsValuesString = $FingerprintsVector->GetIDsAndValuesString();
```

Returns *FingerprintsVector* value IDs and values as string containing space delimited IDs followed by values with semicolon as IDs and values delimiter.

GetNumOfNonZeroValues

```
$NumOfNonZeroValues = $FingerprintsVector->GetNumOfNonZeroValues();
```

Returns number of non-zero values in *FingerprintsVector*.

GetNumOfValueIDs

```
$NumOfValueIDs = $FingerprintsVector->GetNumOfValueIDs();
```

Returns number of value IDs *FingerprintsVector*.

GetNumOfValues

```
$NumOfValues = $FingerprintsVector->GetNumOfValues();
```

Returns number of values *FingerprintsVector*.

GetSupportedDistanceAndSimilarityCoefficients

```
@SupportedDistanceAndSimilarityCoefficientsReturn =
    Fingerprints::FingerprintsVector::GetSupportedDistanceAndSimilarityCoefficients();
```

Returns an array containing names of supported distance and similarity coefficients.

GetSupportedDistanceCoefficients

```
@SupportedDistanceCoefficientsReturn =
    Fingerprints::FingerprintsVector::GetSupportedDistanceCoefficients();
```

Returns an array containing names of supported disyance coefficients.

GetSupportedSimilarityCoefficients

```
@SupportedSimilarityCoefficientsReturn =
    Fingerprints::FingerprintsVector::GetSupportedSimilarityCoefficients();
```

Returns an array containing names of supported similarity coefficients.

GetType

```
$VectorType = $FingerprintsVector->GetType();
```

Returns *FingerprintsVector* vector type.

GetValue

```
$Value = $FingerprintsVector->GetValue($Index);
```

Returns fingerprints vector Value specified using *Index* starting at 0.

GetValueID

```
$ValueID = $FingerprintsVector->GetValueID();
```

Returns fingerprints vector ValueID specified using *Index* starting at 0.

GetValueIDs

```
$ValueIDs = $FingerprintsVector->GetValueIDs();
@ValueIDs = $FingerprintsVector->GetValueIDs();
```

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

GetValueIDsString

```
$ValueIDsString = $FingerprintsVector->GetValueIDsString();
```

Returns fingerprints vector ValueIDsString with value IDs delimited by space.

GetValues

```
$ValuesRef = $FingerprintsVector->GetValues();
@Values = $FingerprintsVector->GetValues();
```

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

GetValuesAndIDsPairsString

```
$ValuesIDsPairsString = $FingerprintsVector->GetValuesAndIDsPairsString();
```

Returns *FingerprintsVector* value and value IDs as space delimited ID/value pair string.

GetValuesAndIDsString

```
$ValuesIDsString = $FingerprintsVector->GetValuesAndIDsString();
```

Returns *FingerprintsVector* values and value IDs as string containing space delimited IDs followed by values with semicolon as IDs and values delimiter.

GetValuesString

```
$Return = $FingerprintsVector->GetValuesString();
```

Returns *FingerprintsVector* values as space delimited string.

HammingDistanceCoefficient

```
$Value = $FingerprintsVector->HammingDistanceCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::HammingDistanceCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Hamming* distance coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm*, *BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

IsFingerprintsVector

```
$Status = Fingerprints::FingerprintsVector::IsFingerprintsVector($Object);
```

Returns 1 or 0 based on whether *Object* is a *FingerprintsVector*.

JaccardSimilarityCoefficient

```
$Value = $FingerprintsVector->JaccardSimilarityCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::JaccardSimilarityCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Jaccard* similarity coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm, BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

ManhattanDistanceCoefficient

```
$Value = $FingerprintsVector->ManhattanDistanceCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::ManhattanDistanceCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Manhattan* distance coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm, BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

NewFromIDsAndValuesPairsString

```
$FingerprintsVector = $FingerprintsVector->NewFromIDsAndValuesPairsString(
                      $ValuesType, $IDsAndValuesPairsString);
$FingerprintsVector = Fingerprints::FingerprintsVector::NewFromIDsAndValuesPairsString(
                      $ValuesType, $IDsAndValuesPairsString);
```

Creates a new *FingerprintsVector* of *ValuesType* using *IDsAndValuesPairsString* containing space delimited value IDs and values pairs and returns new FingerprintsVector object. Possible *ValuesType* values: *OrderedNumericalValues, NumericalValues, or AlphaNumericalValues*.

NewFromIDsAndValuesString

```
$FingerprintsVector = $FingerprintsVector->NewFromIDsAndValuesString(
                      $ValuesType, $IDsAndValuesString);
$FingerprintsVector = Fingerprints::FingerprintsVector::NewFromIDsAndValuesString(
                      $ValuesType, $IDsAndValuesString);
```

Creates a new *FingerprintsVector* of *ValuesType* using *IDsAndValuesString* containing semicolon delimited value IDs string followed by values strings and returns new FingerprintsVector object. The values within value and value IDs tring are delimited by spaces. Possible *ValuesType* values: *OrderedNumericalValues, NumericalValues, or AlphaNumericalValues*.

NewFromValuesAndIDsPairsString

```
$FingerprintsVector = $FingerprintsVector->NewFromValuesAndIDsPairsString(
                      $ValuesType, $ValuesAndIDsPairsString);
$FingerprintsVector = Fingerprints::FingerprintsVector::NewFromValuesAndIDsPairsString(
                      $ValuesType, $ValuesAndIDsPairsString);
```

Creates a new *FingerprintsVector* of *ValuesType* using *ValuesAndIDsPairsString* containing space delimited value and value IDs pairs and returns new FingerprintsVector object. Possible *ValuesType* values: *OrderedNumericalValues, NumericalValues, or AlphaNumericalValues*.

NewFromValuesAndIDsString

```
$FingerprintsVector = $FingerprintsVector->NewFromValuesAndIDsString(
                      $ValuesType, $IDsAndValuesString);
$FingerprintsVector = Fingerprints::FingerprintsVector::NewFromValuesAndIDsString(
                      $ValuesType, $IDsAndValuesString);
```

Creates a new *FingerprintsVector* of *ValuesType* using *ValuesAndIDsString* containing semicolon delimited values string followed by value IDs strings and returns new FingerprintsVector object. The values within values and value IDs tring are delimited by spaces. Possible *ValuesType* values: *OrderedNumericalValues, NumericalValues, or AlphaNumericalValues*.

NewFromValuesString

```
$FingerprintsVector = $FingerprintsVector->NewFromValuesString(
                      $ValuesType, $ValuesString);
$FingerprintsVector = Fingerprints::FingerprintsVector::NewFromValuesString(
                      $ValuesType, $ValuesString);
```

Creates a new *FingerprintsVector* of *ValuesType* using *ValuesString* containing space delimited values string and returns new FingerprintsVector object. The values within values and value IDs tring are delimited by spaces. Possible *ValuesType* values: *OrderedNumericalValues, NumericalValues, or AlphaNumericalValues*.

OchiaiSimilarityCoefficient

```
$Value = $FingerprintsVector->OchiaiSimilarityCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::OchiaiSimilarityCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Ochiai* similarity coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm, BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

SetDescription

```
$FingerprintsVector->SetDescription($Description);
```

Sets *Description* of fingerprints vector and returns *FingerprintsVector*.

SetID

```
$FingerprintsVector->SetID($ID);
```

Sets *ID* of fingerprints vector and returns *FingerprintsVector*.

SetVectorType

```
$FingerprintsVector->SetVectorType($VectorType);
```

Sets *VectorType* of fingerprints vector and returns *FingerprintsVector*.

SetType

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

Sets *FingerprintsVector* values *Type* and returns *FingerprintsVector*. Possible *Type* values: *OrderedNumericalValues, NumericalValues, or AlphaNumericalValues*.

During calculation of similarity and distance coefficients between two *FingerprintsVectors*, the following conditions apply to vector type, size, value and value IDs:

- For OrderedNumericalValues type, both vectors must be of the same size and contain similar types of numerical values in the same order.
- For NumericalValues type, vector value IDs for both vectors must be specified; however, their size and order of IDs and numerical values may be different. For each vector, value IDs must correspond to vector values.
- For AlphaNumericalValues type, vectors may contain both numerical and alphanumerical values and their sizes may be different.

SetValue

```
$FingerprintsVector->SetValue($Index, $Value, [$SkipIndexCheck]);
```

Sets a *FingerprintsVector* value specified by *Index* starting at 0 to *Value* along with optional index range check and returns *FingerprintsVector*.

SetValueID

```
$FingerprintsVector->SetValueID($Index, $ValueID, [$SkipIndexCheck]);
```

Sets a *FingerprintsVector* value ID specified by *Index* starting at 0 to *ValueID* along with optional index range check and returns *FingerprintsVector*.

SetValueIDs

```
$FingerprintsVector->SetValueIDs($ValueIDsRef);
$FingerprintsVector->SetValueIDs(@ValueIDs);
```

Sets *FingerprintsVector* value IDs to specified *ValueIDs* and returns *FingerprintsVector*.

SetValues

```
$FingerprintsVector->SetValues($ValuesRef);
$FingerprintsVector->SetValues(@Values);
```

Sets *FingerprintsVector* value to specified *Values* and returns *FingerprintsVector*.

SoergelDistanceCoefficient

```
$Value = $FingerprintsVector->SoergelDistanceCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::SoergelDistanceCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Soergel* distance coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm*, *BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

SorensonSimilarityCoefficient

```
$Value = $FingerprintsVector->SorensonSimilarityCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::SorensonSimilarityCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Sorenson* similarity coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm*, *BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

TanimotoSimilarityCoefficient

```
$Value = $FingerprintsVector->TanimotoSimilarityCoefficient(
          $OtherFingerprintVector, [$CalculationMode, $SkipValuesCheck]);
$Value = Fingerprints::FingerprintsVector::TanimotoSimilarityCoefficient(
          $FingerprintsVectorA, $FingerprintVectorB,
          [$CalculationMode, $SkipValuesCheck]);
```

Returns value of *Tanimoto* similarity coefficient between two *FingerprintsVectors* using optionally specified *CalculationMode* and optional checking of vector values.

Possible *CalculationMode* values: *AlgebraicForm*, *BinaryForm or SetTheoreticForm*. Default *CalculationMode* value: *AlgebraicForm*. Default *SkipValuesCheck* value: *0*.

StringifyFingerprintsVector

```
$String = $FingerprintsVector->StringifyFingerprintsVector();
```

Returns a string containing information about *FingerprintsVector* object.

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

BitVector.pm, FingerprintsStringUtil.pm, FingerprintsBitVector.pm, Vector.pm

COPYRIGHT

Copyright (C) 2026 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.