

Nonnegative Matrix factorization

1

Generated by Doxygen 1.5.7.1

Mon Dec 8 00:05:43 2008

Contents

1	Class Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	NMF.FileHandler Class Reference	5
3.1.1	Detailed Description	6
3.1.2	Constructor & Destructor Documentation	6
3.1.2.1	FileHandler	6
3.1.2.2	FileHandler	6
3.1.3	Member Function Documentation	6
3.1.3.1	close	6
3.1.3.2	DirectoryExists	7
3.1.3.3	FileExists	7
3.1.3.4	GetLine	7
3.1.3.5	open	7
3.1.3.6	PutLine	7
3.1.3.7	readAll	7
3.1.3.8	RequestForFileName	8
3.1.4	Property Documentation	8
3.1.4.1	Behaviour	8
3.1.4.2	FileName	8
3.2	NMF.INMFComputer Interface Reference	9
3.2.1	Detailed Description	9
3.2.2	Member Function Documentation	9
3.2.2.1	doNMF	9
3.2.2.2	PrintH	9

3.2.2.3	PrintW	9
3.2.2.4	RequestForFactorizationRank	10
3.3	NMF.MATRIX Class Reference	11
3.3.1	Detailed Description	12
3.3.2	Constructor & Destructor Documentation	13
3.3.2.1	MATRIX	13
3.3.2.2	MATRIX	13
3.3.3	Member Function Documentation	13
3.3.3.1	CollSum	13
3.3.3.2	DeleniPoPrvcich	13
3.3.3.3	EuclideanDistanceCalculus	14
3.3.3.4	NasobeniPoPrvcich	14
3.3.3.5	operator*	14
3.3.3.6	operator*	14
3.3.3.7	operator*	15
3.3.3.8	operator+	15
3.3.3.9	operator+	15
3.3.3.10	operator+	15
3.3.3.11	operator-	16
3.3.3.12	Print	16
3.3.3.13	Reduction	16
3.3.3.14	RowSum	16
3.3.3.15	ToArray	16
3.3.3.16	ToBool	17
3.3.3.17	ToInt	17
3.3.3.18	ToString	17
3.3.3.19	Trans	17
3.3.3.20	WholeSum	17
3.3.4	Property Documentation	17
3.3.4.1	this	17
3.3.4.2	X	18
3.3.4.3	Y	18
3.4	NMF.NMF Class Reference	19
3.4.1	Detailed Description	19
3.4.2	Constructor & Destructor Documentation	20
3.4.2.1	NMF	20

3.4.3	Member Function Documentation	20
3.4.3.1	doNMF	20
3.4.3.2	PrintH	20
3.4.3.3	PrintW	20
3.4.3.4	RequestForFactorizationRank	20
3.4.4	Member Data Documentation	20
3.4.4.1	rank	20
3.4.5	Property Documentation	21
3.4.5.1	_H	21
3.4.5.2	_W	21
3.5	NMF.Parser Class Reference	22
3.5.1	Detailed Description	22
3.5.2	Constructor & Destructor Documentation	22
3.5.2.1	Parser	22
3.5.3	Member Function Documentation	22
3.5.3.1	EncapsulateLine	22
3.5.3.2	ParseLine	23
3.5.3.3	TransformToDouble	23
3.6	NMF.Properties.Resources Class Reference	24
3.6.1	Detailed Description	24

Chapter 1

Class Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

NMF.FileHandler	5
NMF.INMFComputer	9
NMF.NMF	19
NMF.MATRIX	11
NMF.Parser	22
NMF.Properties.Resources	24

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

NMF.FileHandler (Provides possibilities for manipulating files. Reading, writing, existence check, requesting user for file name)	5
NMF.INMFComputer (Interface providing required fuctionality for NMF algorithm computer class. Each computer must implement this interface)	9
NMF.MATRIX (Encapsulates working with matrix. Provides ability to basic math operations such as adding, subtracting, dividing etc. Also can convert matrix into bool, integer, real number or into string)	11
NMF.NMF (Method for computation of NMF alogorithm. For easy switch between different NMF algorithms, each NMF algorithm class must implement INMFCounter interface)	19
NMF.Parser (Helps to parse string. It can split string acording to given delimiter into separated tokens, or joined this tokens, back by this delimiter)	22
NMF.Properties.Resources (A strongly-typed resource class, for looking up localized strings, etc)	24

Chapter 3

Class Documentation

3.1 NMF.FileHandler Class Reference

Provides possibilities for manipulating files. Reading, writing, existence check, requesting user for file name.

Public Member Functions

- [FileHandler](#) (string fn)
Constructor: Initializes fileName according to given arugment.
- [FileHandler](#) (string fn, IOModeType mode)
Constructor: Initializes class variable fileName and opens this file for reading or writing (according to second argument).
- void [open](#) (IOModeType mode)
Opens file for reading or writing.
- void [close](#) ()
Closes opened file.
- string [GetLine](#) ()
Gets one line from input file.
- void [PutLine](#) (string line)
Writes line into file.
- void [readAll](#) ()
Reads all input file.

Static Public Member Functions

- static string [RequestForFileName](#) ()
Asks user to set file name of file.

- static bool `FileExists` (string fileName)
Informs if given file exists or not.
- static bool `DirectoryExists` (string dirName)
Informs if give directory exists.

Properties

- string `FileName` [get, set]
Properties for FileName.
- Behaviour `Behaviour` [get, set]
Property for Behaviour: It adjusts whether to nadle exceptions or let them to be thrown and do not care about them.

3.1.1 Detailed Description

Provides possibilities for manipulating files. Reading, writing, existence check, requesting user for file name.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 NMF.FileHandler.FileHandler (string *fn*)

Constructor. Initializes fileName according to given arugment.

Parameters:

fn

3.1.2.2 NMF.FileHandler.FileHandler (string *fn*, IOModeType *mode*)

Constructor. Initializes class variable fileName and opens this file for reading or writing (according to second argument).

Parameters:

fn name of handled file

mode mode of working with file. It can be: FileMode.Open - for file reading (must already exists!) or FileMode.Append - for file writing (if not exists will be created)

3.1.3 Member Function Documentation

3.1.3.1 void NMF.FileHandler.close ()

Closes opened file.

3.1.3.2 static bool NMF.FileHandler.DirectoryExists (string *dirName*) [static]

Informs if give directory exists.

Parameters:

dirName directory to check its existence

Returns:

true, if directory exists, false if not

3.1.3.3 static bool NMF.FileHandler.FileExists (string *fileName*) [static]

Informs if given file exists or not.

Parameters:

fileName file to check its existence

Returns:

true, if file exists, false if not

3.1.3.4 string NMF.FileHandler.GetLine ()

Gets one line from input file.

Returns:

line from input file

3.1.3.5 void NMF.FileHandler.open (IOModeType *mode*)

Opens file for reading or writing.

Parameters:

mode mode of file opening: read or write

3.1.3.6 void NMF.FileHandler.PutLine (string *line*)

Writes line into file.

Parameters:

line line to write

3.1.3.7 void NMF.FileHandler.readAll ()

Reads all input file.

3.1.3.8 static string NMF.FileHandler.RequestForFileName () [static]

Asks user to set file name of file.

Returns:

file name from user

3.1.4 Property Documentation**3.1.4.1 Behaviour NMF.FileHandler.Behaviour** [get, set]

Property for Behaviour. It adjusts whether to handle exceptions or let them to be thrown and do not care about them.

3.1.4.2 string NMF.FileHandler.FileName [get, set]

Properties for FileName.

The documentation for this class was generated from the following file:

- D:/Skola/02_Z_SWI/NMF_Final/NMF/NMF/FileHandler.cs

3.2 NMF.INMFComputer Interface Reference

Interface providing required functionality for [NMF](#) algorithm computer class. Each computer must implement this interface.

Inherited by [NMF.NMF](#).

Public Member Functions

- int [RequestForFactorizationRank](#) (bool rQ)
Request user to specify value of factorization rank he/she prefer.
- void [doNMF](#) ()
A method performing the factorization itself based on iterative update untill the desired precision is met. The way how we determine precision and the iterative operations itselfs are described in Galina's analysis.
- void [PrintW](#) ()
A method used to print content of computed matrix W.
- void [PrintH](#) ()
A method used to print a content of Matrix H.

3.2.1 Detailed Description

Interface providing required functionality for [NMF](#) algorithm computer class. Each computer must implement this interface.

3.2.2 Member Function Documentation

3.2.2.1 void NMF.INMFComputer.doNMF ()

A method performing the factorization itself based on iterative update untill the desired precision is met. The way how we determine precision and the iterative operations itselfs are described in Galina's analysis.

Implemented in [NMF.NMF](#).

3.2.2.2 void NMF.INMFComputer.PrintH ()

A method used to print a content of Matrix H.

Implemented in [NMF.NMF](#).

3.2.2.3 void NMF.INMFComputer.PrintW ()

A method used to print content of computed matrix W.

Implemented in [NMF.NMF](#).

3.2.2.4 int NMF.INMFComputer.RequestForFactorizationRank (bool *rQ*)

Request user to specify value of factorization rank he/she prefer.

Parameters:

returnMax if set to true, only value of max factorization rank is returned and user is not asked. Else user is asked for value of rank

Returns:

value of factorization rank

Implemented in [NMF.NMF](#).

The documentation for this interface was generated from the following file:

- D:/Skola/02_Z_SWI/NMF_Final/NMF/NMF/INMFComputer.cs

3.3 NMF.MATRIX Class Reference

Encapsulates working with matrix. Provides ability to basic math operations such as adding, subtracting, dividing etc. Also can convert matrix into bool, integer, real number or into string.

Public Member Functions

- **MATRIX** (int a, int b)
Matrix constructor with two parameters describing dimension of the matrix. Initial values of matrix' elemets is 0.
- **MATRIX** (double[,] input)
Matrix constructor with one parameter - an array of numerical values. Dimension of matrix and values of it's elements is identical to the input array.
- void **Print** ()
Prints Matrix onto standard output. Heavily used during development and for testing of consistency. Might not be used so much in the future.
- double[,] **ToArray** ()
A conversion method.
- **MATRIX ToBool** ()
A conversion method returning content of Matrix.
- **MATRIX ToInt** ()
A conversion method returning content of Matrix.
- override string **ToString** ()
Overriden method converting object into string.
- **MATRIX Reduction** (int x, int y)
Method used to output a sub-content of Matrix.
- **MATRIX Trans** ()
Method used to return a transposed Matrix.
- double **EuclideanDistanceCalculus** (MATRIX a)
Methods which counts the distance of two Matrices.
- double **WholeSum** ()
Method summing up all elements in a Matrix.
- double **RowSum** (int row)
Method summing up all elements in a specified row.
- double **CollSum** (int coll)
Method summing up all elements in a specified column.
- **MATRIX NasobeniPoPrvcich** (MATRIX a)

A method multiplying two matrices in an "element by element" way.

- [MATRIX DeleniPoPrvcich \(MATRIX a\)](#)

A method dividing two matrices in an "element by element" way.

Static Public Member Functions

- static [MATRIX operator*](#) ([MATRIX a](#), [MATRIX b](#))

An operator overload with purpose to ease our work in the code, and to make it look nicer.

- static [MATRIX operator*](#) (double a, [MATRIX b](#))

An operator overload with purpose to ease our work in the code, and to make it look nicer.

- static [MATRIX operator*](#) ([MATRIX b](#), double a)

An operator overload with purpose to ease our work in the code, and to make it look nicer.

- static [MATRIX operator+](#) ([MATRIX a](#), [MATRIX b](#))

Operator overload for the addition operation upon Matrices.

- static [MATRIX operator+](#) (double a, [MATRIX b](#))

Operator overload for the addition operation upon Matrix and a number.

- static [MATRIX operator+](#) ([MATRIX b](#), double a)

Operator overload for the addition operation upon Matrix and a number.

- static [MATRIX operator-](#) ([MATRIX a](#), [MATRIX b](#))

Operator overload for the subtraction operation upon Matrices.

Properties

- double [this](#) [int row, int column] [get, set]

An operator [] which purpose is to ease our work with Matrix structures in this source code, basically treating it as an array.

- int [X](#) [get]

Getter used for methods not belonging to the Matrix class.

- int [Y](#) [get]

Getter used for methods not belonging to the Matrix class.

3.3.1 Detailed Description

Encapsulates working with matrix. Provides ability to basic math operations such as adding, subtracting, dividing etc. Also can convert matrix into bool, integer, real number or into string.

3.3.2 Constructor & Destructor Documentation

3.3.2.1 NMF.MATRIX.MATRIX (int *a*, int *b*)

Matrix constructor with two parameters describing dimension of the matrix. Initial values of matrix' elements is 0.

Parameters:

- a* Number of rows
- b* Number of collumns

3.3.2.2 NMF.MATRIX.MATRIX (double *input*[,])

Matrix constructor with one parameter - an array of numerical values. Dimension of matrix and values of it's elements is identical to the input array.

Parameters:

- input* An array of numbers

3.3.3 Member Function Documentation

3.3.3.1 double NMF.MATRIX.CollSum (int *coll*)

Method summing up all elements in a specified collumn.

Parameters:

- coll* An index of a collumn in the Matrix.

Returns:

- Returns the sum of all elements in the specified collumn.

3.3.3.2 MATRIX NMF.MATRIX.DeleniPoPrvcich (MATRIX *a*)

A method dividing two matrices in an "element by element" way.

Parameters:

- a* Matrix B with dimension m,n.

Returns:

- Returns a Matrix C with dimension m,n. $C[i,j] = \text{this}[i,j] / B[i,j]$.

3.3.3.3 double NMF.MATRIX.EuclideanDistanceCalculus (MATRIX *a*)

Methods which counts the distance of two Matrices.

Parameters:

a Matrix which we want to find a distance to.

Returns:

Returns a single number, the sum of squared substractions of each element of two Matrices.

3.3.3.4 MATRIX NMF.MATRIX.NasobeniPoPrvcich (MATRIX *a*)

A method multiplying two matrices in an "element by element" way.

Parameters:

a Matrix B with dimension m,n.

Returns:

Returns a Matrix C with dimension m,n. $C[i,j] = \text{this}[i,j] * B[i,j]$

3.3.3.5 static MATRIX NMF.MATRIX.operator* (MATRIX *b*, double *a*) [static]

An operator overload with purpose to ease our work in the code, and to make it look nicer.

Parameters:

a Describes Matrix A with dimension m,n

b Describes number B

Returns:

Returns a Matrix C with dimension m,n. $C[i,j] = A[i,j] * B$

3.3.3.6 static MATRIX NMF.MATRIX.operator* (double *a*, MATRIX *b*) [static]

An operator overload with purpose to ease our work in the code, and to make it look nicer.

Parameters:

a Describes number A

b Describes Matrix B with dimension m,n

Returns:

Returns a Matrix C with dimension m,n. $C[i,j] = B[i,j] * A$

3.3.3.7 static MATRIX NMF.MATRIX.operator* (MATRIX *a*, MATRIX *b*) [static]

An operator overload with purpose to ease our work in the code, and to make it look nicer.

Parameters:

- a* Describes Matrix A with dimension m,n
- b* Describes Matrix B with dimension n,p

Returns:

Returns a Matrix C with dimension m,p and it's elements as a result of the basic Matrix multiplication rule.

3.3.3.8 static MATRIX NMF.MATRIX.operator+ (MATRIX *b*, double *a*) [static]

Operator overload for the addition operation upon Matrix and a number.

Parameters:

- a* Matrix A with dimension m,n.
- b* Number B.

Returns:

Returns Matrix C with dimension m,n. $C[i,j] = A[i,j] + B$

3.3.3.9 static MATRIX NMF.MATRIX.operator+ (double *a*, MATRIX *b*) [static]

Operator overload for the addition operation upon Matrix and a number.

Parameters:

- a* Number A.
- b* Matrix B with dimension m,n.

Returns:

Returns Matrix C with dimension m,n. $C[i,j] = B[i,j] + A$

3.3.3.10 static MATRIX NMF.MATRIX.operator+ (MATRIX *a*, MATRIX *b*) [static]

Operator overload for the addition operation upon Matrices.

Parameters:

- a* Matrix A with dimension m,n.
- b* Matrix B with dimension m,n.

Returns:

Returns Matrix C with dimension m,n. $C[i,j] = A[i,j] + B[i,j]$

3.3.3.11 static MATRIX NMF.MATRIX.operator- (MATRIX *a*, MATRIX *b*) [static]

Operator overload for the subtraction operation upon Matrices.

Parameters:

a Matrix A with dimension m,n.

b Matrix B with dimension m,n.

Returns:

Returns Matrix C with dimension m,n. $C[i,j] = A[i,j] - B[i,j]$

3.3.3.12 void NMF.MATRIX.Print ()

Prints Matrix onto standard output. Heavily used during development and for testing of consistency. Might not be used so much in the future.

3.3.3.13 MATRIX NMF.MATRIX.Reduction (int *x*, int *y*)

Method used to output a sub-content of Matrix.

Parameters:

x Describes number of rows we want to output. Must be lower to or equal to the number of rows in original Matrix.

y Describes number of collumns we want to output. Must be lower to or equal to the number of collumns in original Matrix.

Returns:

Returns a Matrix with dimension x,y. Elements at positions x,y in original matrix and output matrix are identical.

3.3.3.14 double NMF.MATRIX.RowSum (int *row*)

Method summing up all elements in a specified row.

Parameters:

row An index of a row in the Matrix.

Returns:

Returns the sum of all elements in the specified row.

3.3.3.15 double [,] NMF.MATRIX.ToArray ()

A conversion method.

Returns:

Returns an array of floating point numbers.

3.3.3.16 MATRIX NMF.MATRIX.ToBool ()

A conversion method returning content of Matrix.

Returns:

Returns Matrix with elements either 0 or 1, which is decided by a conversion function. For example if the absolute value of element is lower then 0.5, it is converted to 0, otherwise it is converted to 1.

3.3.3.17 MATRIX NMF.MATRIX.ToInt ()

A conversion method returning content of Matrix.

Returns:

Returns Matrix with elements in intergal format, simple rounding of every element is performed.

3.3.3.18 override string NMF.MATRIX.ToString ()

Overriden method converting object into string.

Returns:

String representation of current object of matrix

3.3.3.19 MATRIX NMF.MATRIX.Trans ()

Method used to return a transposed Matrix.

Returns:

Returns transposed Matrix A with dimension x,y as Matrix B with dimension y,x and it's elements swapped around the main diagonal.

3.3.3.20 double NMF.MATRIX.WholeSum ()

Method summing up all elements in a Matrix.

Returns:

Returns a single number, sum of all elements in a Matrix.

3.3.4 Property Documentation

3.3.4.1 double NMF.MATRIX.this[int row, int column] [get, set]

An operator [] which purpose is to ease our work with Matrix structures in this source code, basically treating it as an array.

Parameters:

row Describes in which row our element of interest is.

column Describes in which column our element of interest is.

Returns:

Returns or sets the value of desired element.

3.3.4.2 int NMF.MATRIX.X [get]

Getter used for methods not belonging to the Matrix class.

Returns:

Returns number of rows in Matrix.

3.3.4.3 int NMF.MATRIX.Y [get]

Getter used for methods not belonging to the Matrix class.

Returns:

Returns number of columns in Matrix.

The documentation for this class was generated from the following file:

- D:/Skola/02_Z_SWI/NMF_Final/NMF/NMF/Matrix.cs

3.4 NMF.NMF Class Reference

Method for computation of [NMF](#) algorithm. For easy switch between different [NMF](#) algorithms, each [NMF](#) algorithm class must implement [INMFCounter](#) interface.

Inherits [NMF::INMFComputer](#).

Collaboration diagram for [NMF.NMF](#):

Public Member Functions

- [NMF](#) (double[,] inputMATRIX)
A constructor. Creates instance of [NMF](#) with input matrix of doubles.
- int [RequestForFactorizationRank](#) (bool rQ)
Request user to specify value of factorization rank he/she prefer.
- void [doNMF](#) ()
A method performing the factorization itself based on iterative update untill the desired precision is met. The way how we determine precision and the iterative operations itselfs are described in Galina's analysis.
- void [PrintW](#) ()
A method used to print content of computed matrix W.
- void [PrintH](#) ()
A method used to print a content of Matrix H.

Public Attributes

- int [rank](#)
Factorization rank used in [NMF](#).

Properties

- [MATRIX_W](#) [get, set]
Getter and Setter method for accesing content of Matrix W outside of [NMF](#) class.
- [MATRIX_H](#) [get, set]
Getter and Setter method for accesing content of Matrix H outside of [NMF](#) class.

3.4.1 Detailed Description

Method for computation of [NMF](#) algorithm. For easy switch between different [NMF](#) algorithms, each [NMF](#) algorithm class must implement [INMFCounter](#) interface.

3.4.2 Constructor & Destructor Documentation

3.4.2.1 NMF.NMF.NMF (double *inputMATRIX*[,])

A constructor. Creates instance of [NMF](#) with input matrix of doubles.

Parameters:

inputMATRIX An array of doubles specifying the content of the Matrix. This array is used to form a Matrix object which allows us to easily use all needed operations we defined in a Matrix class.

3.4.3 Member Function Documentation

3.4.3.1 void NMF.NMF.doNMF ()

A method performing the factorization itself based on iterative update until the desired precision is met. The way how we determine precision and the iterative operations themselves are described in Galina's analysis.

Implements [NMF.INMFComputer](#).

3.4.3.2 void NMF.NMF.PrintH ()

A method used to print a content of Matrix H.

Implements [NMF.INMFComputer](#).

3.4.3.3 void NMF.NMF.PrintW ()

A method used to print content of computed matrix W.

Implements [NMF.INMFComputer](#).

3.4.3.4 int NMF.NMF.RequestForFactorizationRank (bool *rQ*)

Request user to specify value of factorization rank he/she prefer.

Parameters:

returnMax if set to true, only value of max factorization rank is returned and user is not asked. Else user is asked for value of rank

Returns:

value of factorization rank

Implements [NMF.INMFComputer](#).

3.4.4 Member Data Documentation

3.4.4.1 int NMF.NMF.rank

Factorization rank used in [NMF](#).

3.4.5 Property Documentation

3.4.5.1 MATRIX NMF.NMF_H [get, set]

Getter and Setter method for accessing content of Matrix H outside of [NMF](#) class.

3.4.5.2 MATRIX NMF.NMF_W [get, set]

Getter and Setter method for accessing content of Matrix W outside of [NMF](#) class.

The documentation for this class was generated from the following file:

- D:/Skola/02_Z_SWI/NMF_Final/NMF/NMF/NMF.cs

3.5 NMF.Parser Class Reference

Helps to parse string. It can split string according to given delimiter into separated tokens, or joined this tokens, back by this delimiter.

Public Member Functions

- [Parser](#) (string *delim*)
Constructor. Initializes class variables.
- string[] [ParseLine](#) (string *line*)
Parses string line containing numbers according to given delimiter.
- double[,] [TransformToDouble](#) (string[,] *matrix*)
Transforms input array of strings into array of doubles.
- string [EncapsulateLine](#) (double[] *line*)
Does opposite as ParseLine. Gets array and each item joins with delimiter. It also handles required type of output array(does not no more);.

3.5.1 Detailed Description

Helps to parse string. It can split string according to given delimiter into separated tokens, or joined this tokens, back by this delimiter.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 NMF.Parser.Parser (string *delim*)

Constructor. Initializes class variables.

Parameters:

delim delimiter of numbers for line parsing

3.5.3 Member Function Documentation

3.5.3.1 string NMF.Parser.EncapsulateLine (double[] *line*)

Does opposite as ParseLine. Gets array and each item joins with delimiter. It also handles required type of output array(does not no more);.

Parameters:

line array of items on one line

Returns:

string of array of items and delimiters

3.5.3.2 string [] NMF.Parser.ParseLine (string *line*)

Parses string line containing numbers according to given delimiter.

Parameters:

line string to be parsed

Returns:

array of numbers

3.5.3.3 double [,] NMF.Parser.TransformToDouble (string *matrix*[,])

Transforms input array of strings into array of doubles.

Parameters:

matrix array to be transformed

Returns:

array of doubles

The documentation for this class was generated from the following file:

- D:/Skola/02_Z_SWI/NMF_Final/NMF/NMF/Parser.cs

3.6 NMF.Properties.Resources Class Reference

A strongly-typed resource class, for looking up localized strings, etc.

Collaboration diagram for NMF.Properties.Resources:

3.6.1 Detailed Description

A strongly-typed resource class, for looking up localized strings, etc.

The documentation for this class was generated from the following file:

- `D:/Skola/02_Z_SWI/NMF_Final/NMF/NMF/Properties/Resources.Designer.cs`

Index

- [_H](#)
 - [NMF::NMF, 21](#)
 - [_W](#)
 - [NMF::NMF, 21](#)
- [Behaviour](#)
 - [NMF::FileHandler, 8](#)
- [close](#)
 - [NMF::FileHandler, 6](#)
- [CollSum](#)
 - [NMF::MATRIX, 13](#)
- [DeleniPoPrvcich](#)
 - [NMF::MATRIX, 13](#)
- [DirectoryExists](#)
 - [NMF::FileHandler, 6](#)
- [doNMF](#)
 - [NMF::INMFComputer, 9](#)
 - [NMF::NMF, 20](#)
- [EncapsulateLine](#)
 - [NMF::Parser, 22](#)
- [EuclideanDistanceCalculus](#)
 - [NMF::MATRIX, 13](#)
- [FileExists](#)
 - [NMF::FileHandler, 7](#)
- [FileHandler](#)
 - [NMF::FileHandler, 6](#)
- [FileName](#)
 - [NMF::FileHandler, 8](#)
- [GetLine](#)
 - [NMF::FileHandler, 7](#)
- [MATRIX](#)
 - [NMF::MATRIX, 13](#)
- [NasobeniPoPrvcich](#)
 - [NMF::MATRIX, 14](#)
- [NMF](#)
 - [NMF::NMF, 20](#)
- [NMF::FileHandler, 5](#)
 - [Behaviour, 8](#)
 - [close, 6](#)
 - [DirectoryExists, 6](#)
 - [FileExists, 7](#)
 - [FileHandler, 6](#)
 - [FileName, 8](#)
 - [GetLine, 7](#)
 - [open, 7](#)
 - [PutLine, 7](#)
 - [readAll, 7](#)
 - [RequestForFileName, 7](#)
- [NMF::INMFComputer, 9](#)
 - [doNMF, 9](#)
 - [PrintH, 9](#)
 - [PrintW, 9](#)
 - [RequestForFactorizationRank, 9](#)
- [NMF::MATRIX, 11](#)
 - [CollSum, 13](#)
 - [DeleniPoPrvcich, 13](#)
 - [EuclideanDistanceCalculus, 13](#)
 - [MATRIX, 13](#)
 - [NasobeniPoPrvcich, 14](#)
 - [operator*, 14](#)
 - [operator+, 15](#)
 - [operator-, 15](#)
 - [Print, 16](#)
 - [Reduction, 16](#)
 - [RowSum, 16](#)
 - [this, 17](#)
 - [ToArray, 16](#)
 - [ToBool, 16](#)
 - [ToInt, 17](#)
 - [ToString, 17](#)
 - [Trans, 17](#)
 - [WholeSum, 17](#)
 - [X, 18](#)
 - [Y, 18](#)
- [NMF::NMF, 19](#)
 - [_H, 21](#)
 - [_W, 21](#)
 - [doNMF, 20](#)
 - [NMF, 20](#)
 - [PrintH, 20](#)
 - [PrintW, 20](#)
 - [rank, 20](#)
 - [RequestForFactorizationRank, 20](#)
- [NMF::Parser, 22](#)

- EncapsulateLine, 22
- ParseLine, 22
- Parser, 22
- TransformToDouble, 23
- NMF::Properties::Resources, 24
- open
 - NMF::FileHandler, 7
- operator*
 - NMF::MATRIX, 14
- operator+
 - NMF::MATRIX, 15
- operator-
 - NMF::MATRIX, 15
- ParseLine
 - NMF::Parser, 22
- Parser
 - NMF::Parser, 22
- Print
 - NMF::MATRIX, 16
- PrintH
 - NMF::INMFComputer, 9
 - NMF::NMF, 20
- PrintW
 - NMF::INMFComputer, 9
 - NMF::NMF, 20
- PutLine
 - NMF::FileHandler, 7
- rank
 - NMF::NMF, 20
- readAll
 - NMF::FileHandler, 7
- Reduction
 - NMF::MATRIX, 16
- RequestForFactorizationRank
 - NMF::INMFComputer, 9
 - NMF::NMF, 20
- RequestForFileName
 - NMF::FileHandler, 7
- RowSum
 - NMF::MATRIX, 16
- this
 - NMF::MATRIX, 17
- ToArray
 - NMF::MATRIX, 16
- ToBool
 - NMF::MATRIX, 16
- ToInt
 - NMF::MATRIX, 17
- ToString
 - NMF::MATRIX, 17
- Trans
 - NMF::MATRIX, 17
- TransformToDouble
 - NMF::Parser, 23
- WholeSum
 - NMF::MATRIX, 17
- X
 - NMF::MATRIX, 18
- Y
 - NMF::MATRIX, 18