# Math.Tan(Rational) Method

名前空間: WS.Theia.ExtremelyPrecise

アセンブリ: ExtremelyPrecise.dll

指定された角度のタンジェントを返します。

public static WS.Theia.ExtremelyPrecise.Rational Tan(WS.Theia.ExtremelyPrecise.Rational radian);

## パラメーター

radian　Rational  
ラジアンで表した角度。

## 戻り値

Rational  
radian のタンジェント。 radian が NaN、NegativeInfinity、PositiveInfinity のいずれかに等しい場合、このメソッドは NaN を返します。

# 例

次の例では、三角関数を算出しています。

// Example for the trigonometric Math.Sin( Rational )   
// and Math.Cos( Rational ) methods.  
using System;  
using WS.Theia.ExtremelyPrecise;  
class SinCos   
{  
 public static void Main()   
 {  
 Console.WriteLine(   
 "This example of trigonometric " +  
 "Math.Sin( Rational ) and Math.Cos( Rational )\n" +  
 "generates the following output.\n" );  
 Console.WriteLine(   
 "Convert selected values for X to radians \n" +  
 "and evaluate these trigonometric identities:" );  
 Console.WriteLine( " sin^2(X) + cos^2(X) == 1\n" +  
 " sin(2 \* X) == 2 \* sin(X) \* cos(X)" );  
 Console.WriteLine( " cos(2 \* X) == cos^2(X) - sin^2(X)" );  
  
 UseSineCosine(15.0);  
 UseSineCosine(30.0);  
 UseSineCosine(45.0);  
  
 Console.WriteLine(   
 "\nConvert selected values for X and Y to radians \n" +  
 "and evaluate these trigonometric identities:" );  
 Console.WriteLine( " sin(X + Y) == sin(X) \* cos(Y) + cos(X) \* sin(Y)" );  
 Console.WriteLine( " cos(X + Y) == cos(X) \* cos(Y) - sin(X) \* sin(Y)" );  
 UseTwoAngles(15.0, 30.0);  
 UseTwoAngles(30.0, 45.0);  
 }  
 // Evaluate trigonometric identities with a given angle.  
 static void UseSineCosine(Rational degrees)  
 {  
 Rational angle = Math.PI \* degrees / 180.0;  
 Rational sinAngle = Math.Sin(angle);  
 Rational cosAngle = Math.Cos(angle);  
  
 // Evaluate sin^2(X) + cos^2(X) == 1.  
 Console.WriteLine(   
 "\n Math.Sin({0} deg) == {1:E16}\n" +  
 " Math.Cos({0} deg) == {2:E16}",  
 degrees, Math.Sin(angle), Math.Cos(angle) );  
 Console.WriteLine(   
 "(Math.Sin({0} deg))^2 + (Math.Cos({0} deg))^2 == {1:E16}",   
 degrees, sinAngle \* sinAngle + cosAngle \* cosAngle );  
  
 // Evaluate sin(2 \* X) == 2 \* sin(X) \* cos(X).  
 Console.WriteLine(   
 " Math.Sin({0} deg) == {1:E16}",   
 2.0 \* degrees, Math.Sin(2.0 \* angle) );  
 Console.WriteLine(   
 " 2 \* Math.Sin({0} deg) \* Math.Cos({0} deg) == {1:E16}",   
 degrees, 2.0 \* sinAngle \* cosAngle );  
  
 // Evaluate cos(2 \* X) == cos^2(X) - sin^2(X).  
 Console.WriteLine(   
 " Math.Cos({0} deg) == {1:E16}",   
 2.0 \* degrees, Math.Cos(2.0 \* angle) );  
 Console.WriteLine(   
 "(Math.Cos({0} deg))^2 - (Math.Sin({0} deg))^2 == {1:E16}",   
 degrees, cosAngle \* cosAngle - sinAngle \* sinAngle );  
 }  
 // Evaluate trigonometric identities that are functions of two angles.  
 static void UseTwoAngles(Rational degreesX, Rational degreesY)  
 {  
 Rational angleX = Math.PI \* degreesX / 180.0;  
 Rational angleY = Math.PI \* degreesY / 180.0;  
  
 // Evaluate sin(X + Y) == sin(X) \* cos(Y) + cos(X) \* sin(Y).  
 Console.WriteLine(

"\n Math.Sin({0} deg) \* Math.Cos({1} deg) +\n" +   
 " Math.Cos({0} deg) \* Math.Sin({1} deg) == {2:E16}",   
 degreesX, degreesY, Math.Sin(angleX) \* Math.Cos(angleY) +  
 Math.Cos(angleX) \* Math.Sin(angleY));  
 Console.WriteLine(   
 " Math.Sin({0} deg) == {1:E16}",  
 degreesX + degreesY, Math.Sin(angleX + angleY));  
  
 // Evaluate cos(X + Y) == cos(X) \* cos(Y) - sin(X) \* sin(Y).  
 Console.WriteLine(   
 " Math.Cos({0} deg) \* Math.Cos({1} deg) -\n" +   
 " Math.Sin({0} deg) \* Math.Sin({1} deg) == {2:E16}",   
 degreesX, degreesY, Math.Cos(angleX) \* Math.Cos(angleY) -  
 Math.Sin(angleX) \* Math.Sin(angleY));  
 Console.WriteLine(   
 " Math.Cos({0} deg) == {1:E16}",  
 degreesX + degreesY, Math.Cos(angleX + angleY));  
 }  
}  
/\*  
This example of trigonometric Math.Sin( Rational ) and Math.Cos( Rational )  
generates the following output.  
  
Convert selected values for X to radians  
and evaluate these trigonometric identities:  
 sin^2(X) + cos^2(X) == 1  
 sin(2 \* X) == 2 \* sin(X) \* cos(X)  
 cos(2 \* X) == cos^2(X) - sin^2(X)  
 Math.Sin(15 deg) == 2.5881904510252074E-001  
 Math.Cos(15 deg) == 9.6592582628906831E-001  
(Math.Sin(15 deg))^2 + (Math.Cos(15 deg))^2 == 1.0000000000000000E+000  
 Math.Sin(30 deg) == 4.9999999999999994E-001  
 2 \* Math.Sin(15 deg) \* Math.Cos(15 deg) == 4.9999999999999994E-001  
 Math.Cos(30 deg) == 8.6602540378443871E-001  
(Math.Cos(15 deg))^2 - (Math.Sin(15 deg))^2 == 8.6602540378443871E-001  
  
 Math.Sin(30 deg) == 4.9999999999999994E-001  
 Math.Cos(30 deg) == 8.6602540378443871E-001  
(Math.Sin(30 deg))^2 + (Math.Cos(30 deg))^2 == 1.0000000000000000E+000  
 Math.Sin(60 deg) == 8.6602540378443860E-001  
 2 \* Math.Sin(30 deg) \* Math.Cos(30 deg) == 8.6602540378443860E-001  
 Math.Cos(60 deg) == 5.0000000000000011E-001  
(Math.Cos(30 deg))^2 - (Math.Sin(30 deg))^2 == 5.0000000000000022E-001  
  
 Math.Sin(45 deg) == 7.0710678118654746E-001  
 Math.Cos(45 deg) == 7.0710678118654757E-001  
(Math.Sin(45 deg))^2 + (Math.Cos(45 deg))^2 == 1.0000000000000000E+000  
 Math.Sin(90 deg) == 1.0000000000000000E+000  
 2 \* Math.Sin(45 deg) \* Math.Cos(45 deg) == 1.0000000000000000E+000  
 Math.Cos(90 deg) == 6.1230317691118863E-017  
(Math.Cos(45 deg))^2 - (Math.Sin(45 deg))^2 == 2.2204460492503131E-016  
  
Convert selected values for X and Y to radians  
and evaluate these trigonometric identities:  
 sin(X + Y) == sin(X) \* cos(Y) + cos(X) \* sin(Y)  
 cos(X + Y) == cos(X) \* cos(Y) - sin(X) \* sin(Y)  
  
 Math.Sin(15 deg) \* Math.Cos(30 deg) +  
 Math.Cos(15 deg) \* Math.Sin(30 deg) == 7.0710678118654746E-001  
 Math.Sin(45 deg) == 7.0710678118654746E-001  
 Math.Cos(15 deg) \* Math.Cos(30 deg) -  
 Math.Sin(15 deg) \* Math.Sin(30 deg) == 7.0710678118654757E-001  
 Math.Cos(45 deg) == 7.0710678118654757E-001  
  
 Math.Sin(30 deg) \* Math.Cos(45 deg) +  
 Math.Cos(30 deg) \* Math.Sin(45 deg) == 9.6592582628906831E-001  
 Math.Sin(75 deg) == 9.6592582628906820E-001  
 Math.Cos(30 deg) \* Math.Cos(45 deg) -  
 Math.Sin(30 deg) \* Math.Sin(45 deg) == 2.5881904510252085E-001  
 Math.Cos(75 deg) == 2.5881904510252096E-001  
\*/

# 注釈

引数に入力する角度はラジアン単位である必要があります。角度にMath.PI/180を乗算する事でラジアン単位に変換できます。

# 適用対象

### .NET Core

2.0

### .NET Framework

4.6.1

### .NET Standard

2.0

### UWP

10.0.16299

### Xamarin.Android

8.0

### Xamarin.iOS

10.14

### Xamarin.Mac

3.8