

```
#region Copyright (c) 2019 TRUST 47 Fund. All Rights Reserved.
//-----//
//          TRUST 47 Fund          //
//          Copyright (c) 2019 All Rights reserved //
//          //                      //
//          //                      //
// This file and its contents are protected by United States and //
// International copyright laws. Unauthorized reproduction and/or //
// distribution of all or any portion of the code contained herein //
// is strictly prohibited and will result in severe civil and criminal //
// penalties. Any violations of this copyright will be prosecuted //
// to the fullest extent possible under law. //
//          //                      //
// THE SOURCE CODE CONTAINED HEREIN AND IN RELATED FILES IS PROVIDED //
// TO THE REGISTERED DEVELOPER FOR THE PURPOSES OF EDUCATION AND //
// TROUBLESHOOTING. UNDER NO CIRCUMSTANCES MAY ANY PORTION OF THE SOURCE //
// CODE BE DISTRIBUTED, DISCLOSED OR OTHERWISE MADE AVAILABLE TO ANY //
// THIRD PARTY WITHOUT THE EXPRESS WRITTEN CONSENT OF TRUST 47 Fund //
//          //                      //
// UNDER NO CIRCUMSTANCES MAY THE SOURCE CODE BE USED IN WHOLE OR IN //
// PART, AS THE BASIS FOR CREATING A PRODUCT THAT PROVIDES THE SAME, OR //
// SUBSTANTIALLY THE SAME, FUNCTIONALITY AS ANY TRUST 47 Fund PRODUCT. //
//          //                      //
// THE REGISTERED DEVELOPER ACKNOWLEDGES THAT THIS SOURCE CODE //
// CONTAINS VALUABLE AND PROPRIETARY TRADE SECRETS OF TRUST 47 Fund //
// THE REGISTERED DEVELOPER AGREES TO EXPEND EVERY EFFORT TO //
// INSURE ITS CONFIDENTIALITY. //
//          //                      //
// THE END USER LICENSE AGREEMENT (EULA) ACCOMPANYING THE PRODUCT //
// PERMITS THE REGISTERED DEVELOPER TO REDISTRIBUTE THE PRODUCT IN //
// EXECUTABLE FORM ONLY IN SUPPORT OF APPLICATIONS WRITTEN USING //
// THE PRODUCT. IT DOES NOT PROVIDE ANY RIGHTS REGARDING THE //
// SOURCE CODE CONTAINED HEREIN. //
//          //                      //
// THIS COPYRIGHT NOTICE MAY NOT BE REMOVED FROM THIS FILE. //
//-----//
#endregion Copyright (c) 2019 TRUST 47 Fund. All Rights Reserved.
```

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Security.Cryptography;

#region No Warranty
/*=====
 * NO WARRANTY
 * Mediacrypt/IT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED,
 * INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OR MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE SOFTWARE
 * AND THE ACCOMPANYING WRITTEN MATERIALS. NO LIABILITY FOR CONSEQUENTIAL DAMAGES.
 * IN NO EVENT SHALL Mediacrypt/IT OR ITS SUPPLIERS BE LIABLE
 * FOR ANY DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION,
 * DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION,
 * LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS)
 * ARISING OUT OF THE USE OF OR INABILITY TO USE THIS Mediacrypt/IT PRODUCT,
 * EVEN IF Mediacrypt/IT HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
 *=====*/
#endregion No Warranty

namespace NXTBlocks.Blockchain.Demo05
{
    public class Block
    {
        public int Index { get; set; }
        public DateTime TimeStamp { get; set; }
        public string PreviousHash { get; set; }
        public string Hash { get; set; }
        public string Data { get; set; }
        public int Nonce { get; set; } = 0;

        public Block(DateTime timeStamp, string previousHash, string data)
        {
            Index = 0;
            TimeStamp = timeStamp;
            PreviousHash = previousHash;
            Data = data;
        }
    }
}
```

```
        Hash = CalculateHash();
    }

    public string CalculateHash()
    {
        string result = string.Empty;

        SHA256 sha256 = SHA256.Create();

        Encoding utf8 = Encoding.UTF8;

        string inputString = string.Empty;
        string outputString = string.Empty;

        try
        {
            inputString = $"{TimeStamp}-{PreviousHash ?? ""}-{Data}-{Nonce}";

            byte[] inputBytes = utf8.GetBytes(inputString);

            byte[] outputBytes = sha256.ComputeHash(inputBytes);

            result = Convert.ToBase64String(outputBytes);
        }
        catch (Exception ex)
        {
            Console.WriteLine("Exception: \n" + ex.Message);
        }

        return result;
    }

    public void Mine(int difficulty)
    {
        var leadingZeros = new string('0', difficulty);
        while (this.Hash == null || this.Hash.Substring(0, difficulty) != leadingZeros)
        {
            this.Nonce++;
        }
    }
}
```

```
        this.Hash = this.CalculateHash();  
    }  
}  
}
```