

Forum:[Hack Forums](#)

Thread: [Checking for identical object](#)

GuRuHaX - December 22, 2015, 11:36 am

Hello visitors and welcome,

I have a small issue concerning an app i am working on.

Its a simple app that handles input (persons name birthdate and so on),
i have a Class person that has "persons name birthdate and so on" as parameters.

Everything is ok and working exept i can't figure out how to check
if the object already exist,in the sense that when the Create button is clicked , i want to check if first
name & last name (parameters)already exists in the Repository (Created class to store a list of
persons).

I tryed the Contains() method but its not working, also tried
a foreach iteration with a check if firstname & lastname from object is same as new input but also
here its not working...

This should be simple and i did this before but i am overseeing something here so i call for a little
help :)

Ps: I debugged and know whats wrong ,just don't now how to solve it.

Peace !
?

+Haze - December 22, 2015, 12:11 pm

So, what are the conditions that must be met for an object to be identical?

Zilog80 - December 22, 2015, 12:17 pm

This should work. Hope you get the idea: (code not tested)

```
Code var Foo = Repository.FirstOrDefault(x=>x.FirstName=="John" & x.LastName=="Doe");  
if (Foo != null)  
    MessageBox.Show("already exists.");  
else  
    // Do something
```

GuRuHaX - December 22, 2015, 12:32 pm

(Dec 22, 2015 01:11 PM)+Haze Wrote: [CITE](#) So, what are the conditions that must be met for an
object to be identical?

If they have the same firstName and lastName then they would be identical .

it must be possible to have 2 persons with the same firstName OR 2 persons with the same
lastName but not 2 persons with the same first & lastName.

?

Thanx Zilog80 will try it out and let you know.

Peace

Compoze - December 22, 2015, 12:53 pm

You could override the == operator and check the different conditions for two object to be identical, in this case if they have the same first name and last name:

PHP Code

```
<?php
public static bool operator ==(Person a, Person b)
{
    return (a.FirstName == b.FirstName) && (a.LastName == b.LastName);
}
```

and then you can do like this:

PHP Code

```
<?php
Person p1 = new Person("David", "Jase");
Person p2 = new Person("David", "Jase");

if(p1 == p2) // This returns true
{

}
```

+Haze - December 22, 2015, 12:59 pm

(Dec 22, 2015 01:53 PM)Compoze Wrote: [\[CITE\]](#) You could override the == operator and check the different conditions for two object to be identical, in this case if they have the same first name and last name:

PHP Code

```
<?php
public static bool operator ==(Person a, Person b)
{
    return (a.FirstName == b.FirstName) && (a.LastName == b.LastName);
}
```

and then you can do like this:

PHP Code

```
<?php
Person p1 = new Person("David", "Jase");
Person p2 = new Person("David", "Jase");

if(p1 == p2) // This returns true
{

}
```

this. Was just about to suggest that.

Also @op you can override the Equals method too.

PHP Code

```
<?php
public override bool Equals(object obj)
{
    if (obj is Person)
    {
        var p = (Person) obj;
        return p.Firstname == Firstname && p.Lastname == Lastname;
    }

    return base.Equals(obj);
}
```

There's really lots of way to do this.

SL x TnT - December 22, 2015, 2:01 pm

(Dec 22, 2015 01:59 PM)+Haze Wrote: [CITE](#) this. Was just about to suggest that.

Also @op you can override the Equals method too.

PHP Code

```
<?php
public override bool Equals(object obj)
{
    if (obj is Person)
    {
        var p = (Person) obj;
        return p.Firstname == Firstname && p.Lastname == Lastname;
    }

    return base.Equals(obj);
}
```

There's really lots of way to do this.

and if you want to use a Dictionary, HashSet, Distinct extension, etc, override GetHashCode too.
Might be worthwhile doing that either way.

GuRuHaX - December 22, 2015, 2:01 pm

+Haze Thanks a lot sir !

Im gonna sleep now will check this out first thing in the morning !

Didn't think to override good idea !

Will post the result here tommorrow !

?

Peace !

(Dec 22, 2015 03:01 PM)SL x TnT Wrote: [CITE](#) and if you want to use a Dictionary, HashSet, Distinct extension, etc, override GetHashCode too. Might be worthwhile doing that either way.

Yes indeed if override equal must override GetHashCode to.

will let you guys know tommorrow ;)

Thanks a lot for your time!

Peace

GuRuHaX - December 23, 2015, 2:14 am

Hello everybody ,

I cannot post links yet so im posting my code.?

Little explanation first,

The person object i talked about earlier are named Yogi ,
I did what +Haze said and override bool operator,
In debugging i saw that this was indeed working (so thank you already for this + Haze !).

Now i have a last issue that i cannot seem to resolve and it frustrates me i must say...?
Issue is put in comments in below code

```
Yogi newYogi = new Yogi (firstName, lastName,...);
    // if not empty list
    if (repository.yogis.Count != 0)
    {
        List<Yogi> temp = new List<Yogi>();
        //check every yogi in repository list
        foreach (Yogi in repository.yogis)
        {
            if (y == newYogi) // this is the operator override that checks the first and
lastName
                {
                    MessageBox.Show("This yogi already exists", "Double yogi not allowed",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
                    break;
                }
            else
            {
                temp.Add(newYogi);
            }
        }
        // PROBLEM 1
        // this always adds the yogi since you always come out the foreach...
        repository.yogis.Add(newYogi);

        /* problem is here , i want to add the newYogi ONLY when it doesn't exist in
repository.yogis
        Since you cannot modify inside a foreach ,i tried with a temporary list but
something is wrong with my logic i know that , i am not able to figure it out... */
```

}

I know i am close and this is pretty basic stuff but just can't find the right logic ...

Kind regards

SL x TnT - December 23, 2015, 3:22 am

(Dec 23, 2015 03:14 AM)GuRuHaX Wrote: [\[CITE\]](#) PHP Code

```
<?php
Yogi newYogi = new Yogi (firstName, lastName,...);
// if not empty list
if (repository.yogis.Count != 0)
{
    List<Yogi> temp = new List<Yogi>();
    //check every yogi in repository list
    foreach (Yogi in repository.yogis)
    {
        if (y == newYogi) // this is the operator override that checks the first and lastName
        {
            MessageBox.Show("This yogi already exists", "Double yogi not allowed", MessageBoxButtons.OK,
            MessageBoxIcon.Warning);
            break;
        }
        else
        {
            temp.Add(newYogi);
        }
    }
    // PROBLEM 1
    // this always adds the yogi since you always come out the foreach...
    repository.yogis.Add(newYogi);

    /* problem is here , i want to add the newYogi ONLY when it doesn't exist in repository.yogis
    Since you cannot modify inside a foreach ,i tried with a temporary list but
    something is wrong with my logic i know that , i am not able to figure it out... */
}
```

Easier to read for others.

You're looping through all the objects.

- Checking if the current object is equal to the new object.
- - If it is, break out of the loop
- - Else, add the new object to a temp List

Instead, you can do this.

Loop through all the objects.

- Check if the current object is equal to the new object.
- - If it is, set a variable to indicate a duplicate. Break out of loop.

If the duplicate variable is not set, add new object to the List.

There are other ways to do this, but that depends on how you're using the data. Looks like yours is a single add from user input, so it'll be fine.

(Dec 23, 2015 04:22 AM)SL x TnT Wrote: [\[CITE\]](#) Easier to read for others.

You're looping through all the objects.

- Checking if the current object is equal to the new object.
- - If it is, break out of the loop
- - Else, add the new object to a temp List

Instead, you can do this.

Loop through all the objects.

- Check if the current object is equal to the new object.
- - If it is, set a variable to indicate a duplicate. Break out of loop.

If the duplicate variable is not set, add new object to the List.

There are other ways to do this, but that depends on how you're using the data. Looks like yours is a single add from user input, so it'll be fine. Sry didn't think of it ;) will use this from now on.

It worked well thank you verry much for this , i used a bool and set it on true when already existing and checked that after the foreach. I feel happy again ?

Thank you all for the help !

(I did not use the bool override operator cause i use it for other things too and that wouldn't have worked for those so i just check the FName & Lname in the if)

Here is the changed code:

PHP Code

```
<?php
foreach (Yogi y in repository.yogis)
{
    // already exist
    if (y.FirstName == newYogi.FirstName && y.LastName == newYogi.LastName)
    {
        exist = true;
        break;
    }
}
if (exist != true)
{
    AddYogi_ClearForm_ShowCreation_Message(newYogi);
}
else
{
    MessageBox.Show("This yogi already exists", "Double yogi not allowed",
    MessageBoxButtons.OK, MessageBoxIcon.Warning);
}
```

I would have another question about the program but its about PDF creation with iTextsharp , when debugging everything works fine and pdf is created succesfully but when i run the "app.exe" it doesn't created it ,no errors or crashes,even get the created succesfull message but no PDF anywhere...

Kind Regards ?

Jopper - December 23, 2015, 12:04 pm

(Dec 23, 2015 07:30 AM)GuRuHaX Wrote: [\[CITE\]](#) Sry didn't think of it ;) will use this from now on.

It worked well thank you verry much for this , i used a bool and set it on true when already existing and checked that after the foreach. I feel happy again ?

Thank you all for the help !

(I did not use the bool override operator cause i use it for other things too and that wouldn't have worked for those so i just check the Fname & Lname in the if)

Here is the changed code:

PHP Code

```
<?php
foreach (Yogi y in repository.yogis)
{
    // already exist
    if (y.FirstName == newYogi.FirstName && y.LastName == newYogi.LastName)
    {
        exist = true;
        break;
    }
}
if (exist != true)
{
    AddYogi_ClearForm_ShowCreation_Message(newYogi);
}
else
{
    MessageBox.Show("This yogi already exists", "Double yogi not allowed",
    MessageBoxButtons.OK, MessageBoxIcon.Warning);
}
```

I would have another question about the program but its about PDF creation with iTextsharp , when debugging everything works fine and pdf is created succesfully but when i run the "app.exe" it doesn't created it ,no errors or crashes,even get the created succesfull message but no PDF anywhere...

Kind Regards ?

PM me about your other question and I will help you fix it. :)

Also, it is a good idea to override your bool operator to compare your "Yogi" objects. The behavior should make the same checks as your if statement does. In different situations, you could always change the parameters that the operator overloading method takes and handle them differently accordingly. Overriding the bool operators instead of manually checking each time also significantly reduces the chance of misusing the datatype.

GuRuHaX - December 24, 2015, 2:32 am

(Dec 23, 2015 01:04 PM)Jopper Wrote: [\[CITE\]](#) PM me about your other question and I will help you fix it. :)

Also, it is a good idea to override your bool operator to compare your "Yogi" objects. The behavior should make the same checks as your if statement does. In different situations, you could always change the parameters that the operator overloading method takes and handle them differently accordingly. Overriding the bool operators instead of manually checking each time also significantly

reduces the chance of misusing the datatype.

Thanks for the advice ,i already changed it ,like you say its a little more performant than checking every time .

I cannot yet send PM must have 25 post so im almost there ?

0xDEAD10CC - December 25, 2015, 6:29 pm

Something noteworthy that lots of .NET people are forgetting here:

[https://msdn.microsoft.com/en-us/library/ms132151\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms132151(v=vs.110).aspx)

[https://msdn.microsoft.com/en-us/library/ms131187\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms131187(v=vs.110).aspx)

These interfaces allow you to extend and expose comparison and equality features for user-defined objects to things that would otherwise not know about the capabilities. So although extending functionality and operator overloading is great, by itself it's nearly useless to most generic external implementations.

Additionally == is typically overloaded to use Equals() in some cases. If you just overload the == operator, what if somewhere along in the code you intend on using Equals()? Overloading == along is a horrible solution without the proper setup. Example of what I'm talking about:

Code public class MyObject

```
{
    public string FirstName { get; }
    public string LastName { get; }

    public MyObject(string firstName, string lastName)
    {
        FirstName = firstName;
        LastName = lastName;
    }

    public static bool operator ==(MyObject lhs, MyObject rhs)
    {
        return (lhs.FirstName == rhs.FirstName) && (lhs.LastName == rhs.LastName);
    }

    public static bool operator !=(MyObject lhs, MyObject rhs)
    {
        return (lhs.FirstName != rhs.FirstName) || (lhs.LastName != rhs.LastName);
    }
}

public static void Main(string[] args)
{
    MyObject a = new MyObject("Bob", "Smith");
    MyObject b = new MyObject("Bob", "Smith");

    Console.WriteLine(a == b ? "Equal!" : "Not equal...");
    Console.WriteLine(a.Equals(b) ? "Equal!" : "Not equal...");

    Console.ReadKey();
}
Result?
```

Code Equal!

Not equal...

Pretty contradictory for functions/operators that should yield the same result based on semantics?

Something like this is the PROPER way to achieve comparison for equality:

Code class Yogi : IEquatable<Yogi>

```
{
    private static uint idCount = 0;

    public uint ID { get; }
    public string FirstName { get; }
    public string LastName { get; }

    public Yogi(string firstName, string lastName)
    {
        ID = ++idCount;
        FirstName = firstName;
        LastName = lastName;
    }

    public bool Equals(Yogi other)
    {
        if (ReferenceEquals(null, other)) return false;
        if (ReferenceEquals(this, other)) return true;
        return string.Equals(FirstName, other.FirstName)
            && string.Equals(LastName, other.LastName);
    }

    public override bool Equals(object obj)
    {
        if (ReferenceEquals(null, obj)) return false;
        if (ReferenceEquals(this, obj)) return true;
        return obj.GetType() == GetType() && Equals((Yogi)obj);
    }

    public override int GetHashCode()
    {
        return ((FirstName?.GetHashCode() ?? 0) * 397)
            ^ (LastName?.GetHashCode() ?? 0);
    }

    public static bool operator ==(Yogi lhs, Yogi rhs)
    {
        return !ReferenceEquals(null, lhs)
            && lhs.Equals(rhs);
    }

    public static bool operator !=(Yogi lhs, Yogi rhs)
    {
        return !ReferenceEquals(null, lhs)
            && !lhs.Equals(rhs);
    }
}
```

```

public override string ToString()
{
    return $"{FirstName} {LastName}";
}
}

```

```

public static void Main(string[] args)
{
    Yogi y1 = new Yogi("Yogi", "Bear");
    Console.WriteLine("[{0}] {1}", y1.ID, y1);
    Yogi y2 = new Yogi("Yogi", "Bear");
    Console.WriteLine("[{0}] {1}", y2.ID, y2);
    Yogi y3 = new Yogi("Yogi", "Fake");
    Console.WriteLine("[{0}] {1}", y3.ID, y3);

```

```

    Console.WriteLine();

```

```

    Console.WriteLine("{0} {1} {2}", $"{y1.ID} {y1}", y1 == y2 ? "==" : "!=" , $"{y2.ID} {y2}");
    Console.WriteLine("{0} {1} {2}", $"{y2.ID} {y2}", y2 == y3 ? "==" : "!=" , $"{y3.ID} {y3}");
    Console.WriteLine("{0} {1} {2}", $"{y1.ID} {y1}", y1 == y3 ? "==" : "!=" , $"{y3.ID} {y3}");

```

```

    Console.ReadKey();
}

```

Result:

Code 1. Yogi Bear

2. Yogi Bear

3. Yogi Fake

Yogi Bear == Yogi Bear

Yogi Bear != Yogi Fake

Yogi Bear != Yogi Fake

(Dec 23, 2015 04:22 AM)SL x TnT Wrote: [CITE](#) Instead, you can do this.

Loop through all the objects.

- Check if the current object is equal to the new object.

- - If it is, set a variable to indicate a duplicate. Break out of loop.

If the duplicate variable is not set, add new object to the List.

Technically Adding data should be a new function, which would make the boolean variable idea redundant since you can just return from the function (with or without some return value/type to indicate success or failure -- it's all up for debate when considering application design).

GuRuHaX - December 27, 2015, 9:53 am

Quote:

PHP Code

```
<?php
```

Something like this is the PROPER way to achieve comparison for equality:

Code:

```
class Yogi : IEquatable<Yogi>
```

```
{
```

```
    private static uint idCount = 0;
```

```
    public uint ID { get; }
```

```

public string FirstName { get; }
public string LastName { get; }

public Yogi(string firstName, string lastName)
{
    ID = ++idCount;
    FirstName = firstName;
    LastName = lastName;
}

public bool Equals(Yogi other)
{
    if (ReferenceEquals(null, other)) return false;
    if (ReferenceEquals(this, other)) return true;
    return string.Equals(FirstName, other.FirstName)
        && string.Equals(LastName, other.LastName);
}

public override bool Equals(object obj)
{
    if (ReferenceEquals(null, obj)) return false;
    if (ReferenceEquals(this, obj)) return true;
    return obj.GetType() == GetType() && Equals((Yogi)obj);
}

public override int GetHashCode()
{
    return ((FirstName?.GetHashCode() ?? 0) * 397)
        ^ (LastName?.GetHashCode() ?? 0);
}

public static bool operator ==(Yogi lhs, Yogi rhs)
{
    return !ReferenceEquals(null, lhs)
        && lhs.Equals(rhs);
}

public static bool operator !=(Yogi lhs, Yogi rhs)
{
    return !ReferenceEquals(null, lhs)
        && !lhs.Equals(rhs);
}

public override string ToString()
{
    return $"{FirstName} {LastName}";
}

public static void Main(string[] args)
{
    Yogi y1 = new Yogi("Yogi", "Bear");
}

```

```

Console.WriteLine("{0} {1}", y1.ID, y1);
Yogi y2 = new Yogi("Yogi", "Bear");
Console.WriteLine("{0} {1}", y2.ID, y2);
Yogi y3 = new Yogi("Yogi", "Fake");
Console.WriteLine("{0} {1}", y3.ID, y3);

Console.WriteLine();

Console.WriteLine("{0} {1} {2}", $"{y1.ID} {y1}", y1 == y2 ? "==" : "!=", $"{y2.ID} {y2}");
Console.WriteLine("{0} {1} {2}", $"{y2.ID} {y2}", y2 == y3 ? "==" : "!=", $"{y3.ID} {y3}");
Console.WriteLine("{0} {1} {2}", $"{y1.ID} {y1}", y1 == y3 ? "==" : "!=", $"{y3.ID} {y3}");

Console.ReadKey();
}

```

Result:

Code:

1. Yogi Bear

2. Yogi Bear

3. Yogi Fake

Yogi Bear == Yogi Bear

Yogi Bear != Yogi Fake

Yogi Bear != Yogi Fake

THANK YOU very much for this well defined and specific answer !!

I didn't thought off using interfaces and this is the perfect situation for using it...

Thanks fo taking the time to have explained this in detail and explained it as you did !

Kind Regards to all !

Thanks for helping me out !?