

# Re: Newbie on Inheritance, Base classes and derived classes

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2005-06/msg03590.html>

---

- *From:* "Morten Wennevik" <[MortenWennevik@xxxxxxxxxxxxx](mailto:MortenWennevik@xxxxxxxxxxxxx)>
  - *Date:* Tue, 21 Jun 2005 16:20:47 +0200
- 

Hi Andy,

You seem to have grasped the inheritance. In fact any derived class can be referred to as the pa

```
Watch w = new DigitalWatch();
```

Only methods defined in Watch will be visible to w.

Furthermore, objects retain knowledge of what they really are so you can cast this Watch back to

```
DigitalWatch d = (DigitalWatch)w;
```

Considering a list of digital and analog watches.

```
ArrayList list = new ArrayList();  
list.Add(new DigitalWatch());  
list.Add(new AnalogWatch());  
list.Add(new DigitalWatch());  
list.Add(new DigitalWatch());  
list.Add(new AnalogWatch());
```

You can then do

```
foreach(Watch w in list)  
{
```

## Re: Newbie on Inheritance, Base classes and derived classes

```
w.SetTime();  
}
```

If you need different ways to set time for analog and digital watches, you can mark the base method as virtual.

In an external class, for instance in a button click event:

```
Watch w = new Watch();  
w.DoStuff();  
w = new DigitalWatch();  
w.DoStuff();  
w = new AnalogWatch();  
w.DoStuff();
```

```
class Watch  
{  
    protected virtual void SetTime(DateTime time){MessageBox.Show("Watch");}  
    public void DoStuff(){SetTime(DateTime.Now);}  
}
```

```
class DigitalWatch : Watch  
{  
    protected override void SetTime(DateTime time){MessageBox.Show("Digital");}  
    void ChangeBattery(){}  
}
```

```
class AnalogWatch : Watch  
{  
    void SetTime(DateTime time){MessageBox.Show("Analog");}  
    void WindUp(){}  
}
```

It will output Watch-Digital-Watch since AnalogWatch does not override SetTime, calling DoStuff will call the base class method.

In addition to calling common base class methods you can implement interfaces and use a common interface for all classes.

## Re: Newbie on Inheritance, Base classes and derived classes

On Tue, 21 Jun 2005 13:12:58 +0200, aaj <aaj@xxxxxxx> wrote:

Hi all

I'm new to OOP and just getting to grips with inheritance.

Could anyone tell me if my understanding so far is correct.....?

If I have a base class, and then some derived classes with extra functionality e.g.

```
Base Class : Watch    Method SetTime
Derived Class : DigitalWatch  Method ChangeBattery
Derived Class : AnalogWatch  Method WindUp
```

is it acceptable to use the base class type to access the SetTime method in all the watches and then in the same app to use the derived class type for only digital watches

so in one place use

```
Watch CurrentWatch = (watch) AndysDigitalWatch; i.e. casting Digital watch
as a base type watch and only seeing the SetTime Method
```

and in another place use

```
DigitalWatch CurrentWatch = AndysDigitalWatch;
```

obviously it wouldn't be exactly like this, I'm thinking more when using

## Re: Newbie on Inheritance, Base classes and derived classes

foreach on the base class object, bringing back all the watches derived from the base class, but only the methods common to the base class and the derived class.

So in summary, I guess what I'm asking is - is it acceptable to declare types of Base class to do some functions common to all objects, and in the same app, but in a different place, use the more specialised derived classes

thanks

Andy

--  
Happy coding!  
Morten Wennevik [C# MVP]  
.