

## Re: OO Design question

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2005-12/msg04053.html>

---

- *From:* "KJ" <n o s p a m@xxxxxxx>
  - *Date:* 20 Dec 2005 11:57:43 -0800
- 

Today your base class *\*only\** has common data. That may change tomorrow, when you realize you need methods x y and z in your base class.

Pure data-only objects are well-expressed using structs. I suggest instantiating your structs as private members of the base class, and provide protected accessors to the structs' members so that the subclasses can use them. This is sometimes known as multiple-inheritance by delegation, and it helps keep things neat. It also happens to be far more flexible than using multiple levels of inheritance (base->subclass level1->subclass level 2 ...).

For example:

```
namespace Example
{
public class BaseClass
{
private PureData _pd;

public BaseClass() : this(int.MinValue, string.Empty) {}

public BaseClass(int id, string name)
{
_pd = new PureData(id, name);
}

protected string Name
{
get
{
return _pd.Name;
}
set
{
_pd.Name = value;
}
}
```

Re: OO Design question

```
protected int ID
{
get
{
return _pd.ID;
}
set
{
_pd.ID = value;
}
}
}
```

```
public struct PureData
{
private int _id;
private string _name;
```

```
public string Name
{
get
{
return _name;
}
set
{
_name = value;
}
}
```

```
public int ID
{
get
{
return _id;
}
set
{
_id = value;
}
}
```

```
public PureData(int id, string name)
{
_id = id;
_name = name;
}
}
```

.

- *Follow-Ups:*
  - ◆ *Re: OO Design question*
    - ◇ *From:* Bruce Wood
  
- *References:*
  - ◆ *OO Design question*
    - ◇ *From:* Kalpesh
  
- Prev by Date: *Re: OO Design question*
- Next by Date: *Re: HTML rendering component*
- Previous by thread: *Re: OO Design question*
- Next by thread: *Re: OO Design question*
- Index(es):
  - ◆ *Date*
  - ◆ *Thread*