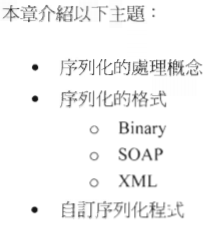
P 187

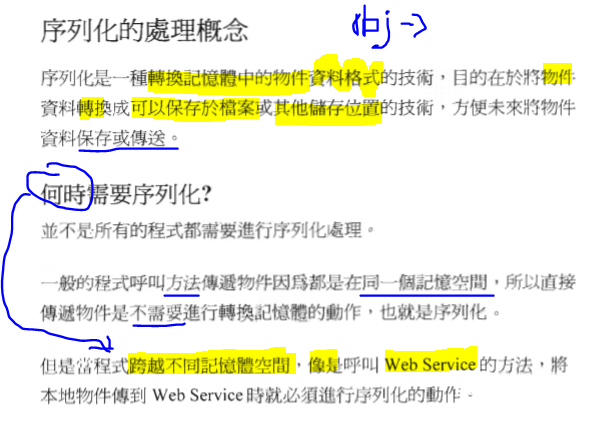


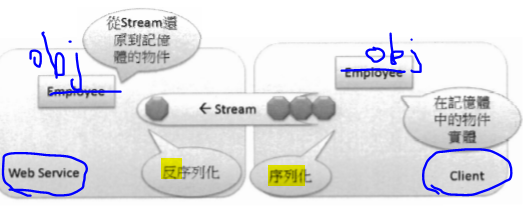
1

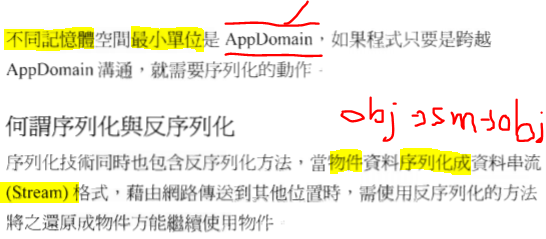
2

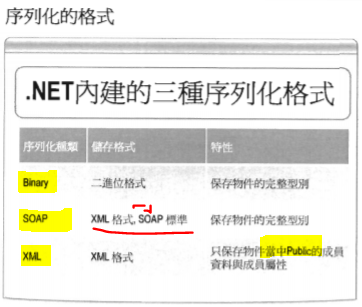
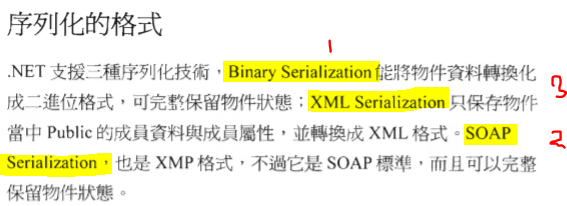
3

1 )

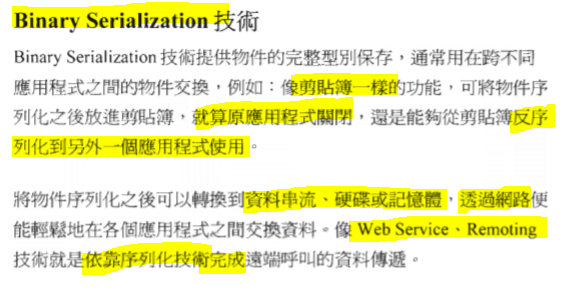




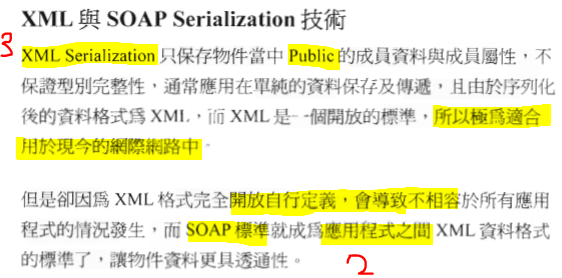


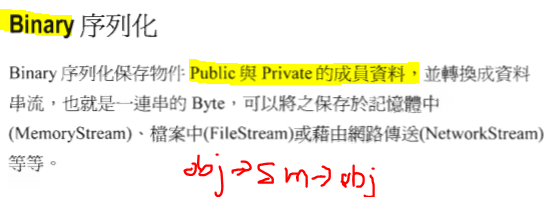
2 ) P 192

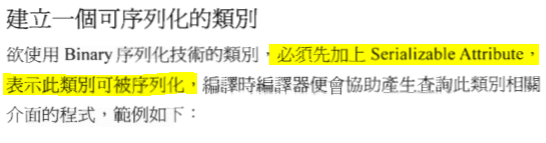
2.1 )



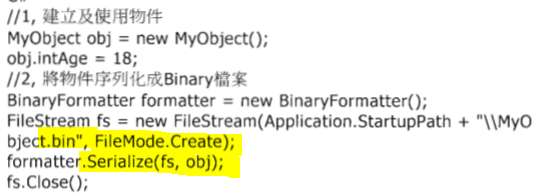
2.2 + 2.3 )

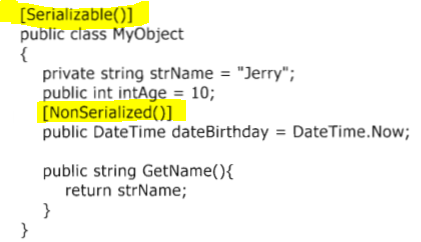


2.1 ) P 194

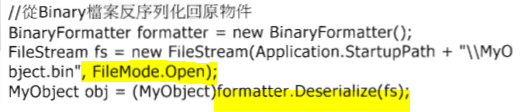


序列化步驟︰





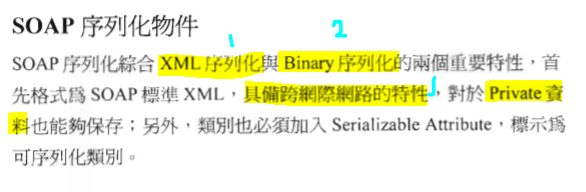
P 196 反序列化︰





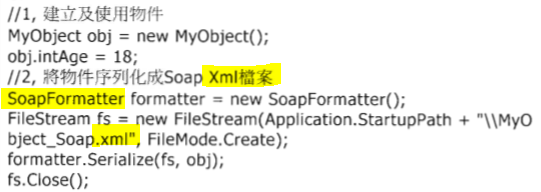
實作︰ note3\_ex1.Serialize.Binary\_serial\_ex.aspx

P 198

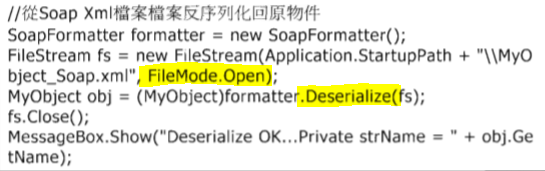




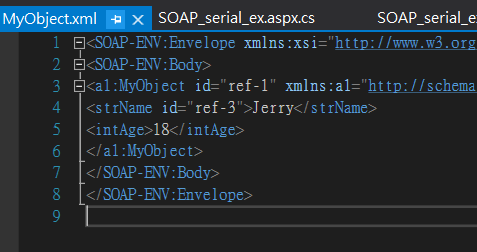
序列化︰



反序列化︰

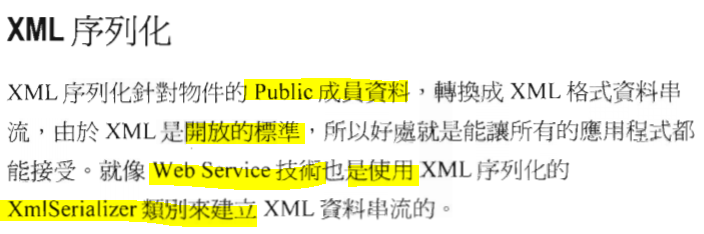


實作︰ note3\_ex1.Serialize.SOAP\_serial\_ex.aspx

 生成的xml 檔

P 201 的練習略，因為上面的那兩個實作，基本上把練習的內容都做了。

P 205



物件的屬性和方法 只能用 Pubilc Pubilc Pubilc是重點

public class MyBook {

public int BookID;

public string BookName; }

序列化︰



 // typeof(MyBook)

string curr\_path = Path.Combine(Server.MapPath("."), "MyObject2.xml");

if (File.Exists(curr\_path))

File.Delete(curr\_path);

FileStream fs = new FileStream(curr\_path, FileMode.Create);

ser.Serialize(fs, obj);

fs.Close();

反序列化︰

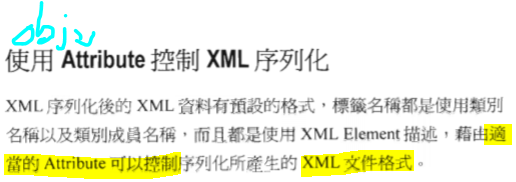
fs = new FileStream(curr\_path, FileMode.Open);

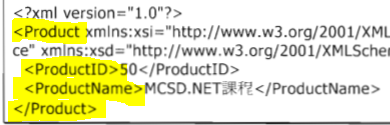
obj = (MyObject)formatter.Deserialize(fs);

fs.Close();

實作上述，無視 P 208 練習 : note3\_ex1.Serialize.XML\_serial\_ex.aspx

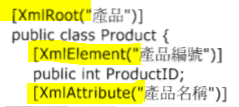
P 212



普通︰

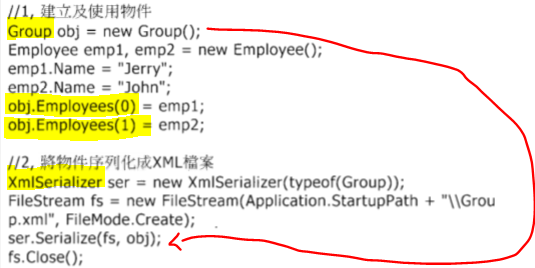


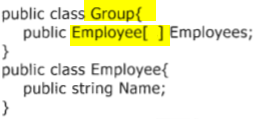
控制後︰

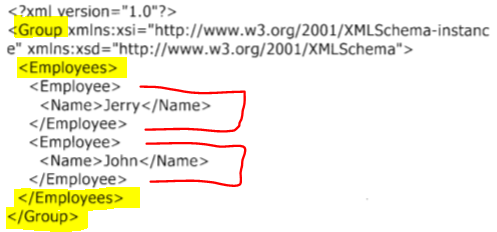




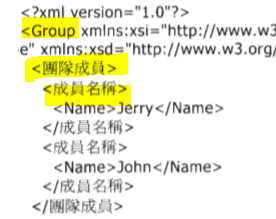


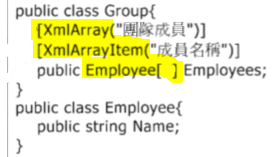
控制為物件串列 ( 不使用標簽的方法一 )︰

在實作有些出入



控制為物件串列 ( 使用標簽的方法二 ) :





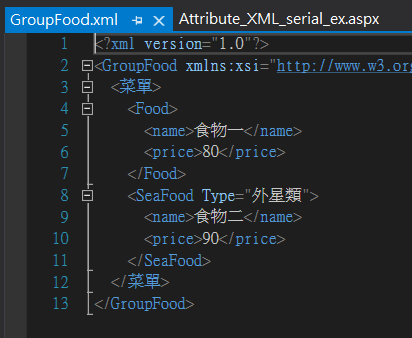
------------------------------------

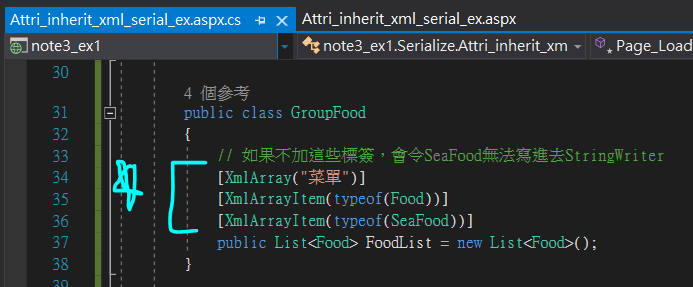
實作上面三種控制︰

note3\_ex1.Serialize.Attribute\_XML\_serial\_ex.aspx

P218 練習 ( 包括物件繼承 ) :

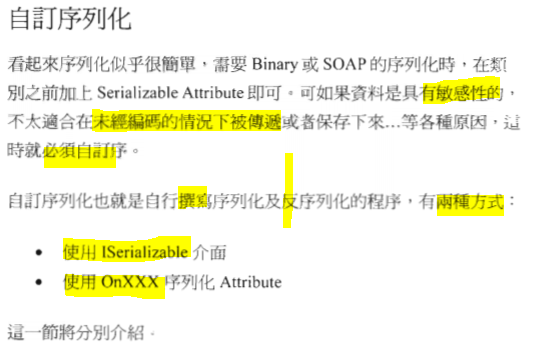
note3\_ex1.Serialize.Attri\_inherit\_xml\_serial\_ex.aspx

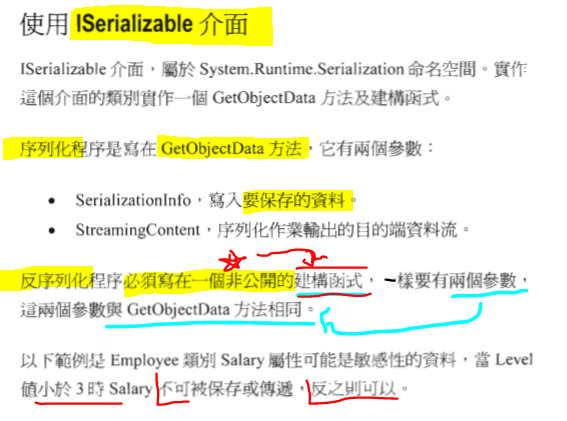
組合物件同時收納父型與子型時



--------------------------------------------------------------------------------------------------

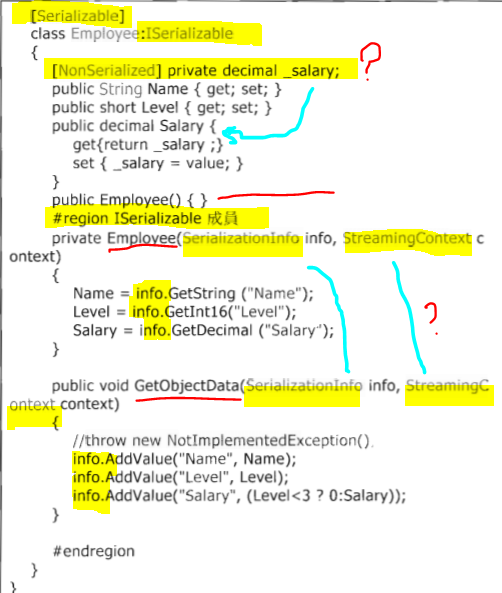
P 223





Use System.Runtime.Serialization;

P 226



注意父型的建構子是一定要被覆寫，不提。

另外一個建構子是用來序列化的。(但是建構的時候我從那裡給出參數？)不用給

GetObjectData 的方法是用來反序列化的。(同，不從從何來參數) 不用給

這樣的寫法，是為了保密 \_salary 不被寫到XML？ SOAP序列化時會判斷 < 3

Salary 得到 \_salary，所以才實作 ISerialization？ 不。

我覺得\_salary更像是個暫存，讓 Salary 本身不要有值，所以直到被序列化時才會判斷要不要抓它的值

StreamingContext 的參數又好像沒有用到？唔，不想知道他有什麼用，反正不用給。

還沒有實作？只好自己上網找補充了。

<https://docs.microsoft.com/zh-tw/dotnet/standard/serialization/custom-serialization>

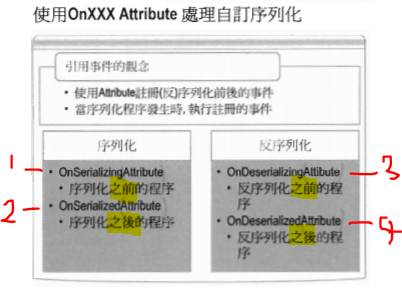
看完官網的介紹，還是不明白。但我推想，這個補充了下個實作之中XmlSerializer 不吃「那一套」的缺點。

所以我把這個練習實作於︰**ISerialization\_XML\_serial\_ex.aspx**

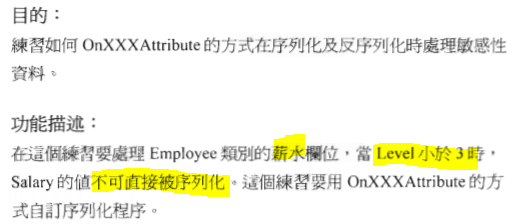
結果XmlSerializer 還是不吃這一套，不過這個方法比下面的方法簡單多了。

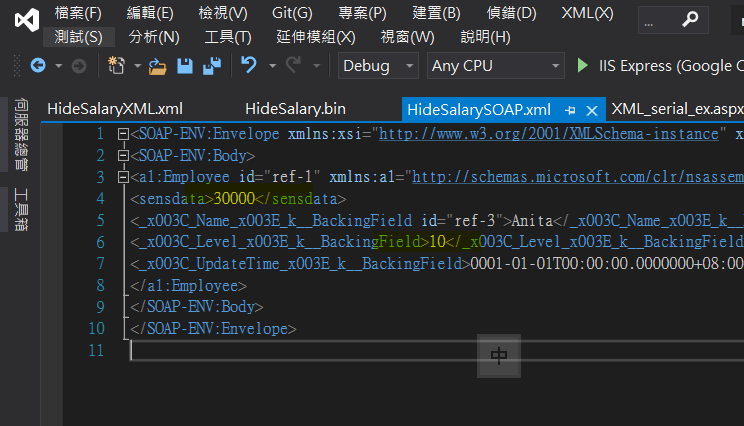
( 和下個實作同功能 )

P 227



P 228 實作 **note3\_ex1.Serialize.Hide\_info\_serial\_ex.aspx**





Level = 10

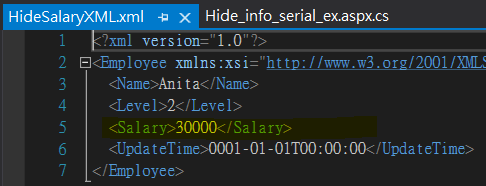
sensdata 有內容

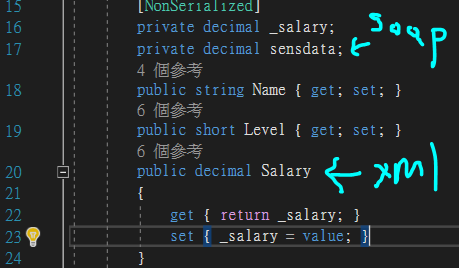
如

Level = 1

sensdata = 0

我有做三個加密和解密的版本，XmlSerializer 不吃這一套。要用SOAP。





它們序列化之後，各取不同的值。

Soap

看得見 private 的值

但是他不取\_salary因為它頭上的標簽

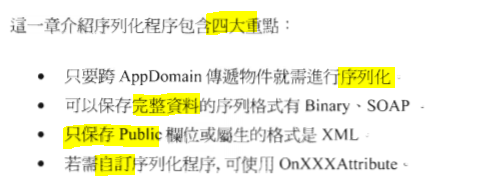
而他看見Salary，不過在序列化的時候，把值經過判斷後，寫入sensdata中。

XML

看不得見 private 的值

也沒有經過序列化時要判斷的方法。

所以永遠就直接抓了Salary 的值。



P 233 finish