

Inlämningsuppgift 2: arv, polymorfism, interface och abstrakta klasser

Inledning

I den här inlämningsuppgiften kommer du att bygga vidare på programmeringsuppgiften (uppgift 3) i inlämningsuppgift 1. Utöver den basklass och de subklasser som du redan har skapat upp i inlämningsuppgift 1 kommer du i den här uppgiften att skapa ett interface och göra förändringar i dina klasser. Därefter ska du skapa upp ett program som använder sig av dina klasser. Avslutningsvis ska du besvara frågorna i Uppgift 4. Läs instruktionerna noga!

Redovisning

Du ska redovisa inlämningsuppgift 2 skriftligen i en rapport samt skicka in en zip-fil som innehåller dina källkodsfiler och eventuella övriga filer som behövs för att kunna köra ditt program. Varje klass ska ligga i en egen fil. Rapporten ska bestå av en redovisning av dina resultat, slutsatser och reflektioner från programmeringsuppgifterna samt svar på frågorna i uppgift 4.

För att bli godkänd på inlämningsuppgift 2 ska du genomföra uppgifterna enligt givna instruktioner. Programmet/programmen ska vara körbara och fungera på ett lämpligt sätt. Vidare ska all kod vara kommenterad med XML-dokumentationskommentarer (det vill säga `///`) så att det går att få intellisense-information när man hovrar över en metod. Denna typ av kommentar skriver du ovanför en metod och kommentaren ska syfta till att beskriva vad metoden gör.

För att skriva XML-dokumentationkommentarer gör du på följande sätt:

```
///  
1 reference  
public override int GetInteger()  
{  
    ...  
    return 5;  
}
```

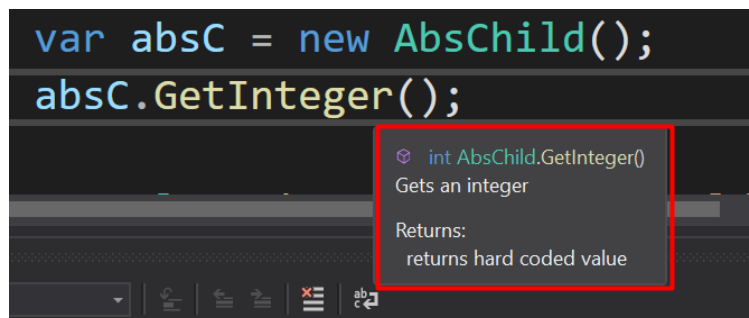
Vilket autogenererar följande:

```
/// <summary>  
///  
/// </summary>  
/// <returns></returns>  
1 reference  
public override int GetInteger()  
{  
    ...  
    return 5;  
}
```

Här ska du fylla på med lämpliga kommentarer och returvärden, till exempel:

```
/// <summary>
/// Gets an integer
/// </summary>
/// <returns>returns hard coded value</returns>
2 references
public override int GetInteger()
{
    return 5;
}
```

Vilket ger följande intellisense:



```
var absC = new AbsChild();
absC.GetInteger();
```

int AbsChild.GetInteger()
Gets an integer
Returns:
returns hard coded value

Observera att ovanstående exempel enbart är till för att illustrera hur man går till väga. Du själv ska ge en så utförlig och ackurat beskrivning som möjligt.

Du kan välja att genomföra uppgiften på svenska eller engelska. Var konsekvent med ditt språkbruk. Det är dock okej att använda engelska ord för nyckelbegrepp som exempelvis interface eller override även om svenska används som arbetsspråk. Du kan välja att skriva din rapport på till exempel svenska men att koda på engelska, huvudsaken är att du är konsekvent inom den kontext du befinner dig i.

Uppgifter som lämnats in i tid rättas inom 15 arbetsdagar. Uppgifter som lämnas in senare än deadline rättas när jag har tid.

Uppgift 1

Denna uppgift bygger vidare på den du genomförde i uppgift 3, inlämningsuppgift 1 och du kommer kunna använda delar av den i den här uppgiften. Observera att instruktionerna skiljer sig något från uppgift 3 i inlämningsuppgift 1.

Du ska skapa en basklass, Shape, för att hantera olika former. Klassen ska ha tre metoder. En för att få arean och en för omkretsen. Utöver det ska du överlagra eller göra override på ToString()¹-metoden i basklassen så att den returnerar en sträng som innehåller data om vad arean och omkretsen är.

¹ Finns i Object-klassen

Efter det ska du skapa tre subklasser, Circle, Triangle och Rectangle, som alla ärver från din basklass Shape. Dessa subklasser ska sedan göra override på metoderna från klassen Shape. Du ska ta emot alla inparametrar som behövs för att beräkna area och omkrets i konstruktorn för respektive subklass.

Uppgift 2

Du ska skapa ett interface, IShape, som säkerhetsställer att dina klasser från uppgift 1 innehåller de metoder som användaren kan förvänta sig.

De metoder som interfacet ska "tvinga" är GetArea och GetCircumference.

När du väl skapat ditt interface så ska dina subklasser från uppgift 1 implementera interfacet.

Se även till att det inte går att skapa instanser av din basklass genom att göra den abstrakt.

Uppgift 3

Skapa ett konsolprogram som tydligt visar användningen av dina klasser, genom att skapa upp minst en instans av varje subklass och visa information om den. Du visar information om den genom att helt eller delvis använda ToString-metoden. Om du vill får du även ta emot användarinput men det är inte ett krav i det här fallet. Skriv en kommentar (// eller /**/) som beskriver vad de är du demonstrerar. Till exempel:

```
/* Demonstrating method call to method on
 * the base class from instance of sub class */
Child c = new();
c.SayHello();
c.HelloChild();
```

Uppgift 4

Besvara nedanstående frågor. Korta svar går bra! Lämna in svaren i din rapport. Observera att du behöver numrera svaren så att det går att koppla en fråga till ett svar. Till exempel "3.1. Svar: bla bla bla". Du söker efter svar på frågorna genom att läsa kurslitteratur, kolla på föreläsning och genom att googla dig fram.

4.1. När ska man använda abstrakta klasser?

4.2. Vad händer när man markerar en klass med nyckelordet sealed?

4.3. Beskriv vad ett interface är.

4.4. Sant eller falskt: Kan en klass som deklarerats utan nyckelordet abstract ha abstrakta metoder?

4.5. Sant eller falskt: Kan man använda polymorfism samtidigt som man använder sig av interface och abstrakta klasser?

Avslutning

Om du kör fast på något så kan detta diskuteras under veckans Q/A-seminarium, antingen med dina medstudenter eller med mig. Skulle frågor uppstå efter seminariet går det bra att skicka ett mail och fråga mig. Ni uppmuntras även att samarbeta och ta hjälp av varandra inom klassen!

Lycka till och kör hårt!

/Elin