ממשקים – INTERFACES

ממשק הינו רעיון מופשט (ולא מחלקה), ומטרתה היא הצהרת כוונות או כוונה שעל המחלקה המיישמת לממש. הממשק מאפיין את התכונות של הרעיון, וניתן לממשו בדרכים שונות עפ"י בחירת המחלקות המבצעות. חובה על כל מחלקה מממשת לעמוד בחוזה המוגדר בממשק.

שימושים – ממשקים מאפשרים תכנות ברמת מופשטות (אבסטרקציה) גבוהה, ותכנון מוקדםשל עבודת תכנות רמת היקף וזאת ע"י הגדרת הממשקים בין רכיבים וחלוקת הפרויקט לחלקים קטנים יותר.

הממשק (או ה-INTERFACE) הינו כלי ב-C# הנועד למימוש עקרון ההפרדה בין הכרזת הממשק לבין המימוש.

המשתמש במחלקה המממשת ממשק אינו יכול לדעת את פרטי המימוש של השיטות ואפילו רצוי שלא יצטרך לחשוב עליהם כדי שיוכל להתרכז במשימה שלפניו ולטפל בה ברמת מופשטות מתאימה.

לדוגמא: כאשר רוצים לממש מערכת גדולה אשר בה כל תת-מערכת נכתבת ע"י גורם אחר. במצב כזה מגדירים הגורמים (המתכנתים) את הממשק של כל תת-מערכת, בכדי שהאינטגרציה בין תת-המערכות תתבצע ללא בעיה. במקרה הנ"ל כל תת מערכת צריכה לדעת מהו הממשק של תת-המערכות אליהן היא אמורה להתחבר בעוד שהמימוש אינו חשוב.

**הגדרה:**

הצהרה על ממשק דומה מאוד להצהרה של מחלקה, רק במקום class נשתמש ב interface.

public interface <INTERFACE\_NAME>

{

<INTERFACE\_METHODS>

}

**מספר כללי יסוד להגדרת ממשק:**

1. ממשק מגדיר פונקציות ציבוריות (public) ללא מימוש כלל.
2. שם ממשק יתחיל בדרך כלל באות I, כל מנת שיהיה קל לזהות ממשקים ולהפרידן ממחלקות.
3. הגדרת פונקציה כפרטית (private) תגרור שגיאת קומפילציה.
4. כל שיטה ללא רמת גישה בממשק היא ציבורית .בכדי לא להתבלבל ,ניתן להגדיר את השיטות תמיד 3 כציבוריות
5. כל השדות בממשק הם ציבוריות ובלתי ניתנות לשינוי גם אם לא הוגדרו כך.
6. ממשק בדומה למחלקה מגדיר טיפוס. שדות, משתנים ופרמטרים יכולים להיות מוגדרים להיות מסוג ממשק. למשל, נניח ש- IProduct הוא שם של ממשק, אזי ניתן להגדיר משתנה x להיות מסוג IProduct

IProduct x;

1. לא ניתן ליצור אוביקט (מופע) של ממשק

IProduct x = new IProduct () // Will not compile

1. מחלקה המממשת ממשק, חייבת לממש את כל הפונקציות הקיימות בו.
2. הצהרה של class העושה שימוש בממשק יהיה כך:

public class *CLASS\_NAME*: *INTERFACE\_NAME*

1. כאמור, על המחלקה לממש את כל הפונקציות בממשק. למרות האמור לעיל, ניתן ליצור מצב בו מחלקה לא מממשת את כל הפונקציות בממשק רק אם המחלקה היא אבסטרקטית, כאשר הפונקציות החסרות יוגדרו כפוונקציות אבסטרקטיות. במצב זה, המחלקות שירשו ממחלקה זו יצטרכו לממשק את הממשק.
2. מחלקה יכולה לממש יותר מממשק אחד:

public class *CLASS\_NAME*: *INTERFACE\_NAME*1, *INTERFACE\_NAME*2, *INTERFACE\_NAME*3

{

<INTERFACE\_NAME1\_METHODS>

<INTERFACE\_NAME2\_METHODS>

<INTERFACE\_NAME3\_METHODS>

}

**דוגמא**:

using System;

public interface IProduct

{

string GetManufacturer();

string GetModel();

double GetPrice();

}

public class LcdScreen: IProduct

{

private string \_manufacturer;

private string \_model;

private double \_price;

private int \_sizeInInches;

public LcdScreen(string manufacturer, string model, double price, int sizeInInches)

{

\_manufacturer = manufacturer;

\_model = model;

\_price = price;

\_sizeInInches = sizeInInches;

}

public string GetManufacturer()

{

return \_manufacturer;

}

public string GetModel()

{

return \_model;

}

public double GetPrice()

{

return \_price;

}

public int GetSize()

{

return \_sizeInInches;

}

}

public class MobilePhone : IProduct

{

private string \_manufacturer;

private string \_model;

private double \_price;

private int \_storageSize;

private int \_numOfCores;

public MobilePhone(string manufacturer, string model, double price, int storageSize, int numOfCores)

{

\_manufacturer = manufacturer;

\_model = model;

\_price = price;

\_storageSize = storageSize;

\_numOfCores = numOfCores;

}

public string GetManufacturer()

{

return \_manufacturer;

}

public string GetModel()

{

return \_model;

}

public double GetPrice()

{

return \_price;

}

public int GetStorageSize()

{

return storageSize;

}

public int GetNumOfCores()

{

return \_numOfCores;

}

}

public class Program

{

public static void Main()

{

IProduct products = new IProduct[4];

products[0] = new MobilePhone("Sumsung", "Galaxy S9", 3000, 128, 8);

products[1] = new MobilePhone("Apple", "iPhone Xs", 4000, 128, 8);

products[2] = new LcdScreen("Sony", "XVDS", 3000, 60);

products[3] = new LcdScreen ("Haier", "FSx^)", 2000, 60);

for (int i = 0; i < products.Length; i++)

{

Console.WriteLine(products[i].GetManufacturer() + " " + products[i].GetModel() + " " + products[i].GetPrice() + " NIS");

}

}

}

בדוגמא ניתן לראות כי יצרנו שתי מחלקות – MobilePhone ו- LcdScreen אשר שתיהן מממשות את הממשק IProduct.

ב-Main ניתן לראות כי אנחנו מתייחסים לשניהם באותה צורה, בתוך מערך המוגדר עפ"י הממשק. זה אומר, כי ניתן להתייחס לכל המחלקות אשר מממשות ממשק אחיד כאל הממשק בלבד, ובאופן כזה ניתן לגשת לממשקים המוגדרים בממשק המאפיין את המערך.