### תיאור קצר של הפיצ'רים שבחרנו לממש בתרגיל הקודם:

* **“Filtered NewsFeed”** (נמצא בקוד בתוך התיקיה “Filtered News Feed”) – האפליקצייה מציגה את רשימת החברים של המשתמש. האפליקציה מאפשרת למשתמש להוסיף חברים מרשימה, לקבוצות חברים קטנות, אחריהם (בלבד) מבקש לעקוב. באותו אופן, מאפשרת גם להסיר חבר/ה מקבוצת המעקב. עבור כל חבר/ה בקבוצת המעקב האפליקציה מציגה: פוסטים, תמונות, אירועים, דפים שסומנו בלייק, תאריכי ימי הולדת ומקומות שסומנו כChecked-in. בלחיצה על שם אירוע מסויים, האפליקציה מציגה מי מבין החברים בקבוצת המעקב סימן שמגיע לאירוע. כמו כן, מאפשרת לשלוח מייל עדכון לחברי הרשימה שסימנו 'הגעה' לאירוע באם המשתמש מגיע גם הוא. בנוסף, האפליקציה מאפשרת לשמור כקובץ את תאריכי ימי ההולדת של חברי קבוצת המעקב.
* **“Zodiac Matching”** (נמצא בקוד בתוך התיקיה “Zodiac Matching”) - האפליקציה מציגה את כל הדפים שהמשתמש סימן בלייק. האפליקציה מאפשרת למשתמש לבחור Page מהרשימה ועבורו מציגה רשימת משתמשים שסימנו גם הם בלייק את אותו הדף. האפליקצייה מאפשרת לסנן את רשימת המשתמשים הנ"ל ע"פ הגדרה. לדוגמא: מין, מזל, במערכת יחסים/רווק. האפליקצייה מאפשרת למשתמש לבחור חבר/ה מרשימת החברים ועבורו/עבורה מציגה הורוסקופ המתמקד בפן הזוגי ואת אחוז ההתאמה שמחושב ע"פ מזלות המשתמשים. האפליקציה מאפשרת לפרסם את תוצאת ההתאמה על Wall המשתמש בפייסבוק.

### תבנית מס' 1 – [Template Method]

* סיבת הבחירה / שימוש בתבנית:

בחרנו להשתמש בתבנית זו בפיצ'ר Zodiac Matching,

בפיצ'ר הנ"ל קיימת מחלקה אבסטרקטית בשם ComboBoxFilter המורכבת משתי מתודות : insert וcreate, כאשר מימוש המתודה insert רלוונטית לכל האובייקטים הקונקרטים היורשים מהמחלקה האבסטרקטית הנ"ל והמתודה Create היא overridable כאשר כל אחד מהאובייקטים הקונקרטים היורשים מהמחלקה הנ"ל צריכים לממש באופן שונה בהתאם לצרכיהם האישיים.

(לכן המתודה Create תהווה את ה"injection point").

כך בעצם מימשנו מנגנון כללי באופן קונקרטי בעזרת הורשה.

* אופן המימוש:

יצרנו מחלקת בסיס אבסטרקטית (ComboBoxFilter) בעלת 2 מתודות : insert וCreate כאשר insert היא מתודה קבועה עבור כל המחלקות היורשות מהמחלקה הנ"ל ומתודה אבסטרקטית, Create, אותה כל אחת מהמחלקות היורשות מממשת באופן שונה/ קונקרטי לצרכיה.

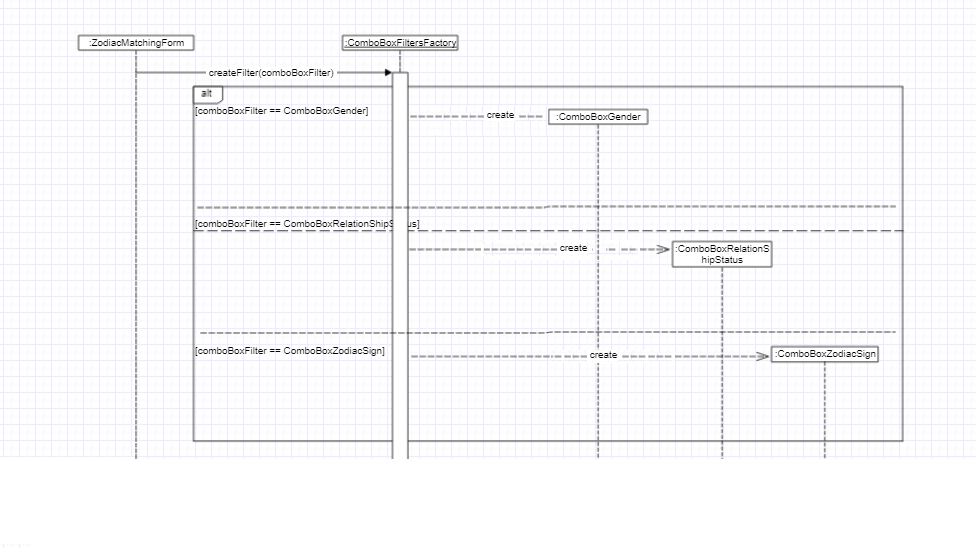
**את הקוד ניתן למצוא תחת התיקיה Zodiac Matching.**

**המחלקות הרלוונטיות לתבנית עיצוב הנ"ל הן :**

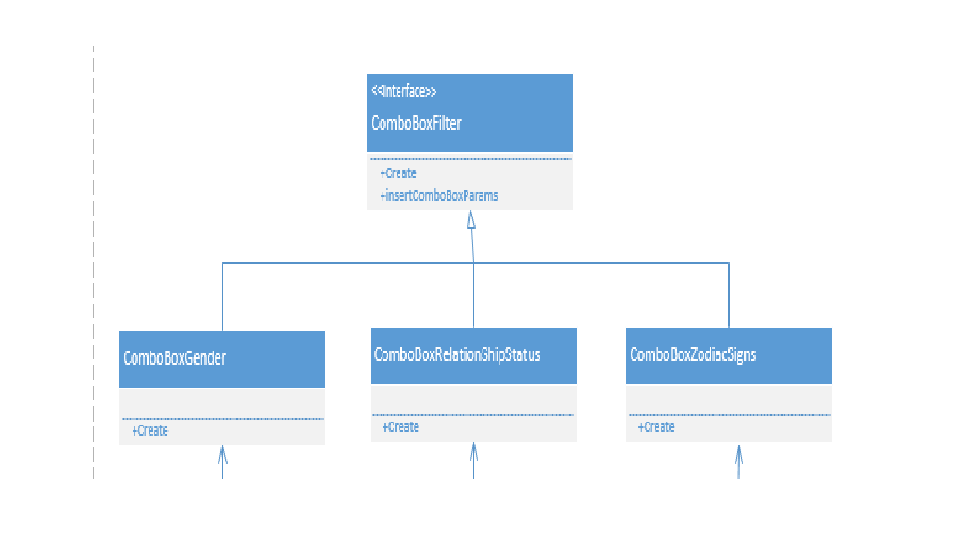
**ComboBoxFilters, ComboBoxGender, ComboBoxRelationShipStatus,**

**ComboBoxZodiacSign.**

* Sequence Diagram



* Class Diagram



מחלקות מקבילות למחלקות בתבנית עיצוב template method :

Abstract class : **ComboBoxFilter**

Concrete class 1 : **ComboBoxGender**

Concrete class 2 : **ComboBoxRelationShipStatus**

Concrete class 3 : **ComboBoxZodiacSigns**

### תבנית מס' 2 – [Strategy]

* סיבת הבחירה / שימוש בתבנית:

הוספנו למערכת בפיצ'ר Zodiac Matching את האפשרות למיין את רשימת הדפים שהמשתמש אהב בסדר עולה/יורד לפי ערך לקסיקורגרפי.

אלגוריתם המיון בסדר עולה/יורד בנוי מאותה שלדה של אלגוריתם עם שינוי קל באלגוריתם בנקודה מסוימת על מנת לקבל את 2 התוצאות הרצויות (סדר עולה/יורד).

לכן על מנת להמנע משכפול קוד של האלגוריתם כולו, מצאנו לנכון להשתמש בתבנית עיצוב Strategy אשר עוזרת לנו להמנע ממצב כזה.

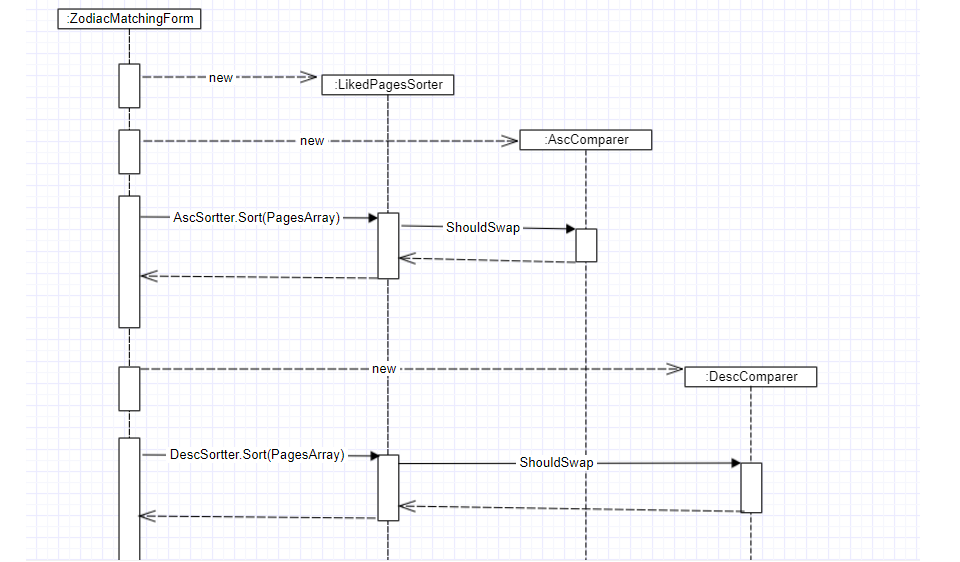
* אופן המימוש:

הגדרנו מחלקה חדשה בשם LikedPagesSorter המכילה את "שלדת" האלגוריתם, ומחזיקה רפרנס לאובייקט מסוג IComparer המהווה את השוני בין האלגוריתמים.

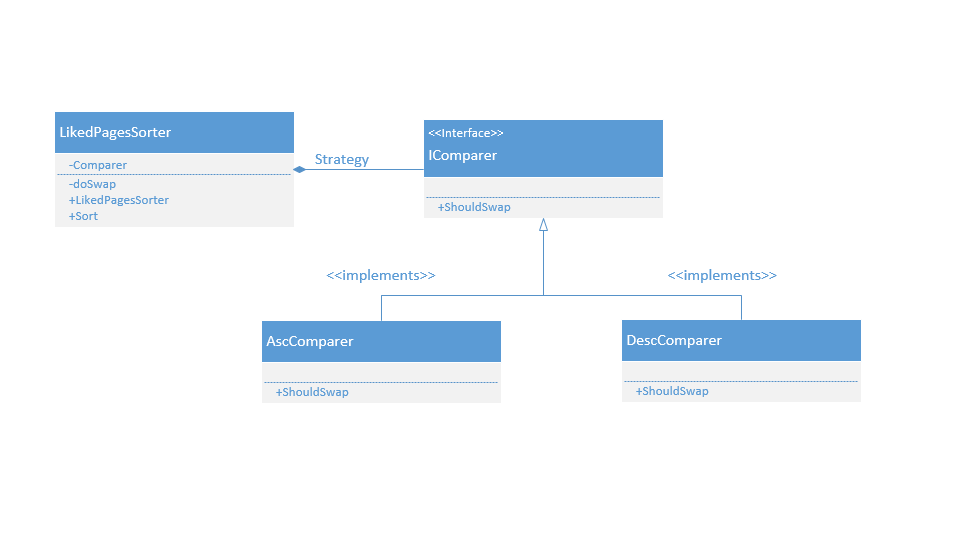
הגדרנו מחלקה IComparer המצהיר על המתודה ShouldSwap שאותה כל אחד מהמחלקות היורשות ממנה מיישם באופן קונקרטי בהתאם לרצונו, על מנת לשנות את אופי האלגוריתם – בדוגמא שלנו - הצגת הדפים בסדר עולה/יורד לקסיקוגרפית (AscComparer וDescComparer יורשים מIComparer ומממשים את המתודה המוצהרת בה באופן קונקרטי).

ניתן למצוא את התבנית בקוד תחת התיקיה **ZodiacMatchingForm** במחלקות : **LikedPagesSorter, IComparer, AscComparer, DescComparer.**

* Sequence Diagram



* Class Diagram

מחלקות מקבילות למחלקות בתבנית עיצוב template method :

Context : **LikedPagesSorter**

Strategy : **IComparer**

ConcreteStrategyA : **AscComparer**

ConcreteStrategyB : **DescComparer**

### תבנית מס' 3 – [Obsrever]

* סיבת הבחירה / שימוש בתבנית:

בחרנו להשתמש בתבנית זו בפיצ'ר הFiltered News-Feed. הסיבה לכך היא:

במערכת הנ"ל קיימת תלות בין כפתור השידוך (AttachButton), לבין החלון הראשי (FilteredNewsFeedForm) וקבוצת המעקב (FeedGroup). בלחיצה על הכפתור, בעצם נשייך חבר מרשימת החברים בחלון הראשי לרשימת החברים בקבוצת המעקב.

במערכת היחסים הזאת אנו רוצים ליצור מצב שבעת נשנה את מצבו של הכפתור, נניח בלחיצה עליו, יעודכנו גם מצבם של הישויות הקשורות בו כפי שציינו לעיל.

תבנית Observer מגדירה תלות של אחד לרבים בין אובייקטים, כך שכאשר אובייקט אחד משנה את מצבו, כל התלויים בו מיודעים על השינוי ומתעדכנים אוטומטית ולכן התאימה לצרכינו.

* אופן המימוש:

ניתן למצוא את מימוש התבנית תחת התיקיה Filtered News Feed.

Subject/Notifyer: AttachButton, בעת לחיצה על הכפתור "יעוררו" כל המתודות אשר נרשמו כlisteners.

Observer: FilterefNewsFeedForm, FeedGroup, מצבם ישתנה ויושפע מלחיצה על כפתור הAttachButton.

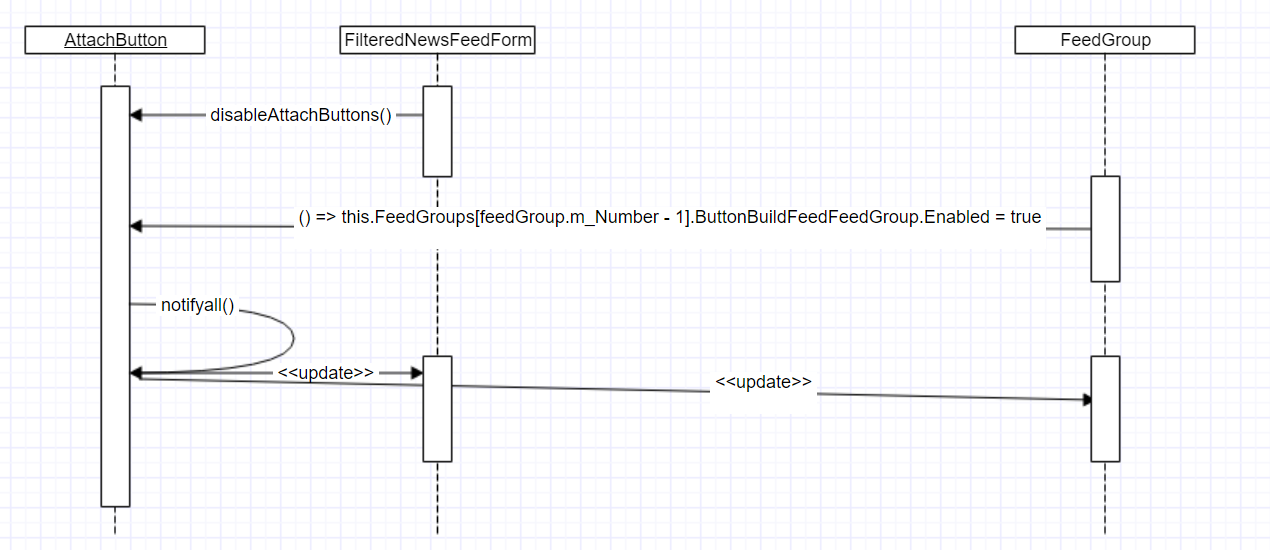
במחלקה AttachButton מימשנו Delegate,

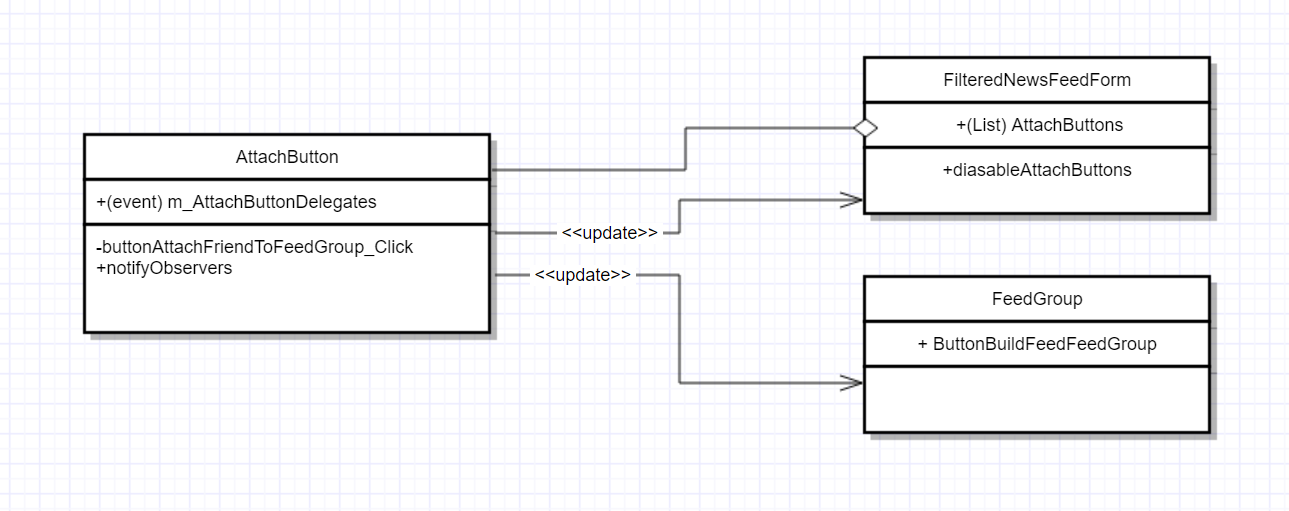
public event Action m\_AttachButtonDelegates;

בעת לחיצה על כפתור הAttachButton נפעיל את המתודה notifyObservers(), שתבצע Invoke לכל המתודות אשר "נרשמו" ע"י אובייקטים שמייצגים את הObservers ע"י מתן refrence למתודות שלהם:

1. attachButton.m\_AttachButtonDelegates += this.diasableAttachButtons;
2. this.AttachButtons[feedGroup.m\_Number - 1].m\_AttachButtonDelegates += (() => this.FeedGroups[feedGroup.m\_Number - 1].ButtonBuildFeedFeedGroup.Enabled = true);

* Sequence Diagram



* Class Diagram