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

* **ברכות יום הולדת מותאמות אישית לפי הגיל והמין של המשתמש** – הפיצ'ר עובר על רשימת החברים של המתשמש ובונה רשימה של החברים החוגגים יום הולדת, לאחר מכן עובר על הרשימה שנוצרה ובהתאם למין של המשתמש בודק האם קיימת ברכה מיוחדת לגילו (לדוגמה – עבור נקבה בת 12 קיימת ברכת בת מצווה) ומפרסם אותה בקיר שלו, במידה ולא קיימת ברכה מיוחדת מפרסם ברכת יום הולדת סטנדרטית.
* **סינון התמונות בעלות מספר התגובות הגבוהה ביותר עבור קהל יעד ספציפי** - הפיצ'ר עובר על רשימת התמונות ויוצר רשימת תמונות ממויינת על פי מספר התגובות לתמונה שהתקבלו ממשתמשים העומדים בדרישות הסינון של המשתמש (גיל, מין וסטטוס זוגי) ומחזיר את התמונות הראשונות (בעלות מספר התגובות הגבוה ביותר שהתקבלו מקהל היעד שהמתמש הגדיר), כמות התמונות שתוחזר גם היא מוגדרת על פי דרישת המשתמש (1, 3 או 5 תמונות).

### תבנית מס' 1 – Singleton

* סיבת הבחירה / שימוש בתבנית:

במערכת שלנו קיימת מחלקה (Wrapper) העוטפת את השירותים של המחלקות FacebookService ו FacebookWrapper ומאפשרת להשתמש בהם בצורה בטוחה (על ידי טיפול ב Exceptions).

רצינו לוודא שיהיה מופע אחד ויחיד עבור המחלקה Wrapper על מנת לוודא אחידות המידע ונכונות המידע מכיוון שהמחלקה מחזיקה reference למשתמש אשר מגדיר בתוכו גם את ה s\_CollectionLimit שמשפיע באופן ישיר על פעולת המתודות שבחלקה Wrapper והתוצאות שהן מחזירות.

בנוסף במידה ונוסיף בעתיד טפסים נוספים אשר יעשו שימוש בלוגיקה חדשה יהיה צורך להשתמש באותו המופע של המחלקה Wrapper שיצרנו על מנת להבטיח אחידות המידע כלומר במידה ויצטרפו רכיבים נוספים למערכת יהיה צורך בקבלת reference למופע של המחלקה Wrapper ונרצה להבטיח שהם יוכלו לקבל אותו ביוזמתם וללא שינוי של המערכת.

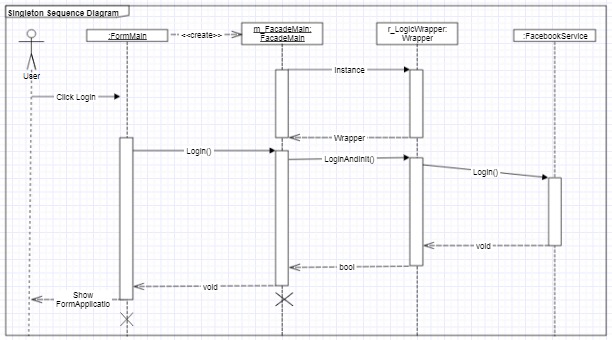
* אופן המימוש:

יצרנו Properties סטטי בשם Instace במחלקה Wrapper אשר משתמשת במנגנון נעילה כפול על מנת להחזיר את המופע היחיד של המחלקה במידה וקיים או ליצור מופע כזה במידה ולא קיים.

בכל מקום שבו יש צורך בשימוש בשירותים של המחלקה Wrapper (לדוגמה במחלקות – FacadeApplication, FacadeMain, MostCommentablePhotosLogic) ניתן לראות שבמקום ליצור מופע של המחלקה Wrapper השתמשנו ב Properties הסטטי Instace על מנת לקבל את המופע שלה או ליצור אותו במידת הצורך.

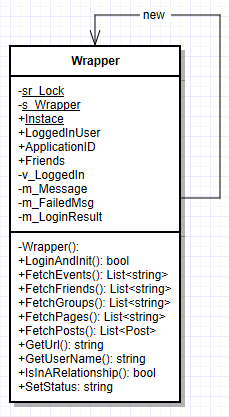
הגדרנו את הקונסטרקטור של המחלקה Wrapper כprivate כדי שלא תהיה אפשרות ליצור מופעים שלה בדרך אחרת.

* Sequence Diagram



* Class Diagram

|  |  |
| --- | --- |
| Our Implementation | Pattern |
| Wrapper | Singleton |



### תבנית מס' 2 – Factory Method

* סיבת הבחירה / שימוש בתבנית:

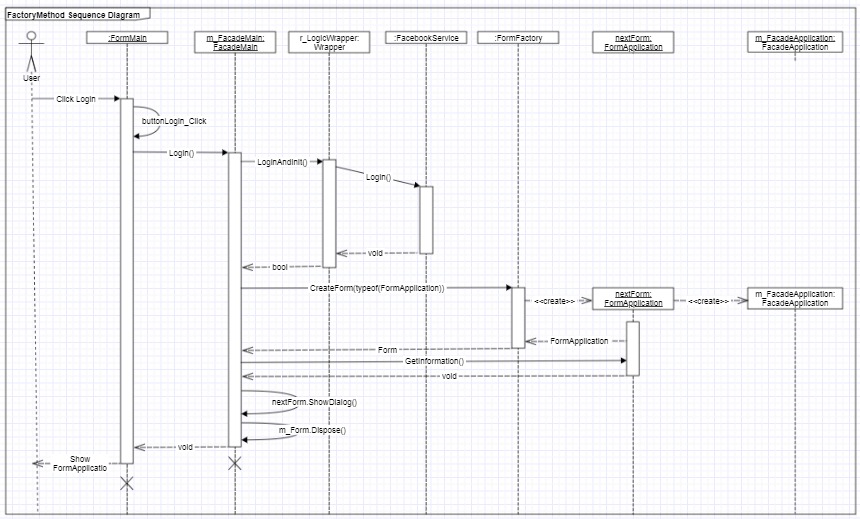
כל הטפסים באפליקציה שלנו שייכים לאותה משפחה פולימורפית Form)), לכן אם נרצה בעתיד להוסיף טופס נוסף או לשנות את הדרך שבה נוצר טופס קיים נרצה לעשות זאת מבלי לשנות דבר בקוד הקיים מלבד השינוי/ הוספה בקוד היצירה של הטופס.

* אופן המימוש:

מימשנו את התבנית במחלקה סטטית חדשה הנקראת FormFactory שבה יש מתודה סטטית הנקראת CreateForm, המתודה מקבלת את סוג טופס הנדרש ומייצרת טופס חדש מאותו סוג.

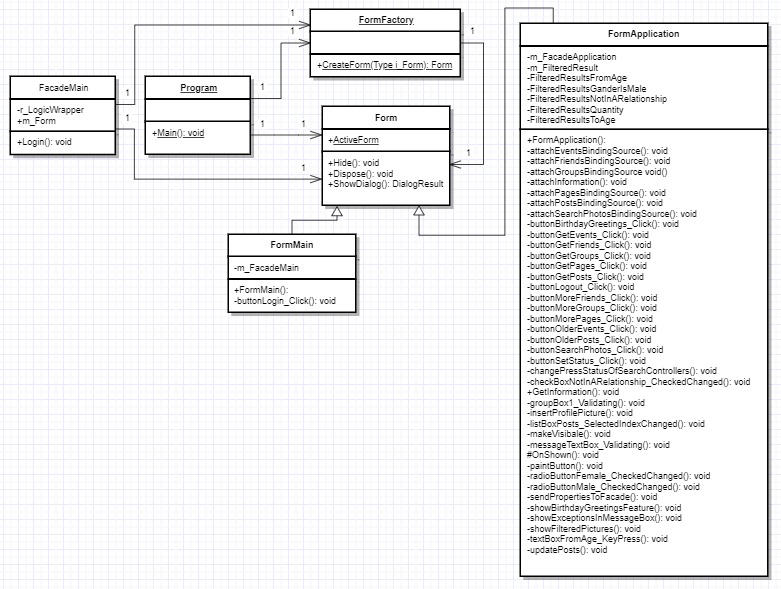
השימוש במחלקה נעשה בכל מקום שבו יש דרישה ליצירת טופס חדש – אצלנו זה קורה במחלקה Program ובמחלקה FacadeMain.

* Sequence Diagram



* Class Diagram

|  |  |
| --- | --- |
| Our Implementation | Pattern |
| FormFactory | StaticFactoryClass |
| * Program * FacadeMain | Client |
| Form | AbstractProduct |
| * FormMain * FormApplication | ConcreteProduct |



### תבנית מס' 3 – Facade

* סיבת הבחירה / שימוש בתבנית:

בחרנו בתבנית זו על מנת להסתיר מפני הצרכן את המורכבות של תתי המערכות ובעזרת פעולה פשוטה של פניה לפסד תתבצע כל המורכבות הנ"ל.

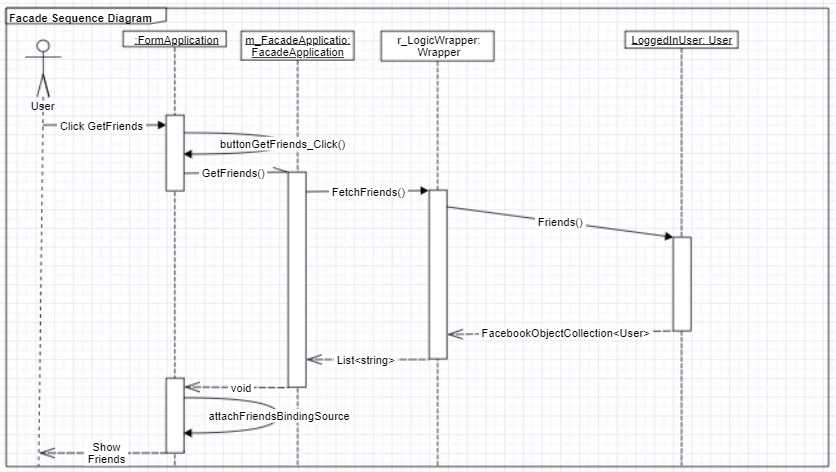
אי לכך ובהתאם לזאת לפסד יתרונות רבים כגון נוחות שימוש, בטיחות שימוש, נוחות תחזוקה, שימוש חוזר.

הפסדים שלנו מתווכים בין הטפסים (UI) ללוגיקה שמאחוריהם ע"י הסתרה של מורכבות הלוגיקה ופישוט הפניה על ידי ה- UI.

* אופן המימוש:

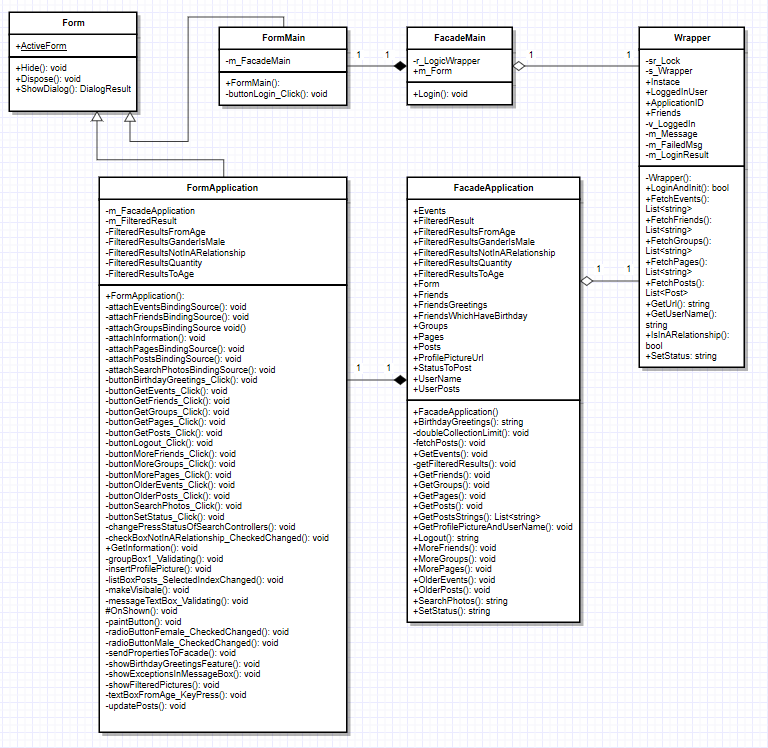
יצרנו שני פסדים (אחד לכל טופס) הנקראים FacadeApplication ו- FacadeMain המאגדים בתוכם את הלוגיקה שבה נעשה שימוש בכל טופס (הצרכים הפונקציונליים של כל טופס).

* Sequence Diagram



* Class Diagram

|  |  |
| --- | --- |
| Our Implementation | Pattern |
| * FormMain * FormApplication | Client |
| * FacadeMain * FacadeApplication | Facade |
| Wrapper | SubSystem |



### Threads:

בכל מקום שבו בחרנו להשתמש בעבודה אסינכורנית ע"י thread אסינכרוני נפרד (כמפורט בהמשך) עשינו זאת על מנת שהמשתמש יהנה מחוויית שימוש רציפה ולא יצטרך להמתין לקבלת המידע ועיבודו (בין אם מהשרת ובין אם ממחלקה פנימית שמימושה עלול להשתנות בעתיד) - תהליכים שידוע שעלולים לקחת זמן רב יחסית ולגרום לתחושה שהאפליקציה נתקעה ובכך לפגוע בחווית השימוש של המשתמש ולכן העדפנו לדמות מצב שבו פעולות אלו "רצות ברקע".

השימוש בעבודה אסינכורנית ע"י thread אסינכרוני נפרד נעשה במקומות הבאים:

1. במחלקה FormApplication

* בשורה 124 במתודה attachInformation - על מנת לקבל לקבל את תמונה הפרופיל ושם המשתמש מהשרת.
* בשורה 187 במתודה buttonSearchPhotos\_Click - על מנת לקבל לקבל את התמונות לאחר סינון על פי בחירות המתשמש מהפסד (שכידוע המימוש בו עלול להשתנות).
* בשורה 267 במתודה updatePosts - על מנת לקבל לקבל את רשימת הפוסטים מהשרת.
* בשורה 284 במתודה buttonOlderPosts\_Click - על מנת לקבל לקבל את רשימת הפוסטים מהשרת.
* בשורה 294 במתודה buttonGetEvents\_Click - על מנת לקבל לקבל את רשימת האירועים מהשרת.
* בשורה 310 במתודה buttonOlderEvents\_Click - על מנת לקבל לקבל את רשימת האירועים מהשרת.
* בשורה 320 במתודה buttonGetPages\_Click - על מנת לקבל לקבל את רשימת הדפים מהשרת.
* בשורה 336 במתודה buttonMorePages\_Click - על מנת לקבל לקבל את רשימת הדפים מהשרת.
* בשורה 346 במתודה buttonGetGroups\_Click - על מנת לקבל לקבל את רשימת הקבוצות מהשרת.
* בשורה 362 במתודה buttonMoreGroups\_Click - על מנת לקבל לקבל את רשימת הקבוצות מהשרת.
* בשורה 372 במתודה buttonGetFriends\_Click - על מנת לקבל לקבל את רשימת החברים מהשרת.
* בשורה 388 במתודה buttonMoreFriends\_Click - על מנת לקבל לקבל את רשימת החברים מהשרת.

1. במחלקה FacadeApplication

* בשורה 41 ב C'tor - על מנת לקבל לקבל את תמונה הפרופיל ושם המשתמש מהשרת.
* בשורה 230 במתודה SearchPhotos - על מנת לקבל לקבל את התמונות לאחר סינון על פי בחירות המתשמש מהשרת.

### Data Binding:

1. במחלקה FormApplication נעשה שימוש ב DataBinding

* בשורה 132 על מנת לקשר את ה FacadeApplication לטופס.
* בשורה 138 על מנת לקשר את ה Property של ה FriendsGreeing מה FacadeApplication לטופס.
* בשורה 139 על מנת לקשר את ה Property של ה FriendsWhichHaveBirthday מה FacadeApplication לטופס.
* בשורה 140 על מנת לקשר את ה Property של ה FilteredResult מה FacadeApplication לטופס.
* בשורה 277 על מנת לקשר את ה Property של ה Posts מה FacadeApplication לטופס.
* בשורה 278 על מנת לקשר את ה Property של ה UserPosts מה FacadeApplication לטופס.
* בשורה 304 על מנת לקשר את ה Property של ה Events מה FacadeApplication לטופס.
* בשורה 330 על מנת לקשר את ה Property של ה Pages מה FacadeApplication לטופס.
* בשורה 356 על מנת לקשר את ה Property של ה Groups מה FacadeApplication לטופס.
* בשורה 382 על מנת לקשר את ה Property של ה Friends מה FacadeApplication לטופס.
* בשורות 442 ו 448 על מנת לקשר את ה Post הנוכחי (עליו אנחנו מצביעים ברשימת הפוסטים) מה Property של ה UserPosts מה FacadeApplication לטופס באופן דו כיווני (**2 Way Data Binding**) על מנת שנוכל לשנות את הפוסט עצמו דרך הטופס, כמובן שהשינוי מתבצע רק במידע השמור באפליקציה ולא במידע שקיים בשרתי Facebook שכן הם חוסמים את האפשרות לערוך את הפוסטים מהאפליקציה שלנו אבל רצינו בכל זאת להדגים שימוש באפשרות של קישור דו כיווני.