### תבנית מס' 1 – [Strategy]

* סיבת הבחירה / שימוש בתבנית:

בחרנו לממש את Strategy behavioral pattern כיוון שהוא מספק פלקסביליות ואקסטנסביליות: ה Strategy מאפשר להגדיר משפחה של אלגוריתמים או התנהגויות ומאפשר להם להיות ברי שינוי. במצב שלנו ישנם מס' אסטרטגיות לשינוי הכפתור צבע/ מצב.  
השימוש ב Strategy behavioral pattern מתבצע כאשר המשתמש מתחבר לפייסבוק דרך ה DesignForm , מתבצע שימוש ב EnableStrategy על מנת לאפשר לחיצה על הכפתורים הקיימים, בנוסף בעת לחיצה על הכפתורים Posts , Albums ו Groups מתבצע שימוש ב Strategy pattern כאשר ה thread טוען את הפוסטים/ אלבומים הקיימים הכפתור משתנה לצבע AliceBlue וכאשר מסתיימת הפעולה הכפתור משנה צבעו חזרה לצבע המקורי.

* אופן המימוש:

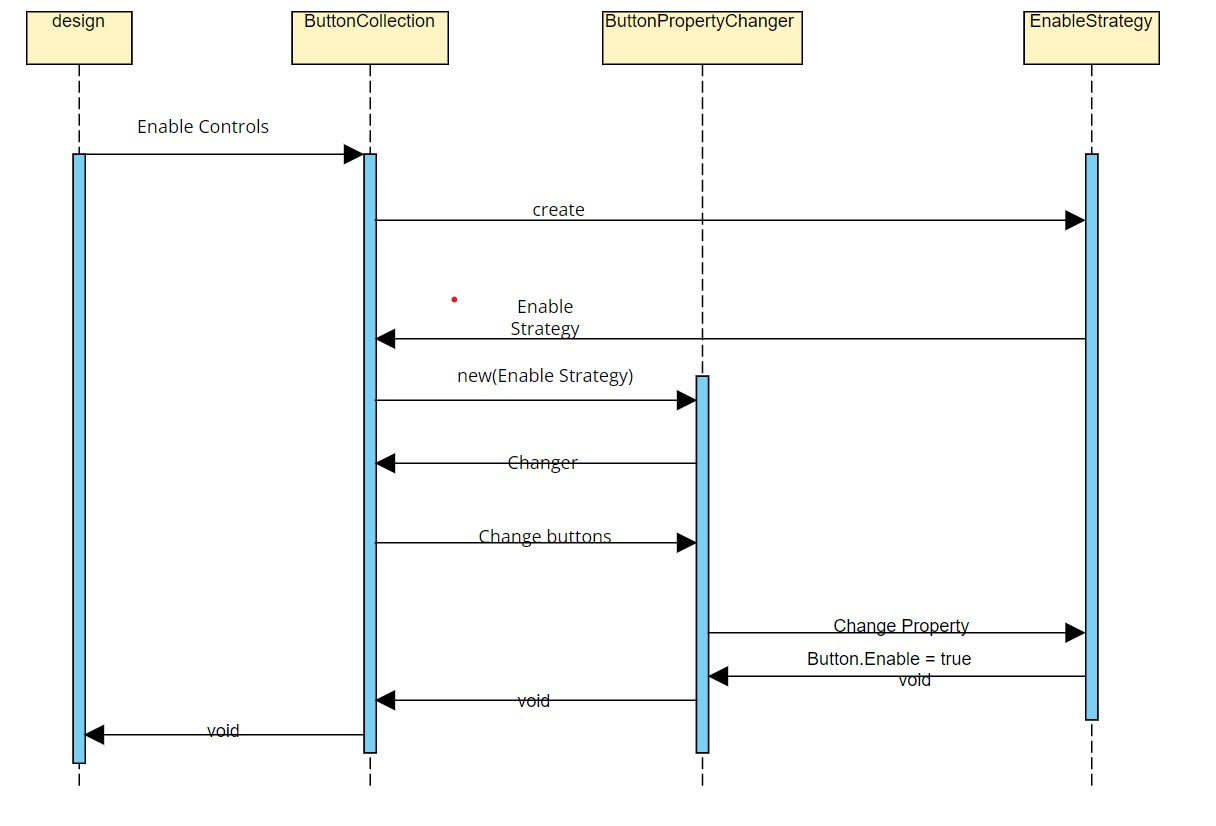
1. IChangeButtonStrategy Interface :   
   זהו ממשק המגדיר את החוזה עבור השינוי של הכפתורים.
2. ButtonPropertyChanger class :  
   המחלקה הזו מתפקדת כמעטפת של ה Strategy Pattern . היא מחזיקה רפרנס ל Strategy ספציפי ומאפשר לשנות את התכונה של ה Strategy הספציפי.
3. ButtonCollection :  
   מחלקה זו מרחיבה את מחלקת Button מבלי לשנות אותה ומאפשר לממש את השינויים הרצויים עבור הקוד שלנו (צבע, enable/disable)
4. ClickedColorStrategy class :   
   מחלקה זו משנה את צבע הרקע של הכפתור לצבע AliceBlue כאשר הכפתור נלחץ.
5. UnclickedColorStrategy class :  
   מחלקה זו משנה את צבע הרקע של הכפתור חזרה לצבעו המקורי.
6. EnableStrategy class :  
   מחלקה זו אחראית על שינוי המצב של הכפתור ממצב disable ל enable.

השחקנים – IchangeButtonStrategy – מייצג את ה Strategy Interface .

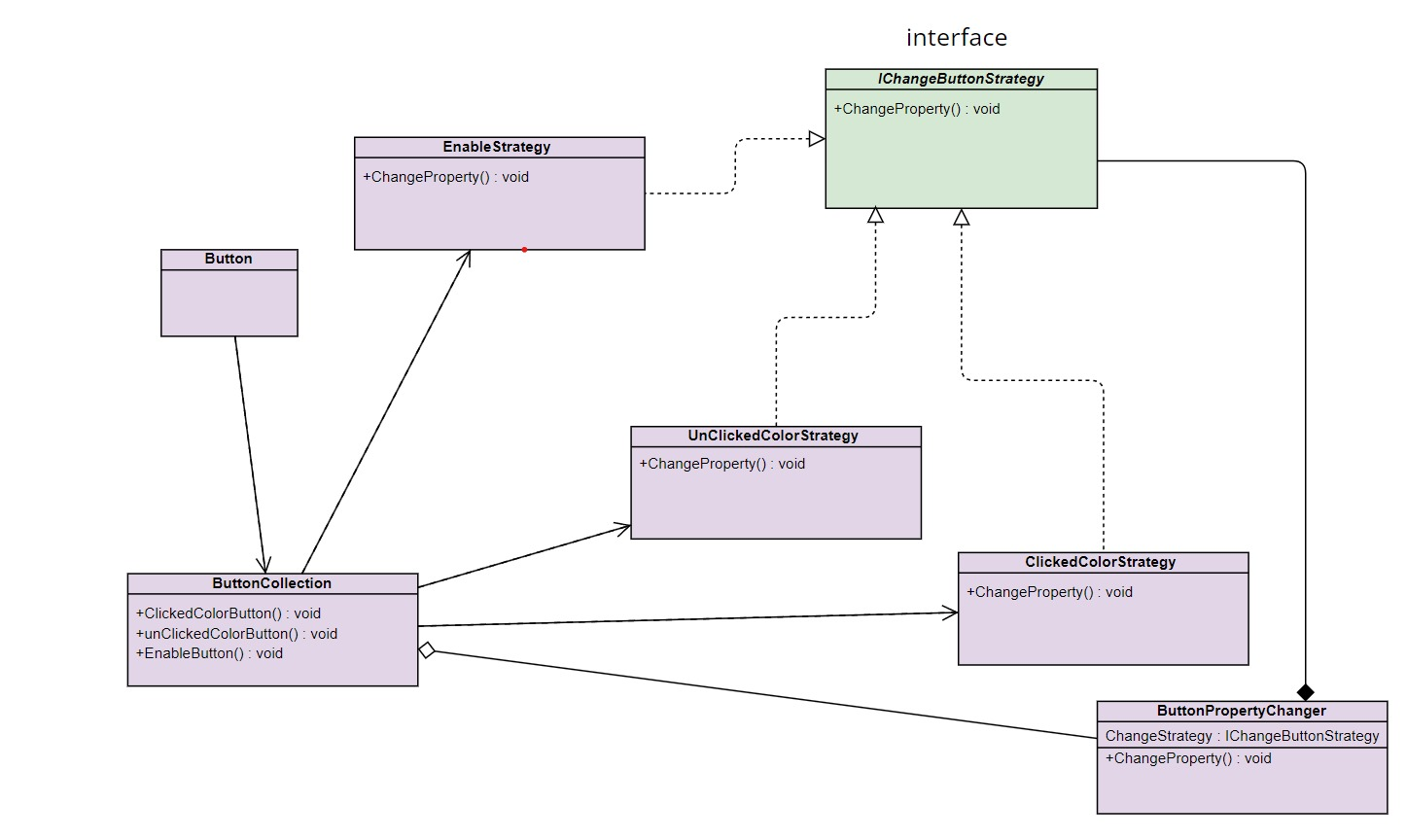
ButtonPropertyChanger class – מחזיקה רפרנס ל strategy הספציפי

ClickedColorStrategy class, UnclickColorStrategyClass , EnableStrategy class – מייצגים כל אחד את האסטרטגיה הספציפית שלו לשינוי הכפתור.

* Sequence Diagram



* Class Diagram



### תבנית מס' 2 – [Iterator]

* סיבת הבחירה / שימוש בתבנית:

בחרנו לממש את ה Iterator Pattern כיוון שהוא מפריד את הלוגיקה של המעבר על אובייקטים של Post ו Album מה Collection עצמו. ההפרדה הופכת את האופן בו ניגשים לאובייקטים ליותר קל מבלי לשנות את ה Collection או את הקוד שמתמש בו. ה Iterator Pattern מספק דרך סטנדרטית ונקייה לעבור על מקבץ אובייקטים ומספק Maintainability ו Reusability על ידי אינקפסולציה של לוגיקת ה Iterator.

* אופן המימוש:

גם PostIterator וגם AlbumsIterator מחזיקים רפרנס ל List של פוסטים או אלבומים ואינדקס m\_CurrentIndex בכדי לעקוב אחרי המיקום בנוכחי בזמן איטרציה.

המתודה HasNext בודקת האם יש יותר אלמנטים לעבור עליהם על ידי השוואה של האינדקס הנוכחי ומספר האלמנטים.

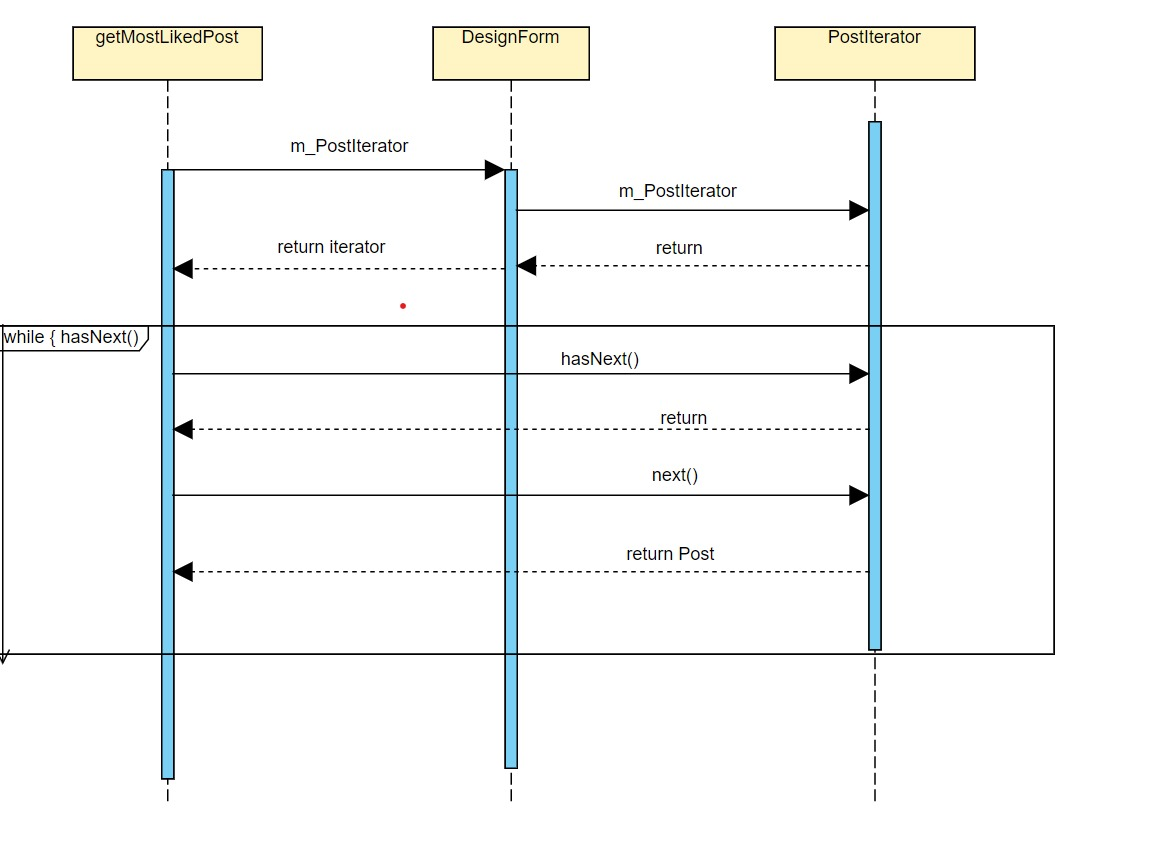
המתודה Next מנגישה את האלמנט הבא ומעלה את האינדקס. אם אין עוד אלמנטים מוציא שגיאה של InvalidOperationException   
המתודה CurrentIndex מחזירה את האינדקס הנוכחי.  
השימוש ב Iterator מתבצע כאשר המשתמש לוחץ על הכפתורים של getMostLikedButton ו getNumberOfPhotos. ב DesignForm מוחזקים 2 Instances של InterfaceIterator<T> וה events של הכפתורים משתמשים ב Iterator על מנת לעבור על האלבומים והפוסטים.

1. InterfaceIterator<T> interface : הוא Interface גנרי שמגדיר את החוזה עבור ה Iterator . הוא כולל את המתודות HasNext, Next , CurrentIndex .
2. PostIterator class: המחלקה הזו מייצגת את ה Iterator עבור ה Collection של האובייקט Post. הוא מממש את InterfaceIterator<Post> המאפשר גישה לאובייקטים של posts.
3. AlbumsIterator class: המחלקה הזו מייצגת את ה Iterator עבור ה Collection של האובייקט Album. הוא מממש את InterfaceIterator<Albumt> המאפשר גישה לאובייקטים של Albums.

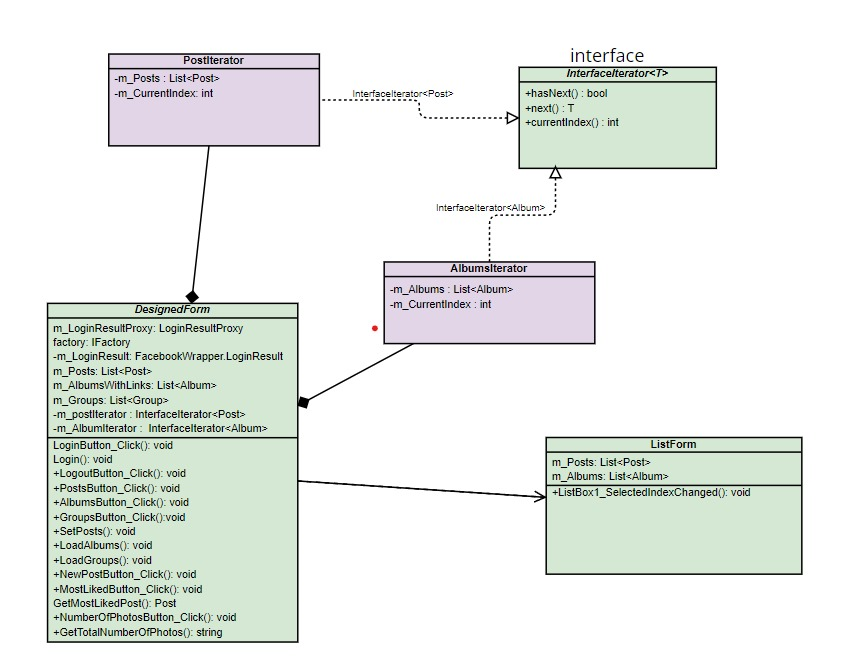
השחקנים – InterfaceIterator - Iterator Interface

PostIteratorClass , AlbumIteratorClass – Iterator classes

* Sequence Diagram



* Class Diagram



### תבנית מס' 3 – [Template Mathod]

* סיבת הבחירה / שימוש בתבנית:

סיבת הבחירה לשימוש ב Template Method היא שה Pattern הזה מאפשר פלקסביליות מנטנביליות ואקסטנסביליות. הוא מאפשר להגדיר אלגוריתמים במחלקה בסיסית ומאפשר לתתי המחלקות להתאים את האלגוריתמים כנדרש. זה עוזר במניעת שכפול קוד וגורם לקוד להיות יותר מינטנבילי.

בנוסף, הוא מאפשר בקלות להוסיף סוגים חדשים של data loaders במקרה שלנו מבלי לערוך את הקוד הקיים. כדי להוסיף סוג חדש של data laoder אנחנו בקלות יוצרים תת מחלקה של LoadDataTamplateMethod ומממשים את המתודה Load() על מנת לטעון את ה data הספציפי.

* אופן המימוש:

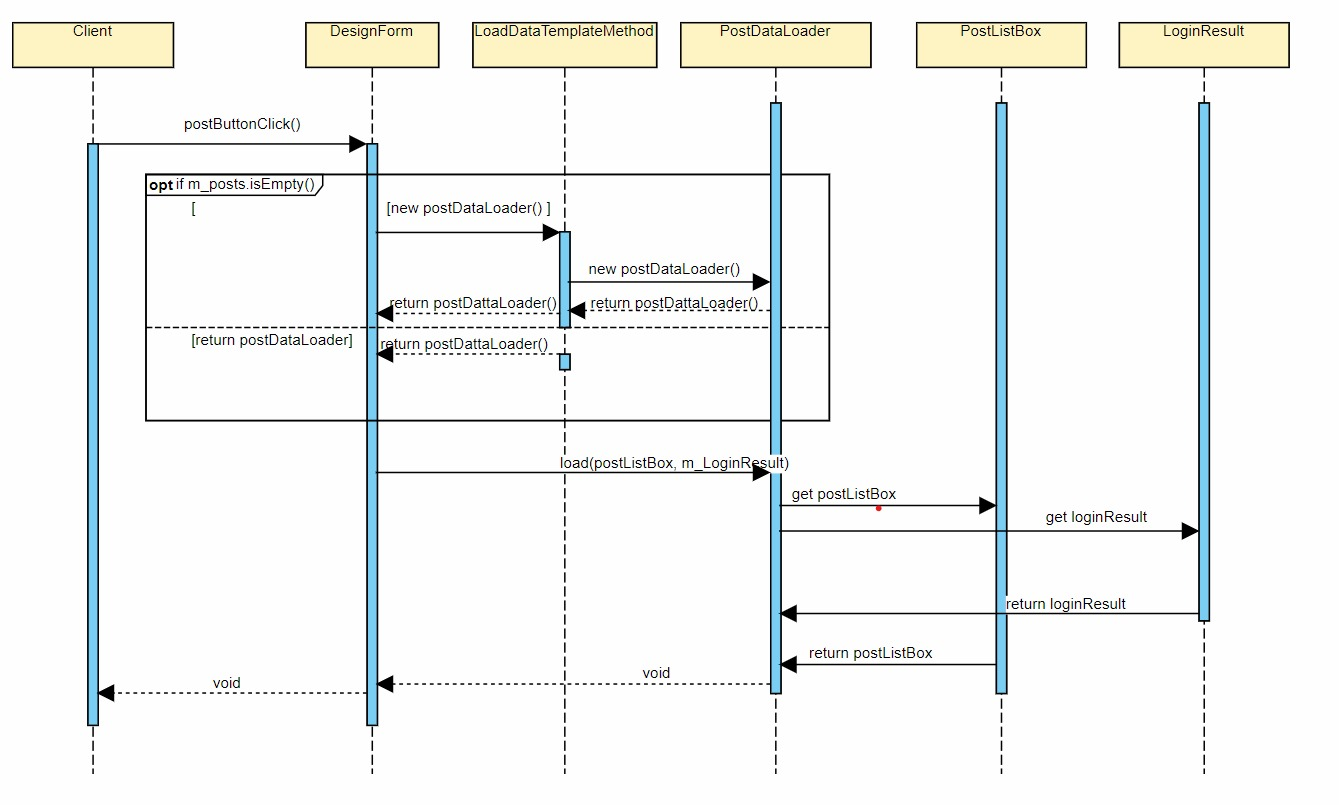
המחלקה LoadDataTamplateMethod מגדירה את הצעדים הבסיסים לטעינת ה data אל ה ListBox control . המתודה Load() היא template method – מגדירה את צעדי האלגוריתם אבל משאירה חלקים להיות ממומשים בעזרת תתי המחלקות.

המחלקות AlbumDataLoader ו PostDataLoader מרחיבות את LoadDataTamplateMathod ומממשות את הצעדים הספציפיים לטעינת האלבומים והפוסטים.

כאשר הכפתורים PostButton או AlbumButton נלחצים ה click event שלהם יוצר Instance של LoadDataTamplateMethod בשם loader ולאחר מכן קורא למתודה Load של loader אשר טוענת את האלבומים/ פוסטים.  
  
השחקנים – LoadDataTamplateMethod – Abstract template class

AlbumDataLoader , PostDataLoader – template classes

* Sequence Diagram



* Class Diagram

