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

* פיצ'ר מזג אוויר: מביא אינפורמציה על מזג האוויר, דרך API בשימוש בJson.
* פיצ'ר מניות מביא אינפורמציה על מניות , דרך API בשימוש בJson.
* יצירה\עריכת פוסטים מוכנים מראש:

המשתמש יכול ליצור פוסטים מוכנים מראש ב editor ,לערוך אותם,למחוק, להשתמש בהם כtemplate ולפרסם בפוסט. ההודעות המתאימות ישמרו על קובץ מקומי במחשב.

* פיצ'ר לסידור פוסטים שפורסמו לפי הא"ב.
* **יצירת יום הולדת לחבר (חדש**): אפשרות ליצור יום הולדת לחבר קרוב ויום הולדת לחבר רחוק על פי בחירת המשתמש. (פרסום הודעת מזל טוב ואלבום תמונות משותף – עבור חבר קרוב ועבור חבר רחוק יהיו 2 פוסטים שונים . עבור החבר הקרוב יותר תכתב ברכה אישית יותר, ועבור החבר הרחוק תכתב הודעה שאינה אישית). בנוסף הופסנו באיזור החברים באפליקציה לוח שנה משופר המשמש להצגת יום ההולדת הלועזי והעברי של החבר שנבחר מרשימת החברים בlistbox . לוח שנה זה מאפשר קבלת אינפורמציה אודות תאריכים עבריים על ידי בחירת תאריך מסוים. לועזי בלוח).
* הוספנו לפרוייקט אופציית Dark Mode כאשר המשתמש מסמן ב"צ'ק בוקס" אופציית זאת המסך הופך לכהה יותר

### תבנית מס' 1 – Template Method

* סיבת הבחירה / שימוש בתבנית:  
  השתמשנו בתבנית זאת על מנת למממש בפיצ'ר Dark Mode . בפרוייקט יש לנו מחלקה אבסטרקטית ,בעלת פונקציונליות של יצירת ערכת נושא וקיימות 2 מחלקות שצריכות לממש את הפונקציונליות הזאת באופן דומה אך לא זהה.המחלקה האבסטרקטית משאירה "חורים" עבור הבנים שירשו אותה למלא את "החורים" האלה בפונקציונליות קונקרטית.

פונקציונליות זו במחלקה האבסטרקטית משמשת כtemple method.

* אופן המימוש:   
  השחקנים: ThemeMaker היא המחלקה האבסטרקטית המחזיקה את מתודת התבנית MakeTheme (יוצרת את ערכת הנושא באופן כללי ואבסטרקטי) יחד עם ClearSpeisificTheme ("מנקה" את ערכת הנושא הספיצפית) וIntializeSpesificTheme (המאתחלת את ערכת הנושא) .

ThemeControlMaker וThemeComponentMaker הן מחלקות קונקרטיות היורשות מהמחלקה האבסטרקית.

ThemeControlMaker משתמשת במתודות שהוזכרו למעלה עבור איתחול ערכת הנושא עבור אותו "פקד" ספיציפי ואילו ThemeCompnentMaker מאתחלת עבור כל בניה של אותה קומפוננטה.

מתודת "ניקוי" עבור ThemeControlMaker מאתחלת את הפקד לערכת הנושא (הצבעים ) המקוריים ואילו ThemeComponentMaker עבור כל בניה.

MakeTheme עבור מחלקת הרכיב, מאתחלת את הפקד בצבעים הנבחרים (עם מתודה נפרדת של איתחול באיטרצייה) ואילו עבור הפקד רק עבור אותו פקד.





### תבנית מס' 2 - Visitor

* סיבת הבחירה / שימוש בתבנית:

בחרנו להשתמש בתבנית זאת עבור פיצ'ר הdark mode .

במערכת שלנו יש לנו 2 רכיבים שעובדים עם הdarkmode שרוצים להימנע משיכפול קוד ע"י קומפוזיציה.

רצון ליצור ערכות נושא – צבעים הפקדים ,רקעים וכו' . יש לנו עבור פקדים (כפתורים,listview ,textbox וכו) רצון לשנות לdark mode ולשנות את ערכת הנושא. ויש לנו עבור רכיבים (panel,pane בעל ילדים (שהם פקדים) רצון לשנות את ערכת הנושא. המימוש הוא לא זהה עבור פקדים ועבור רכיבים ואנו רוצים להימנע משיכפול קוד (וכפי שכבר למדנו, ירושה היא דרך לא נכונה) ולהשתמש בקומפוזיציה "רגילה" תגרום לנו להכרות ציקלית . (הפקדים יכירו את מחלקת הtheme והtheme יאלץ להכיר את הפקדים\רכיבים.

בנוסף לרצון לא לשכפל קוד, שלא תהיה הכרות ציקלית, בצורה זו לא נוכל לעשות reuse למחלקה כי היא צריכה להכיר באופן קונקרטי את הרכיב המסויים ולכן לא להיות כללית.

לכן השתמשנו בinterface אשר יהיה themable, וכל כל רכיב ופקד יממשו את הממשק (ובעצם יהיו יצור מסוג זה). והמחלקות הקונקרטיות ידעו לעבוד עם יצור מסוג זה בלבד (ולא קונקרטי עם כפתור,טקסטבוקס וכו').

* אופן המימוש:  
  ה"שחקנים" בתבנית הזו : - IThemeableהממשק אשר מחזיק Thememaker עם get בלבד.

ThemeMaker היא מחלקה אבסטרקטית אשר מממשת את הIThemeable ומחזיקה את הthemeable מסוג הממשק

ממחלקה זו יורשות ThemeComponentMaker וThemeControlMaker .

כל קומפוננטה וכל פקד (לדוגמא LabelTheme) תממש את IThemeable ואת פונקציונליות הThemeControlMaker באופן קונקרטי (תהפוך את הטקסט בצבע המסויים של הdarkmode ,תשנה את הרקע וכו')





### תבנית מס' 3 – Observer

* סיבת הבחירה / שימוש בתבנית:

בפיצ'ר של darkmode כאשר המשתמש מסמן את הצ'ק בוקס ורוצה שהאפליקציה תהפוך לדארק מוד הרכיבים והפקדים צריכים להתעדכן בכך ובעצם לשנות את ערכת הנושא.

השתמשנו בobserver delegate.

כאשר המשתמש מסמן\מוריד את הסימון של הdarkmode המסך הראשי מיידע את מי ש"נרשמו" שהוא נלחץ וגורם להם לשנות את ההתנהגות שלהם (במקרה שלנו את הצבעים של הרכיבים והפקדים)

* אופן המימוש:

השחקנים: MainForm – משמש כsubject המחזיק ב delegate ColorSchemeNotifyerDelegate(bool i\_Dark) אשר צריך ליידע כאשר הstate הוא על darkMode או לא. שם נמצאות 2 המתודות AddListeners(ColorSchemeNotifyerDelegate) וRemoveListeners(ColorSchemeNotifyerdelegate) המשמות כ-attach &detach

המתודה notifyAllListeners אחראית ליידע את כל מי שנרשם לכך ולעשות invoke לפי הstate ( (i\_dark ל-ColorSchemeNotifyerDelegate

הFormTheme שממנו יורשים כל ה forms הן הobserver ומתודת update היא

enableOrDisableColorScheme(bool i\_Dark)



