

**Bob Project**

**תוכנה ליצירת XML-ים**

**מסמך תכנון ועיצוב**

**סדנה בתכנות מכוון עצמים (20586)**

**מנחה: דני כלפון**

**מגישים: אייל לוי ושאול רויזן**

אוגוסט 2014

# תיאור כללי של המערכת

המערכת בנויה ממודל 4 שכבות מופרדות זו מזו:

* שכבת ה Data Base (DB).
* שכבת ה Data Access Layer (DAL).
* שכבת ה Business Logic (BL).
* שכבת ה Graphical User Interface (GUI).

כל שכבה מתקשרת עם השכבה שמתחתיה, כאשר שכבת ה GUI היא העליונה ומכירה רק את ה BL, ה BL מעל ה DAL וה DAL מעל ה DB. בצורה זו, הממשק בין השכבות מאפשר החלפת כל אחת מהשכבות ללא שינוי בשכבות שהאחרות, בלבד שיתקיים הממשק ביניהן.

### הנחות עבודה בכתיבת הפרוייקט

???

### מוסכמות רישום

1. כל שם מחלקה/מאפיין/מתודה/enum שכוללת יותר ממילה אחת, נכתבת עם אות גדולה בתחילת כל מילה נוספת.
2. כל מחלקה מתחילה באות גדולה. לממשקים נוספת האות I לפני שם המחלקה.
3. כל שדה (member) פרטי במחלקה מתחיל ב-'\_m'
4. Enum-ים מתחילים באות גדולה.
5. פרמטרים למתודה או משתנים בתוך מתודה מתחילים באות קטנה.
6. אין הבדל ברישום מתודות ציבוריות (public) ומתודות פרטיות (או מוגנות) מתחילות באות גדולה. כך גם לגבי מאפיינים (Property).

# שכבת בסיס הנתונים – XML ו XSD

בשכבה זו נבנה בסיס הנתונים עם המידע לו נדרשת המערכת. זוהי שכבה שמיוצגת ע"י קבצי נתונים, בעלי מבנה מוסכם ומתוקנן, ממנו שואבים את המידע. הסבר מפורט על הרכיבים:

* **קובץ XML:** קובץ עפ"י תקנון W3C, המתאר עץ היררכי, עם צמתים ומאפיינים, המתארים טיפוסים מסוגים שונים.
* **סכמה (XSD):** קובץ XML המתאר את משפחת ה XML-ים אותה הוא מייצג, נותן חוקיות, שמות וסדר המבנה של הקבצים.

# שכבת הגישה – DAL

שכבה זו אחראית על הגישה לבסיס הנתונים, בכך שהיא קוראת את הקבצים והופכת אותם לאובייקטי C#. ישנן מחלקות המיועדות לעבודה מול XSD ומול XML, היודעות לקבל נתיב והגדרות ולייצא אובייקט C#, בהן שכבת ה DAL משתמשת. שכבת ה DAL דואגת לתת את ההגדרות המתאימות, לטפל בתקינות ולהעביר את השגיאות והאובייקטים המוכנים לשכבות מעל.

הגישה מתחלקת, כמו ה Database, לשני חלקים: גישה ל XSD וגישה ל XML-ים השונים.

מחלקות גישה ל XSD

**XsdReader.cs**

אחראית לקריאה והחזקת האובייקטים ה .NET-יים המתארים את סכמת ה XSD, אחראית על ווידוא המבנה התקין של הקובץ והעלאת שגיאות במקרה הצורך.

public XsdReader(string schemaPath)

הבנאי מקבל את נתיב קובץ ה XSD, יוצר את האובייקט ה C# עבורו ע"י מחלקות .NET שיודעות לקרוא את הנתונים ולבסוף מבצע קומפילצייה של הסכמה בתוך ה SchemaSet, על מנת לאפשר ולידציה ושימוש בנתונים. את האובייקט הסופי שומרת כמאפיין public שדרכו שכבת הלוגיקה פועלת הלאה.

public XmlSchema Schema { get; set; }

בנוסף, נשמר הנתיב של הקובץ לשימוש עתידי:

public string Path { get; set; }

פונקציית הווידוא, שקורית באתחול האובייקט, ניתנת לקריאה בכל זמן והיא סטאטית על מנת לאפשר שימוש בה לאו דווקא עם הנתיב הנוכחי, אלא על מנת לוודא סכמה שרוצים לטעון בעתיד:

public static bool ValidateSchema(string schemaPath, bool throwException)

בנוסף, קיימת פונקצייה להתאמת XML כלשהו לסכמה הטעונה במערכת. זאת על מנת לוודא שה XML שטוענים מתאים ל XSD וניתן לעבוד עימו:

public bool IsXmlMatchSchema(XmlDocument doc, out string errorMessage)

מחלקות גישה ל XML

**XmlWriterWrapper.cs**

מחלקה לכתיבת תוצר לתוך XML. זוהי מחלקה סטאטית המכילה פונקציה יחידה המקבלת את אובייקט ה XML ה .NET-י ואת הנתיב לכתיבה, ומייצאת קובץ XML על פי האובייקט הניתן לנתיב הרצוי, תוך ווידוא תקינות מיקום.

public static bool WriteXml(XmlDocument document, string destinationPath)

**XmlQueryPartType.cs**

מחלקת מבנה נתונים המתארת בקשת חיפוש של מידע בתוך XML. במחלקה המאפיינים הבאים:

public string QueriedNode { get; set; }

public string AttributeName { get; set;

public string AttributeValue { get; set; }

והם: שם הצומת שרוצים לחפש בתוכו, שם המאפיין של הצומת לעשיית השאילתה, ערך המאפיין הדרוש.

**XmlSearcher.cs**

מחלקה המטפלת בחיפוש בתוך XML אחר המידע הרצוי. בעלת פונקציה סטאטית בודדה:

public static List<XmlNode> SearchXml(XmlDocument document, XmlQueryPartType query)

מחפשת בתוך XML את הנתונים הנדרשים עפ"י השאילתה המבוקשת, ומחזירה רשימה של צמתים העונים על הדרישה.

**XmlLoaderWrapper.cs**

מחלקה לטעינת XML מתוך קובץ לתוך אובייקט .NET סטנדרטי. בעלת פונקציה סטאטית בודדה:

public static XmlDocument LoadXml(string xmlPath)

יוצרת ומחזירה את אובייקט ה .NET המתאים תוך ווידוא שגיאות.

# שכבת הלוגיקה – BL

שכבה זו מקשרת בין שכבת קריאת בסיס הנתונים, ה DAL, לבין שכבת התצוגה, ה GUI. תפקידה העיקרי של השכבה הוא לעטוף את האובייקטים ה .NET-יים שה DAL מספק לה ולאפשר גישה נוחה לנתונים, מניפולציה שלהם, ניתוח המידע וחשיפת ממשק לשכבת התצוגה הפשוט ביותר.

***עקרונות מנחים***

* ה BL תייצא את המידע הדרוש בלבד מתוך אובייקטי ה DAL.
* כלל ה Exceptions, הן מה DAL והן מה BL עצמו, יועלו הלאה את שכבת ה GUI על מנת לאפשר הצגת שגיאות למשתמש.
* עיקרון הכימוס (Encapsulation) בא לידי ביטוי בשכבה זו, בכך שהיא מסתיר את המידע הנשמר במחלקות וחושפת אותו כלפי חוץ באמצעות תכונות (Properties) בלבד. הדבר נעשה גם אם אין הגבלות מסוימות על הערך של המידע ולא מתבצעת שום בדיקה בגוף של התכונה, וזאת כדי לאפשר בעתיד הוספה של בדיקות כאלו לכל מידע שנשמר, בלי צורך לשכתב קוד קיים במקומות אחרים.
* עקרון ה SRP בא לידי ביטוי ע"י כך שהאחריות בין המחלקות השונות, כאשר כל מחלקה מקבלת תפקיד ספציפי ורק הממשק כלפי ה GUI מכיל את כל שאר יכולות ה BL.
* עקרון הפולימורפיזם בא לידי ביטוי בכך שישנן מחלקות abstract מהן יורשות מספר מחלקות ומאפשרות להשתמש בהן מבלי לדעת את המימוש הספציפי של כל אחת מהן או אפילו מאיזה טיפוס מדויק הן.
* מחלקות מכילות מחלקות אחרות על מנת להימנע מירושה מסורבלת ולהרחיב יכולות, תוך שימוש ב Design Patterns שונים.