**מבוא**

ייזום:

הפרויקט הוא שפת תכנות חדשה שהמצאתי, ששמה "Little". הפרויקט כתוב בשפת התכנות Haxe, ומאפשר הרצה של קוד הכתוב ב-Little מצד המשתמש, ובמיוחד מאפשר למפתח המשתמש בפרויקט לאלמנטים באפליקציה שלו לעקוב בקלות אחרי כל אירוע שקורה, ולהוסיף לשפה פונקציונליות צד-שלישי, על מנת לשפר פרודוקטיביות באותה אפליקציה.

קיבלתי את ההשראה להכין פרויקט בסדר גודל כזה משני אלמנטים בספריית תכנות שלי, Texter:

* Parser של Markdown, שגם נותן אופציית ויזואליזציה של הMarkdown
* אלגוריתם החלטת כיוון וסידור טקסט למחרוזות המכילות טקסטים שאמורים לזרום לכיוונים שונים (לדוגמה: אנגלית ועברית, צרפתית וערבית), בכמה שורות/פסקאות שונות.

שני האלמנטים עירבו יצירה והחלטה על Tokens שייצגו את המידע, ומניפולציה על אותם Tokens, על מנת להציג את התוצאה הרצויה (טקסט שמוצג נכון ומסמך בסטייל Markdown, מהמקרים שהוזכרו קודם). כשלמדתי שקומפיילר/אינטרפרטר של שפת תכנות זה, בסופו של יום, אותו הדבר, רק בסדר גודל יותר רציני - התעניינתי, והתחלתי לנסות לתכנת שפה משלי...

ממש בהתחלה, חשבתי שזה יהיה יחסית פשוט – הנחתי שהתהליך יהיה ישיר בערך כמו הדברים שעשיתי בעבר, אבל תוך כדי עשייה, הבנתי שאני הולך להיתקל ביחסית הרבה מכשולים. העיקריים היו:  
 - זריקת שגיאות מובנות ומתאימות להקשר  
 - מערכת סוגים – מה עדיף - מערכת חזקה או מערכת חלשה?  
 - ניהול זיכרון – באילו מבנים אשתמש על מנת לאחסן זיכרון?  
 - וכמובן: איך בכלל מתמודדים עם כמות כזאת גדולה של נתונים, בלי לקחת הרבה זמן?

מההתחלה, החלטתי שהפרויקט יכוונן לשתי מטרות, יחסית ספציפיות, אך שימושיות:

* ראשונה: יצירת שפת תכנות שקל ללמד וגם קל "לשנות", על מנת להנגיש לימוד תכנות לילדים, ובמיוחד לילדים שלא בהכרח יודעים אנגלית. כאן לפרויקט שלי יש יתרון מובהק על פני אופציות אחרות זמינות: הפרויקט עוצב תוך כדי התחשבות בריבוי שפות, כך שכל keyword או אלמנט בStandard Library יכול להיות מוחלף על ידי כל מילה אחרת בכל שפה אחרת, דבר שמאפשר לתכנת לא רק בשפה האנגלית, אלא גם בשפות מדוברות אחרות, כמו עברית וערבית.
* שנייה: שימוש ככלי פרודוקטיביות בתוך תוכנות קיימות – כמו שהתוכנה Excel מאפשרת לשנות ערכים של משבצות בעזרת קוד הכתוב ב-Visual Basic Analysis, ככה תוכנות המשתמשות בLittle כספרייה יוכלו לספק דרכי אינטראקציה עם התוכנה באמצעות קוד הכתוב בLittle. גם כאן לפרויקט יש יתרון ברור, שכן לא רק שמצד הלקוח, כתיבת קוד בLittle זה דבר קל בגלל הsyntax הפשוט והעקבי של השפה, גם מצד המפתח, מכיוון שחלק משמעותי מהפרויקט מתרכז בהקלה על הוספת אלמנטים אקסטרניים לשפה, זה יהיה מאוד קל ומהיר לשלב את הפרויקט בתוכנה של המפתח, ויחסוך ממנו המון כאב ראש.