מה ההבדל בין מפרשן (interpreter) למהדר (compiler)?

שניהם הרי גורמים לקוד תוכנה לרוץ על מכונה כלשהי.

**מהדר (Compiler)** – מתרגם תוכנה שנכתבה בשפה “קריאה לאדם”, כיחידה שלמה, לקוד בשפת מכונה (אפסים ואחדים) שמבוצע ישירות על המעבד.

**מפרשן (Interpreter)** – מריץ את התוכנה, שורה אחר שורה ומפרשן כל שורה בנפרד. כל שורה עוברת תרגום “משפת אדם” לשפת מכונה – ואז מורצת, וחוזר חלילה.

אז אם מהדר בעצם מבצע תרגום משפת אדם לשפת מכונה,

למה צריך בעצם גם מפרשן?

המוטיבציה העיקרית לכתיבת מפרשן היא, כנראה, העובדה שקל הרבה יותר לכתוב אותו. עבור שפות שאינן דורשות חישובים מהירים במיוחד, המפרשן הוא אלטרנטיבה המספקת חסם כניסה נמוך לשפות חדשות. ניתן בהשקעה קטנה להגדיר שפה חדשה ולגרום לה לרוץ. מאוד אג’ייל.

מפרשן יכול לספק debugger בהשקעה אפסית כמעט,

עוד יתרון. הקלות בכתיבת מפרשן יכולה להיות מתורגמת בקלות לכתיבת שפה

cross-platform

אולי כדאי שבעצם נשאל את עצמנו מה היתרון בכתיבת מהדר? מדוע להשקיע מאמץ רב בתוכנה שתתרגם שפה לפלטפורמה יחידה, ושמציבה אתגר משמעותי בכתיבת debugger?

התשובה היא פשוטה: ביצועים. מהדר עובד קשה פעם אחת – אך הוא מספק קוד מכונה שיכול לרוץ בצורה אופטימלית ללא התערבות בזמן ריצה. מפרשן לעומת זאת – היא תוכנה שרצה ומפרשנת שורה אחר שורה. היתרון די ברור...

**שפת JavaScript**

ג’אווהסקריפט הייתה במשך שנים דוגמה קלסית לשפת מפרשן. הסיבה העיקרית הייתה כנראה הרצון להריץ קוד שלה על סביבות שונות: כתוב את הקוד פעם אחת והוא ירוץ על ספארי (מק), FF (לינוקס) או IE (חלונות). אם הקוד קומפל לשפת מכונה אחת – הוא לא יתאים למכונה אחרת ולכן האפשרות היחידה היא לשלוח את קוד המקור למחשב עליו רץ הדפדפן – ושם לפרשן אותו.

בשנים האחרונות, הפכו נפוצים מהדרי JIT לשפת ג’אווהסקריפט. דוגמה בולטת היא מהדר ה JIT של V8 – מנוע הגא’ווהסקריפט של גוגל. התוצאה: קוד המקור נשלח לדפדפן לפני ההרצה, אך הדפדפן מחליט להדר אותו ואז להריץ אותו בשפת מכונה – כך שבפועל כיום, JavaScript רצה מהודרת. הנה דוגמה ליתרון שבהידור. דמיינו את הקוד הנפוץ הבא:

**var a.b.c.d.e.f.g.h.x = 100;**  
**var a.b.c.d.e.f.g.h.y = 200;**

כאשר a.b.c.d.e.f.g.h הם שמות namespaces או אובייקטים בג’אווהסקריפט. כאשר יש מפרשן, עליו לקחת את אובייקט a ואז לחפש בו property בשם b, ואז על b לחפש property בשם c וחוזר חלילה. ללא שיפורי ביצועים מיוחדים יכול להיות שמדובר בעשרות פעולות בזיכרון.  
מהדר לעומת זאת קורא את 2 השורות ומבין שיש פה אופציה לייעל את הקוד. הוא יכול להפוך את הקוד ל:

**var \_p = a.b.c.d.e.f.g.h;**  
**var p.x = 100;**  
**var p.y = 200;**

שיהיה יעיל בהרבה. מכיוון שג’אווהסקריפט היא שפה דינמית, עליו לוודא ש a עד h לא השתנו בעקבות ההשמה של x. כאן שוב יש מין “מיקרו מפרשן” שהמהדר מייצר ופועל בזמן ריצה.