אפיון ומדריך לביצוע

מערכת: פרקטיקוד
קורס: Experience 1 - כלי Files Bundler
ספר: אפיון ומדריך לביצוע

תוכן עניינים

תיאור הפרויקט

כתלמידת מגמת תכנות מפעם לפעם את נדרשת להגיש תרגילים לבדיקה.

כאשר התרגילים כוללים מספר קבצי קוד, הבדיקה מסורבלת ולא נוחה. את מגישה תיקיה של הפרויקט והמורה שבודקו בתוכנה המתאימה ולעבור על הקבצים השונים כדי לראות את הקוד.

מכיון שפעמים רבות אין למורה ענין להריץ את הקוד, אלא רק לקרוא את הקוד כדי להתרשם מהעבודה שעשית, מדובר

בפרויקט זה נפתח כלי לאריזת קוד מקבצים שונים בקובץ אחד.

הכלי יהיה בתצורה של CLI - ממשק שורת פקודה, והשימוש בו יהיה באופן של הרצת פקודה על תיקיית הפרויקט הרלו

הרצת הפקודה תיצור את הקובץ שמכיל את כל הקוד הרצוי שאותו תוכלי להגיש למורה הבודקת.

פיתוח CLI ב-.NET

הספריה System.CommandLine מספקת את הפונקציונליות שנדרשת לאפליקציות שורת פקודה.

הספריה נמצאת כרגע בגירסה ניסיונית ויתכנו שינויים בגירסאות הבאות.

בפרקים הבאים תמצאי הסבר על צורת העבודה עם הספריה והמונחים הקשורים אליה.

ההסבר מבוסס על המדריך המלא של מיקרוסופט בקישור הזה:

otnet/standard/commandline/syntax

תחביר שורת הפקודה

System.CommandLine מנתח את הקלט של שורת הפקודה לטוקנים שהם מחרוזות מופרדות ברווחים.

לדוגמא הפקודה הזו מנותחת לטוקנים tool, install, dotnet-suggest, --global, --verbosity, quiet

.NET CLI Copy

```
dotnet tool install dotnet-suggest --global --verbosity quiet
```

טוקנים יכולים להתפרש כפקודות, אפשרויות או ארגומנטים.

אפליקציית ה-CLI היא זו שקובעת איך הטוקנים יתפרשו.

Commands - פקודות

פקודה מציינת פעולה או מגדירה קבוצה של פעולות קשורות.

לדוגמא:

בפקודה dotnet run, הטוקן run הוא פקודה שמציינת פעולה.

בפקודה dotnet tool install, הטוקן install הוא פקודה שמציינת פעולה ו-tool הוא פקודה שמציינת קבוצה של פקודו כמו: tool uninstall, tool list ועוד.

Root commands

פקודת שורש היא זו שמציינת את שם קובץ ההפעלה של האפליקציה.

לדוגמא הפקודה dotnet מציינת את קובץ ההרצה dotnet.exe

Subcommands

רוב אפליקציות ה-CLI תומכות בפקודות משנה שנקראות גם פעלים (verbs).

לדוגמא לפקודה dotnet יש פקודת משנה run שמופעלת בהזנה של dotnet run.

לפקודות משנה יכולות להיות פקודות משנה משלהן.

לדוגמא בפקודה dotnet tool install הפקודה install היא פקודת משנה של פקודת המשנה tool.

Options

אפשרות היא פרמטר שניתן להעביר לפקודה. המוסכמה המוגדרת בתקן POSIX היא להוסיף ל-option קידומת של שו

הדוגמא הבאה מראה פקודה עם שתי אפשרויות

.NET CLI Copy

```
dotnet tool update dotnet-suggest --verbosity quiet --global
                                  ^---------^       ^------^
```

כפי שהדוגמא הזו ממחישה, הערך של אפשרות יכול להיות מפורש (explicit), כמו הערך quiet עבור האפשרות osity
(implicit), כמו באפשרות global שלא מקבלת ערך אחריה. אפשרויות שאין להן ערך הן בדרך כלל פרמטרים בוליאניי
בפקודה ו-false אם האפשרות לא מופיעה בפקודה.

Arguments

ארגומנט הוא ערך שמועבר לאפשרות או פקודה.

בדוגמא הבאה הערך quiet הוא ארגומנט של האפשרות verbosity

Console Copy

```
dotnet tool update dotnet-suggest --verbosity quiet --global
                                              ^---^
```

לארגומנטים יכולים להיות ערכי ברירת מחדל שיחולו אם לא סופק ארגומנט במפורש.

פעמים רבות, אפשרויות הן באופן משתמע פרמטרים בוליאניים עם ערך ברירת מחדל של true כאשר שם האפשרות מ

בדוגמא הבאה שתי הפקודות שקולות.

.NET CLI Copy

```
dotnet tool update dotnet-suggest --global
                                  ^------^

dotnet tool update dotnet-suggest --global true
                                  ^-----------^
```

סדר האפשרויות והארגומנטים

סדר הכתיבה של אפשרויות וארגומנטים לא משנה. ניתן לכתוב אפשרויות לפני ארגומנטים או ארגומנטים לפני אפשרויו

בדוגמא הבאה שתי הפקודות שקולות.

.NET CLI Copy

```
dotnet add package System.CommandLine --prerelease
dotnet add package --prerelease System.CommandLine
```

כמו כן, ניתן לציין את האפשרויות בכל סדר שהוא.

לעומת זאת, כאשר יש מספר ארגומנטים לאותה פקודה או אפשרות, הסדר בהחלט משמעותי.

בדוגמא הבאה נשלחים אותם ארגומנטים בסדר שונה. התוצאה עשויה להיות שונה בהתאם למימוש.

Console Copy

```
myapp argument1 argument2
myapp argument2 argument1
```

Aliases

בהתאם לסטנדרט POSIX זה נפוץ שלחלק מהפקודות והאפשרויות יש כינויים. בדרך כלל צורה מקוצרת שמקלה על הר
גם למטרות אחרות, כמו לאפשר חוסר רגישות לאותיות גדולות וקטנות ולתמוך באיות אלטרנטיבי של המילה.

בדרך כלל כינוי של אפשרות הוא באמצעות מקף אחד ואות בודדת.

לדוגמא

.NET CLI Copy

```
dotnet build --verbosity quiet
dotnet build -v quiet
```

Response files

קובץ תגובה הוא קובץ שמכיל סט של טוקנים עבור אפליקציית CLI.

זהו פיצ'ר של System.CommandLine ששימושי בשני סוגי מקרים:

- להפעיל פקודה עם קלט ארוך שחורג ממגבלת התווים של הטרמינל (ה-cmd).
- להפעיל את אותה פקודה שוב ושוב ללא צורך בהקלדה חוזרת של כל השורה.

כדי להשתמש ב-response file יש להכניס את שם הקובץ עם תחילית של @ במקום שבו רוצים להכניס את הפקודות,

סיומת הקובץ יכולה להיות כל סיומת שהיא. נפוץ להשתמש בסיומת של rsp.

הפקודות הבאות שקולות.

.NET CLI Copy

```
dotnet build --no-restore --output ./build-output/
dotnet @sample1.rsp
dotnet build @sample2.rsp --output ./build-output/
```

התוכן של sample1.rsp

Console Copy

```
build
--no-restore
--output
./build-output/
```

התוכן של sample2.rsp

Console Copy

```
--no-restore
```

להלן כללי התחביר שקובעים איך הטקסט ב-response file מיוצג:

- טוקנים מופרדים ברווח. שורה שמכילה את המילים Good Morning מטופלת כשני טוקנים: Good ו-Morning.
- מספר טוקנים שעטופים במרכאות מתפרשים כטוקן אחד. שורה שמכילה את המילים "Good Morninig" מטופלת

- כל טקסט שבין הסימן # ועד סוף השורה מטופל כהערה וזוכה להתעלמות.
- טוקנים שמתחילים ב-@ יכולים להפנות ל-response file נוסף.
- response file יכול להכיל שורות מרובות של טקסט. השורות משורשרות ומתפרשות כרצף אחד של טוקנים.

כללי אצבע בפיתוח CLI

שמות קצרים

מומלץ להגדיר שמות קצרים וקלים לאיות ככל האפשר.

שמות באותיות קטנות

מומלץ להגדיר שמות באותיות קטנות בלבד. ניתן להגדיר בנוסף כינויים כדי לתמוך גם באותיות גדולות.

kebab case

שם שמורכב מכמה מילים, מומלץ להפריד ביניהן במקף. לדוגמא: --additional-probing-path

יחיד ורבים

מומלץ לשמור על עקביות בשמות עבור אפשרויות שיש להן כמה ערכים ולא לערב שמות ביחיד וברבים.

Option names	Consistency
`--additional-probing-paths` and `--sources`	✓
`--additional-probing-path` and `--source`	✓
`--additional-probing-paths` and `--source`	✗
`--additional-probing-path` and `--sources`	✗

פעלים לעומת שמות עצם

מומלץ להשתמש בפעלים עבור פקודות שמתיחסות לפעולות ובשמות עצם עבור אפשרויות.

יצירת הפרויקט

צרי פרויקט מסוג Console Application. קראי לו בשם קצר ככל האפשר, כיון ששם הפרויקט משפיע על שם קובץ ה-
שיש להקליד כאשר רוצים להריץ את הפקודות.

(תוכלי לקרוא ל-solution בשם מלא ולשנות אחר כך את שם הפרויקט הפנימי לשם קצר)

התקיני את הספריה System.CommandLine בלחיצה ימנית על הפרויקט -> בחרי בתפריט age NuGet Pacakges
השם (שימי לב שאת בכרטיסיית Browse). לחצי על Install.

עלייך לפתח CLI שיכיל שתי פקודות כפי שמפורט בפרקים הבאים.

📺 **ראי סרטון הדרכה בשם CLI in .NET**

פקודת bundle

צרי פקודה בשם bundle שתטפל באריזת קבצי קוד לקובץ אחד.

הגדירי לפקודה את האפשרויות (options) שלהלן:

language

רשימת שפות תכנות. האפליקציה תכלול ב-bundle רק קבצי קוד של השפות הנבחרות.

אם המשתמש יקיש את המילה all, יכללו כל קבצי הקוד שבתיקיה.

אפשרות זו היא חובה (required).

output

שם קובץ ה-bundle המיוצא.

ניתן להקיש שם קובץ בלבד ואז הקובץ ישמר במיקום שבו המשתמש הריץ את הפקודה.

לחילופין ניתן להקיש ניתוב מלא.

אם האפליקציה לא תצליח לשמור בניתוב זה היא תספק הודעת שגיאה מתאימה.

❗ יש לשים לב שאם יש רווח בניתוב של הקובץ (לדוגמא אם שם התיקיה מורכב משתי מילים) יש להקיש אותו בשורר

note

האם לרשום את מקור הקוד כהערה בקובץ ה-bundle.

כאשר המשתמש מקליד אפשרות זו בפקודה, האפליקציה תדאג לרשום בקובץ ה-bundle הערה עם מקור הקוד - שם

sort

סדר העתקת קבצי הקוד, לפי א"ב של שם הקובץ או לפי סוג הקוד.

ערך ברירת מחדל יהיה לפי א"ב של שם הקובץ.

remove-empty-lines

האם למחוק שורות ריקות.

כאשר המשתמש מקליד אפשרות זו בפקודה, האפליקציה תנקה את השורות הריקות מקוד המקור לפני שתעתיק אותו

author

רישום שם יוצר הקובץ.

המשתמש יוכל לספק את שם יוצר הקובץ והאפליקציה תדאג לרשום בראש קובץ ה-bundle הערה עם השם שסופק.

פקודת create-rsp

צרי פקודה נוספת בשם create-rsp שתיצור קובץ תגובה עם פקודה מוכנה.

הרצת הפקודה תציג למשתמש שאלה מה הערך הרצוי עבור כל אחת מהאפשרויות הקיימות, ותיצור בהתאם לתשובות

הנחיות כלליות

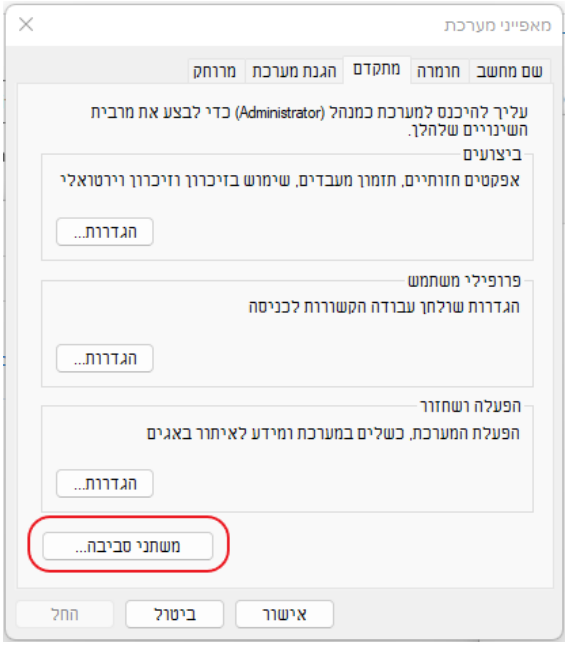
❤️‍🔥 שימי לב לנקודות הבאות

- הגדירי כינוי מקוצר (alias) עבור כל אחת מהאפשרויות.
- עלייך לדאוג לבצע בדיקות תקינות (ולידציה) על הקלט מהמשתמש ולהציג הודעה מסודרת במקרה שהקלט לא תקי
- אין לכלול ב-bundle קבצי קוד שנמצאים בתיקיות bin, debug וכדומה.

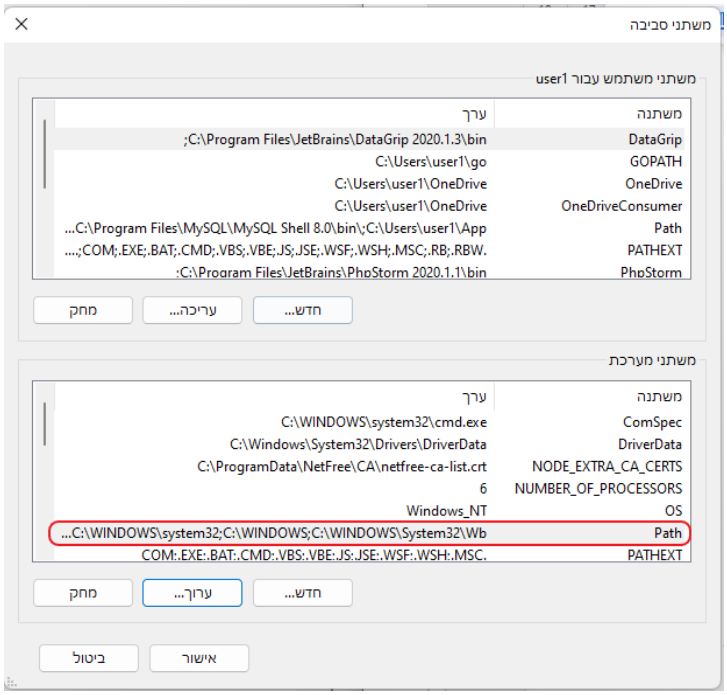
הגדרת ה-CLI ב-PATH

כדי שנוכל להריץ את פקודות ה-CLI שלנו בכל מיקום שבו נפתח את ה-cmd, נגדיר את מיקום קובץ ה-exe במשתנה ה

נפתח את לוח הבקרה -> מערכת -> הגדרות מערכת מתקדמות -> משתני סביבה...



(אפשר גם פשוט להקליד בתיבת החיפוש של התחל את המילים משתני סביבה ולהגיע ישירות לחלון.)



נלחץ לחיצה כפולה על המשתנה Path. בחלון שנפתח נלחץ על חדש ונוסיף את המיקום של תיקיית ה-publish שלנו ש

כעת נוכל לפתוח את ה-cmd מכל מיקום שהוא במחשב ולהריץ את פקודות ה-CLI שלנו.