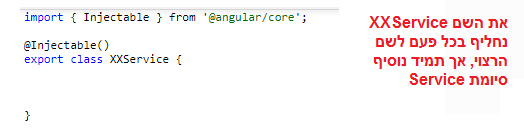
Services

התבנית הבסיסית ביותר להוספת service לפרויקט אנגולר, היא התבנית הבאה:

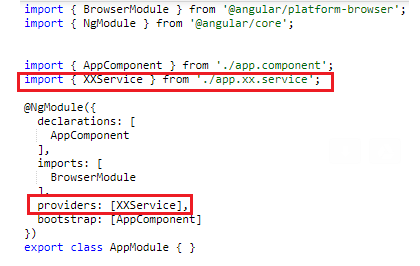
**שלב 1:**



יש לשים לב שעבור יצירת service נבצע import של injectable על מנת שנוכל להגדיר מעל הclass שניצור עבור הservice את הדקורייטיב של @injectable, ובכך לאפשר שהservice יהיה singleton ויוזרק אותו אובייקט לכל קומפוננטה שתשתמש בservice הזה.

**שלב 2:**

הוספת הservice לapp.module ע"י import, והוספת הclass של הservice למערך providers:



**שלב 3:**

השימוש בservice בתוך הקומפוננטה יעשה ע"י ביצוע import של הservice, ויצירת משתנה מטיפוס הservice בסוגריים של הconstructor.

שימו לב שלא מתבצע שום אתחול ע"י new ליצירת מופע של הservice, מכיוון שמדובר על שימוש באובייקט הsingleton שנוצר עבור כל האפליקציה, ולא נוצר אובייקט חדש של הservice הזה במיוחד עבור הקומפוננטה הנוכחית.

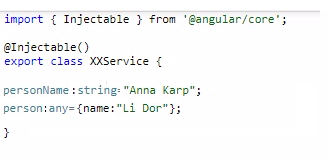


**חשוב לזכור: service נוצר פעם אחת בלבד עבור כל הקומפוננטות, ולכן כל שינוי שיבוצע בservice על ידי קומפוננטה מסוימת, ישתקף גם בקומפוננטות האחרות.**

נראה כעת דוגמא פשוטה בה הוספנו שני משתנים לservice והצגנו אותם בAppComponent:

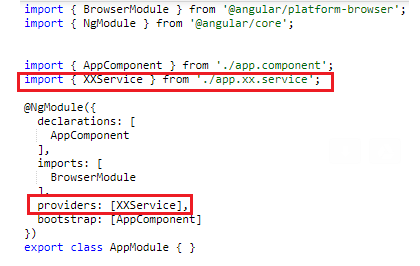
**שלב 1:**

יצרנו את קובץ הservice :



**שלב 2:**

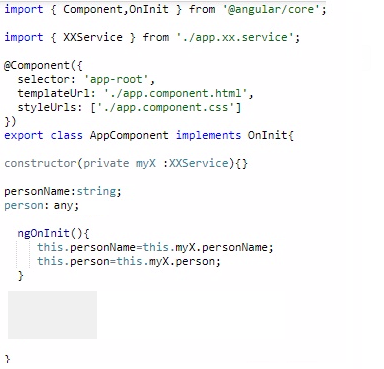
הוספת הservice לapp.module ע"י import, והוספת הclass של הservice למערך providers:



**שלב 3:**

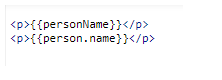
השימוש בservice בתוך הקומפוננטה ע"י ביצוע import של הservice, ויצירת משתנה מטיפוס הservice בסוגריים של הconstructor.

וכן בתוך הפונקציה ngOnInit , הוספנו אתחול ערכים של משתנים מקומיים השייכים לקומפוננטה, ע"י גישה לקריאת הערכים של משתני הservice והעתקתם לתוך משתני הקומפוננטה.



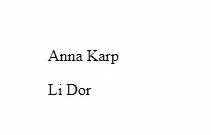
**שלב 4:**

ניצור את הhtml template ובו בעזרת אינטרפולציה נציג את שני המשתנים שהגדרנו בקומפוננטה:

****

**שלב 5:**

התוצאה שנקבל בדפדפן היא:



**דגש חשוב: בפונקציה ngOnInit בצענו העתקה של הערכים הקיימים בservice לתוך משתנה מקומי של הקומפוננטה, ואת המשתנה המקומי של הקומפוננטה הצגנו בhtml template.**

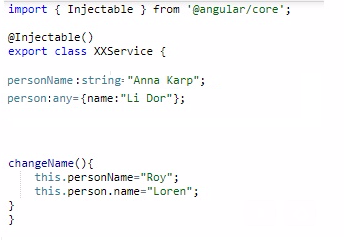
**עלינו לזכור שמשתנה פרימיטיבי (string / number / Boolean ) מועתקים למשתנה אחר ע"י העתקה by value , ולכן מרגע שביצענו בתוך הngOnInit את העתקת הערך של משתנה ה string השייך לservice אל משתנה מקומי של הקומפוננטה, כל שינוי שנבצע על המשתנה מסוג string בתוך הservice לא ישתקף לנו בקומפוננטה.**

**אבל – משתנה מטיפוס byRef (כגון: אובייקטים של class, או אובייקטי json וכו'), מועתקים רק ע"י העתקת המצביע לאותו אובייקט, ולכן כל שינוי על האובייקט השייך לservice ישתקף גם באובייקט של הקומפוננטה, מכיוון ששניהם מצביעים לאותו אובייקט.**

נפשט את האמור לעיל, על ידי דוגמא פשוטה, בה נוסיף לservice פונקציה שתשנה את תוכן המשתנים של הservice.

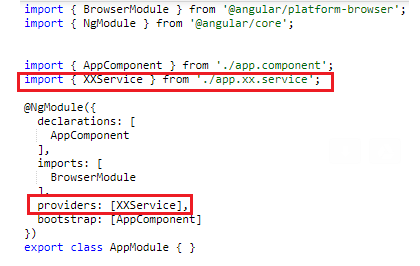
**שלב 1:**

יצרנו את קובץ הservice :



**שלב 2:**

הוספת הservice לapp.module ע"י import, והוספת הclass של הservice למערך providers:



**שלב 3:**

השימוש בservice בתוך הקומפוננטה ע"י ביצוע import של הservice, ויצירת משתנה מטיפוס הservice בסוגריים של הconstructor.

וכן בתוך הפונקציה ngOnInit , הוספנו אתחול ערכים של משתנים מקומיים השייכים לקומפוננטה, ע"י גישה לקריאת הערכים של משתני הservice והעתקתם לתוך משתני הקומפוננטה.

בנוסף, הוספנו לקומפוננטה פונקציה בשם setName שתבוצע ע"י האירוע click שנוסיף לכפתור שניצור ב html template, הפונקציה setName פונה לservice ומפעילה בו את הפונקציה changeName שמשנה בservice את ערכי המשתנים של הservice:



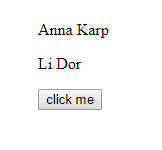
**שלב 4:**

ניצור את הhtml template ובו בעזרת אינטרפולציה נציג את שני המשתנים שהגדרנו בקומפוננטה, וכן נוסיף כפתור ונגדיר בו באירוע (click) קריאה לפונקציה setName שנמצאת בתוך הקומפוננטה:

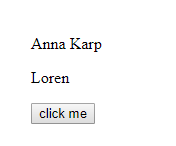
****

**שלב 5:**

התוצאה שנקבל בדפדפן היא:

****

וכאשר נלחץ על הכפתור נראה שנקבל את התוצאה הבאה:



כלומר – כפי שהוסבר לעיל, משתנה פרימיטיבי מטיפוס string לא התעדכן, ואילו משתנה by ref מטיפוס אובייקט, ידע להציג את השינוי שהתבצע באובייקט הservice מכיוון שהוא מצביע בדיוק לאותו אובייקט עליו משתנה הservice מצביע.