מגזין
עולם הבדיקות
רבעון שלישי 2020 גיליון מס' 22
www.testingworld.co.il

אבולוציית התשתית בהרצת סלניום, מהרצה מקומית ועד לדוקר בענן - יואב לוינסון

ראיון עם מנהל בדיקות אלברט אלרום - שירה נוסבוים

טיפים וטריקים בבדיקות סלנר חלק שלישי - סלביק פשנין

בדיקות משחקים אופיר אור

דבר העורך | ניצן גולדנברג

ניצן גולדנברג

מזה 5 שנים בתחום בדיקות התוכנה, תפקיד נוכחי מהנדס בדיקות בחברת SeatGeek, המוביל הראשי של קבוצת המיטאפ TestIL, מרצה בקורסים לבודקי תוכנה וחבר בצוות המייעץ AB של ITCB®.

קוראים יקרים,

לאחר תקופה של 3 חודשים, אנו רואים מעט אור בקצה המנהרה. השוק חוזר לחיים לאט לאט ואיתו גם הדרישה לבודקי תוכנה.

למרות שהמשבר הביא עימו לא מעט הרס, אנו ראינו שהוא הביא גם למידה, אני מדבר על הלמידה מרחוק. ראינו שבתקופה לא קלה זו השוק העולמי החל להשתמש יותר ויותר בשירותי למידה מקוונים וזאת על ידי שימוש בפלטפורמות שונות כגון: פלטפורמת תקשורת וידאו המאפשרת מתן שיעורים מהבית ודרך פלטפורמת שיתוף קבצים. אנו הוכחנו שלמרות המשבר אנו יכולים לקיים שגרת לימודים ומיטאפים ויתרה מכך, למדנו מה נחוץ לנו מבחינה טכנולוגית על מנת לאפשר קיום חיי שגרה במשבר עולמי.

מונח לפניכם גיליון מספר 22 של מגזין "עולם הבדיקות".

בגליון זה תוכלו למצוא מגוון רחב של מאמרים, בנוסף לטורים המעולים והקבועים שלנו:

אופיר מור כתב לנו מאמר על בדיקת משחקים ובכתיבתו, הוא מראה לנו שמשחקים זה לא רק לילדים שבתוכנו.

סלבה פשנין מביא לנו את החלק השלישי בסדרת המאמרים על טיפים וטריקים בבדיקות סלולר והפעם, סלבה יצור עבורנו "רשימת מכולת" של תכנים הנדרשים לנו ביום יום.

טליה יורמן מספרת לנו על הדרך שהיא עשתה בכדי להגיע לתפקיד הנוכחי שלה.

יואב לוינסון מספר לנו על האבולוציה של תשתית הרצת סילניום ועל טכנולוגיית הדוקר (Docker).

כמו כן תוכלו להנות מהטורים הקבועים שלנו: עולם הנגישות, סקירת כלים, האנציקלופדיה לבדיקות, בחן את עצמך, טור המייסדים, מחפש צרות ונגיסה מ-TestIL.

בנוסף, לאחר אין ספור מיילים ובקשות מכם הקוראים אנו מחזירים לכם את הטור "ראיון עם מנהל בדיקות" אשר נכתב ע"י שירה נוסבוים.

קריאה מהנה,
ניצן גולדנברג

עדכונים

- בחינות ההסמכה של ISTQB® חזרו כמובן לפי ההנחיות והמגבלות של משרד הבריאות
- ימי העיון אשר תוכננו להתקיים לפני משבר הקורונה חוזרים, לפרטים והרשמה יש להרשם http://bit.ly/TestIL
- תחרות הבדיקות הישראלית ISTC 2020 מתקיימת, ניתן להתעדכן בפורום ובאתר התחרות
- כנס Sela Developer Practice למפתחים, בודקים ומנהלים מתקיים בתאריכים 9-12/12/2020, למחיר מיוחד בהרשמה מוקדמת לחץ כאן

תוכן העניינים

ברכה לקהילת הבודקים | ירון צוברי 4
ראיון עם מנהל בדיקות - אלברט אלרום | שירה נוסבוים..... 5
בחן את עצמך - שאלה | טל פאר 8
בדיקות משחקים | אופיר מור 9
האנציקלופדיה לבדיקות | קובי יונסי..... 12
בחן את עצמך - תשובה | טל פאר..... 13
טיפים וטריקים בבדיקות סלולר - חלק שלישי | סלביק פשנין .. 14
המייסדים | אייל זילברמן 15
ארץ, עיר, חי, צומח ו-QA | טליה יורמן 16
מחפש צרות | מיכאל שטאל 18
כלי לבדיקות - Qase | ניצן גולדנברג..... 20
עולם הנגישות | אלון פרידמן ויסברד 22
אבולוציית התשתית בהרצת סלניום, מהרצה מקומית ועד לדוקר בענן | יואב לוינסון 24
מה הסקרים אומרים?... | יאן ברון 27
נגיסה מ-TestIL - מפגשים לבודקי תוכנה | ניצן גולדנברג..... 29
עורכי הגיליון..... 31

מו"ל
Israeli Testing Certification Board
ITCB®

ניהול המגזין
יאן ברון, *iMDsoft*

ניהול התוכן
קובי הלפרין, *Nokia*

עורך
ניצן גולדנברג, SeatGeek

עיצוב גרפי
בית נלי מדיה
סטניסלב קולנקו
www.beitnelly.com

יצירת קשר
אימייל:
info.testingworld@gmail.com

הרשמה
http://bit.ly/TW-Reg
פקס:03-6176605
כתובת: ברוך הירש 14 בני ברק 51202

www.testingworld.co.il

עולם הבדיקות נכתב ע"י בודקים עבור בודקים

ITCB® מקדמים את קהילת הבודקים בישראל

www.itcb.org.il

מגזין עולם הבדיקות

עולם הבדיקות הינו מגזין רבעוני.
כל הזכויות שמורות. זכויות היוצרים על חומר שפורסם על ידי המפרסם הינן רכושו של המחבר. הדעות המובאות במאמרים והתוכן לא בהכרח משקפים את דעת המפרסם. המחברים הינם האחראים הבלעדיים על תוכן מאמרם. מובהר כי העתקה ו/או נטילה שיטתית של מידע מהמגזין לצורך פעילות מסחרית ו/או עסקית, או לצורך כל פעילות אחרת שיש בה כדי לפגוע בפעילות העמותה, אסורה בהחלט.
לקבלת אישור לשימוש בתוכן צור קשר בדוא"ל info.testingworld@gmail.com

ברכה לקהילת הבודקים - ירון צוברי

חברים יקרים!

נראה כי אנחנו חוזרים אט אט ובבטחה לשיגרה. אני מניח כמו חברים נוספים, שהחזרה ההדרגתית וההתאוששות ממשבר הקורונה חשובים, למרות ההבנה שיתכן ויקח זמן מה עד שלחלוטין נתאושש ממנה בתעשיות השונות.

אכן, הקורונה תפסה את כולנו לא מוכנים והותירה אותנו לרגע מסויים מופתעים. היוזמות הוותיקות והחדשות שהצבנו לעצמנו לקדם עם הקהילה ועם החברים החדשים שמידי חודש מצטרפים לקהילה, החלו לקרום עור וגידים, ובאחת נחתה עלינו הקורונה ואילצה אותנו לעצור ולבצע הערכה מחדש. לי ולחברי ארגון ITCB היה ברור כי לשדר "עסקים כרגיל" מחד זה חשוב, אך מאידך המציאות היא זו שמכתיבה והיה לנו באותה מידה ברור שלא כל העסקים לא ישארו במתכונת הרגילה. פתחנו את השנה עם קריאה להשתתפות בתחרות הבדיקות הישראלית ISTC, שנערכת זו השנה הרביעית. חידשנו יחד עם הצוות המופלא את האתר של התחרות, ונערכנו כמו בכל השנים האחרונות לקיים אותה יחד עם ג'ון ברייס וחברות נוספות אשר מתמידות לשתף איתנו פעולה גם השנה. לצערנו נאלצנו להודיע על דחיה ולבקש את סליחת אלו שמיהרו להירשם ולומר תודה רבה על הנחישות, רוח הספורט וההבנה שגיליתם בתקופה זו. אנחנו עדיין לא אמרנו את מילתנו האחרונה בנושא התחרות ובפגישת הצוות האחרונה הוחלט לקיים את התחרות ולעדכן אתכם במועדים החדשים, וכך יעשה בקרוב.

ההדדיות ושיתוף הפעולה שקיימים עם נציגי המוסדות האקדמיים, עם השותפים לדרך במרכזי ההדרכה השונים, עם הנציגים של גופי ההכשרה הצבאית השונים ובסמ"ח בעיקר, מנהלים מענף ההכשרות הממשלתי – התמ"ת - וחברים מהחברות בתעשיה, העצימה בתקופה הזו. ההבנה שאנחנו פועלים ביחד חיזקה את הקשר ואנחנו באירגון נערכנו לתת מענה לצרכים ולדרישות שהועלו על ידי הגופים השונים, כגון: הפצת שאלוני דוגמה נוספים ומעודכנים לתוכנית הלימוד הבסיסית שיצאה לאור לפני שנה, התאמת מתכונת קיום בחינת ההסמכה לכללים שנקבעו ע"י משרד הבריאות, התאמת התכנים במערכי הלימוד שישמשו בהכשרות של עובדים חדשים ועוד. אני רוצה להודות לכל החברים והחברות באירגונים השונים על שיתוף פעולה הדדיות והבנה מחממי הלב.

מדהים לראות עד כמה הקהילה חזקה, והיה חשוב לנו להמשיך ולהפיק גליונות נוספים של מגזין "עולם הבדיקות", ולהעשיר את מאגר הכתבות לגליונות הבאים. חוויה מעשירה וכייף אמיתי לראות איך חברי הקהילה משתפים בידע והניסיון שרכשו עם השנים. אני רוצה לומר תודה לכל העוסקים במלאכה, לכותבים החדשים והוותיקים, לסוקרים, לחברי העריכה, לחברים שדואגים להפיץ את הכתבות לקהילה ולכל אלו שעובדים ברקע ואינם נראים או מוזכרים בכתובים. העברנו את המסגרת האינטרנטית של המגזין לאתר חדש, עדכני ועכשווי יותר, המאפשר לנו להציע גישה נוחה ויעילה לכל הקוראים והקוראות. בנוסף, אני רוצה לנצל את ההזדמנות הזו ולהודות מעומק הלב לחברה יקרה, לאיריס קוטליצקי שהיתה העורכת הראשית של המגזין, דאגה לכל פרט ופרט, לכל כותב וכותבת, סוקר וסוקרת, לעורכים, למעצבים ולכך שהמגזין ימשיך ויפרח כל פעם מחדש, תודה רבה איריס ובהצלחה בכל אשר בחרת ותבחרי לעשות. אני רוצה לברך את ניצן גולדנברג שלקח על עצמו להוביל ולהחליף את איריס כעורך הראשי של המגזין, לאחל לו בהצלחה רבה ובטוחני כי ימשיך להוביל אותו בהצלחה מתמדת למול הקהילה.

לבד מהמגזין, אנחנו גאים גם במיטאפ TestIL שהקהילה פעילה בו באופן יוצא מן הכלל בשנים האחרונות ולנו יוצא ללוות אותו באמצעות החברים מהארגון ובסיוע כספי בהתאם לצרכים שמגדירים מובילי המיטאפ. ישר כח ליוזמי התכנים והמפגשים, למארחים מהחברות השונות ולחברי הקהילה שלוקחים חלק, משתפים בחוויות וידע מקצועי. אני רוצה להודות במיוחד לאיסי חזן, שהיה ממייסדי המיטאפ הראשון שצמח תחילה בירושלים ועם השנים הפך לארצי ולמוכר של הקהילה, וכמובן לשאר מובילי המיטאפ, קובי הלפרין, עמית ורטהיימר, שמואל גרשון וניצן גולדנברג. ישר כח חברים!

אם במיטאפ עסקינן, אי אפשר לדלג ולעבור לסדר היום מבלי לציין את הפעילות למול הקהילה ברשתות החברתיות, מקום בו ניתן לשאול שאלות ולקבל תשובות מהחברים בקהילה ובארגון באופן שוטף ומיידי. כישראלים, אנחנו מאד ישירים ולעיתים אף נוקבים במסרים שאנחנו משתפים ברשתות החברתיות השונות, וזהו תענוג אמיתי לראות איך אנחנו מאזנים אחד את השני ומוצאים את הדרכים לסייע ולהעביר בינינו מידע עכשווי, קוהרנטי ובעל משמעות; וגם אם אין הסכמה מלאה, יש שיתוף פעולה והערכה הדדית בין החברים. תודה לכל העוסקים במלאכה החשובה והדורשת, לחברים, קובי הלפרין, גיל דנון, ניצן גולדנברג ועוד שידי קצרה מלציין ועמכם הסליחה.

בשנים האחרונות היינו ערים לעליה בהכשרות של בודקי התוכנה, בין אם בתעשיה, בין אם במסגרות הצבאיות השונות ובעיקר במסגרת ההכשרות שמשרד הכלכלה מעודד באיזורים שונים בארץ ומול אוכלוסיות מיוחדות.

המעקב אחר היכולת של בוגרי הקורסים השונים להשתלב בעבודה בחברות השונות, הקשר עם הגוף הממשלתי (התמ"ת) ועם אנשי הקשר שלנו בתוכו, הגב' רחל בן-זקן, מר גדי יעקב ומר אלקיש סלימאן, הובילו אותנו לקבל החלטה לקיים מפגשי הכנה להשתלבות בתעשיה, שבמסגרתם יתקיימו סימולציות של ראיונות עבודה ע"י מגייסות מקצועיות, יועברו תכנים מעשיים ע"י וותיקים ומקצוענים מהתעשייה ועוד. יצאנו בפרסומים וההיענות הייתה גדולה מצד המועמדים. לצערנו, משבר הקורונה אילץ אותנו בשנית לדחות את הפעילות, אך אני שמח להודיעכם כי גם כאן, החלטנו בארגון יחד עם שותפים מהתעשייה, לקיים את המפגשים, אפילו באמצעים הווירטואליים הקיימים - שחלקנו התרגלנו אליהם או אף משתמשים בהם ביום-יום באופן כזה או אחר. בקרוב נוציא פרסומים בנושא ונעדכן על המתכונת הווירטואלית החדשה שהמפגשים יתקיימו עד שיתאפשר לנו לקיימם במסגרת התכנסות בנוכחות פיסית ולא רק וירטואלית.

אני תקווה שמיזמים ופעילויות מעשירות, חדשות ונוספות לעידוד המקצוע יצמחו בקרוב, שמיזמים ותיקים שהושהו יעלו מחדש כצורך מהקהילה, וכמובן שהפעילויות השונות שמתקיימות ערב לבקרים ימשיכו, יעלו ויצליחו בחודשים ובשנים הקרובות.

אסיים במילה אחת, תודה!

תודה לכם חברים בקהילה, ולקוראים היקרים, על ההזדמנות שאתם נותנים לנו לייצג אתכם ובתמיכה שאנחנו מקבלים מכם בכל הנוגע לקידום מקצוע הבדיקות בארץ ובעולם.

ישר כח!
שלכם,
ירון

ראיון עם מנהל בדיקות אלברט אלרום, investing.com

מראיינת

הייטקיסטית ואמא במשרה מלאה, בדקות הבודדות שנשארות ביום בלוגרית אפייה.
בעלת תואר ראשון במדעי המחשב ובכימיה, מעל 10 שנות ניסיון כמפתחת תשתיות אוטומציה וכלים אוטומטיים בחברות גדולות, בסטארטאפים שונים בתעשייה במגוון תחומים. מובילת תחום, מרצה ומפתחת קורסי אוטומציה. בשבילה החיים זה לא מספיק.

אלברט, ספר לנו קצת על עצמך

שמי אלברט אלרום, נשוי +3 ילדים, נמצא בתחום בדיקות התוכנה כ-20 שנה.

התחלתי את דרכי בתחום התוכנה לאחר קורס להסבת אקדמאים מהנדסאי תעשיה וניהול לתוכנה. מקום עבודתי הראשון היה בחברת CA שם עבדתי כחמש שנים, עד שהעבירו את תחום הבדיקות להודו. ב-2009 הגעתי לחברת סטארט-אפ קטנה בשם Wix לתפקיד ר"צ בדיקות צד לקוח (Client) וניהלתי צוות של שני בודקים. בהמשך גדלתי עם החברה, פיתחתי את מערך הבדיקות בסניפים שמעבר לים, גייסתי עשרות בודקים ובהמשך קיבלתי תפקיד של QA Guild Manager.

ניהלתי קבוצה של כ-50 אנשי בדיקות, ראשי צוותים ובודקים בסניפי Wix בארץ ובעולם. הייתי חלוץ בתחום של הסבת בודקים ידניים לבודקים "היברידיים" אשר מצד אחד בודקים באופן ידני כל פיצ'ר חדש ובהמשך כותבים לו אוטומציה באמצעות Selenium Web Driver.

פרשתי מ-Wix לאחר שמונה שנים טובות לטובת עסק עצמאי שהקמתי, חנות אינטרנטית למתנות ממותגות על הפלטפורמה של Wix. לאחר כשנה הרגשתי געגועים לעולם הבדיקות ובשנת 2018 חזרתי לתפקיד Head of QA בחברת Investing.com. שם הרמתי את צוות הבדיקות למחלקת בדיקות תוכנה עם גאוות יחידה, הקמתי צוות תשתיות אוטומציה ודאגתי להכשיר את הבודקים הידניים לבודקים "היברידיים" בדיוק כפי שעשיתי ב-Wix.

Investing.com היא פלטפורמת שווקים פיננסיים המספקת ניתוחים, מבזקים, כלים פיננסיים, גרפים, ציטוטים ונתונים בזמן אמת עבור למעלה מ-100 בורסות ברחבי העולם ב-30 מהדורות שונות. היא אחת מ-3 האתרים הפיננסיים הגלובליים הגדולים בעולם לפי SimilarWeb ו-Alexa, עם 20 מיליון משתמשים ויותר מ-170 מיליון הפעלות מדי חודש.

ממה מורכבת הקבוצה, במה היא עוסקת ואיך היא בנויה?

כאשר התחלתי לעבוד ב-investing.com קיבלתי על עצמי אתגר גדול - לבנות את מחלקת הבדיקות. בתחילת הדרך היו במחלקה כשלושה בודקים. היום ישנם חמישה בודקים היברידיים ועוד שבעה במחלקת האוטומציה.

כשהגעתי היה בודק בכל צוות. אני מאמין שצריכים להיות לפחות שני בודקים בכל תחום כדי לגבות אחד את השני במקרה הצורך ולכן דאגתי שיהיו לפחות שני בודקים בכל פלטפורמה. עבורי, חשוב מאוד להכניס את תחום האוטומציה לכל התחומים מכיוון שהיום כולם עוברים לקצב עבודה מהיר ויעיל וכאן למעשה פיתחתי את שיטת הבודק "ההיברידי".

אנחנו כרגע במעבר לשיטת עבודה אג'ילית שנקראת **SQUAD**, מדובר בשיטה שהיא תוצאה של שתי שיטות אג'ייל נפוצות: Kanban ו-Scrum. בשיטה זו אנחנו עובדים על ספרינט שבועי כאשר בסיומו מתחיל ספרינט חדש. יש זמן מוגדר לסיום הבדיקות ולא תמיד ניתן להספיק לבדוק את כל הפיצ'רים החדשים בגרסה ולכן, בסוף הספרינט מחכים לאישור ממנהל המוצר שבודק האם הפיצ'רים המשמעותיים עברו בדיקה, במידה וכן הספרינט הנוכחי מסתיים. הספרינט החדש יכלול המשך עבודה על פיצ'רים שלא הסתיימו בספרינט קודם ואליו יתווספו פיצ'רים נוספים. כמובן שפיצ'רים חדשים שלא עברו בדיקות לא ייכנסו לגרסה הנוכחית.

גרסאות בקבוצת המובייל:

כאשר רוצים להוציא גרסה אוספים פיצ'רים שעברו בדיקות בנפרד, בונים גירסת שחרור ומריצים רגרסיה מלאה כאשר רוב הבדיקות הן אוטומטיות בשילוב עם ידני. כמו כן מבצעים בדיקת שפיות קצרה לפיצ'רים החדשים על מנת לוודא שהם מתפקדים טוב יחד עם שאר הפיצ'רים החדשים האחרים בגרסה ובסוף התהליך מעלים לחנויות App Store ,Google Play.

גרסאות בקבוצת הדפדפנים:

בדומה לקבוצת המובייל עובדים בכל SQUAD על ספרינט שבועי. היות ואנו עובדים על פורטל חדש, כל פיצ'ר שעבר בדיקות נכנס לאתרי הדמו שלנו מבלי לעבור סדרה מלאה של בדיקות רגרסיה אלא בדיקות רגרסיה אוטומטיות בלבד שכרגע מכסות כ-48% מהמערכת.

הגענו למודל של הבודק ה"היברידי":
הבודק הידני שיודע גם לכתוב תסריטים אוטומטיים

ספר קצת על שיטת הבודק ההיברידי?

בתקופת עבודתי בחברת Wix, קבוצת הבדיקות הורכבה משני צוותים: צוות של בודקים ידניים וצוות של בודקי אוטומציה.

הצוות גדל ל-20 בודקים, כולל התרחבות לבודקים בחו"ל והחלה חלוקה לשלושה צוותים קטנים יותר תחתיי. במקביל אליי היה צוות של מפתחי אוטומציה שפיתחו את התשתיות עבור אנשי הבדיקות והם אלו שגם כתבו והריצו את הבדיקות על סמך התרחישים של הצוותים שלי. עם הזמן, למדנו איך לנתח את תוצאות ההרצות ולהבין האם הבאגים במוצר או שמקור הבאגים בתשתית האוטומציה. יום אחד פנו אלי שני בודקים שלי מאוקראינה, אמרו שהם יודעים ג'אווה וביקשו להצטרף לצוות האוטומציה. בנקודה הזו למעשה היה שינוי בתפיסה שלי.

התחלנו ניסיון עם שני הבודקים האוקראינים וראינו שזה עובד נהדר. לכל בודק (ידני) שהיה כותב אוטומציה הוצמד אדם מקביל מצוות האוטומציה ויחד הם עבדו על התכונות (הפיצ'רים) השונות. עם הזמן בודקים נוספים רצו לכתוב אוטומציה, אז שלחנו אותם לקורסי בדיקות אוטומציה ולהדרכות שונות והגענו למצב שכל הצוות (שכבר גדל והגיע קרוב לארבעים בודקים) יודע לכתוב בדיקות אוטומטיות. ככה הגענו למודל של הבודק ה"היברידי": הבודק הידני שיודע גם לכתוב תסריטים אוטומטיים. את ההצלחה הזו הבאתי איתי ל-investing.com. היום שלושה אנשים כבר כותבים אוטומציה ויש עוד שניים שכרגע בקורס. במקביל יש צוות שכותב את התשתיות, מתחזק את הפורטל ואת הג'ינקינס (Jenkins). מעבר לזה, ישנם עוד מתכנתים שמנגישים את האוטומציה לבודקים ויש את הבודקים עצמם שגם בודקים את התכונות וגם כותבים את הטסטים האוטומטיים.

מה האתגרים שלכם בתחום הבדיקות וכיצד אתם מתגברים עליהם, האם נראה כי אלו ישתנו בעתיד?

היום אנחנו מבינים שבעזרת סלניום ואפיום ניתן לחקות את פעולות המשתמש אבל האתגר שלנו הוא בתרחישים יותר מסובכים. אנחנו עובדים עם מכשירים אמיתיים וכדי שהבדיקות יתבצעו בצורה הטובה ביותר על המכשירים להיות "נקיים". בכל פעם שמתבצעת בדיקה, מותקנות תוכנות ויש רישום של המכשירים במסדי הנתונים. במצב כזה המכשירים למעשה "מתלכלכים" ואנחנו מגיעים למצב שהבדיקות כבר לא מדויקות. לכן אנחנו צריכים לעשות הרבה ניקיונות לפני כל בדיקה ומכיוון שהבדיקות רצות כמה פעמים ביום זהו אתגר רציני עבורנו.

ראיון עם מנהל בדיקות אלברט אלרום, investing.com

אתגר נוסף הוא להגיע ל-CI מושלם. זהו אתגר של כל מערכת הפיתוח. היום כל אחד יכול להריץ את הבדיקות אבל אנחנו רוצים שזה יהיה מחובר לפיתוח, כלומר, שהריצות האוטומטיות יתחילו לרוץ ברגע שהמפתח "דוחף" את הקוד שלו.

אתגר נוסף הוא לקבוע מי יתקן את הטסטים שנשברים בעקבות שינוי בקוד של המתכנת. אנחנו מנסים לחשוב מהי הדרך הנכונה ואנחנו בהחלט עובדים על זה.

אני בעיקר דואג להיות עם אנשים, זו מבחינתי ההגדרה של מנהל טוב

כיצד אתה מניע את הבודקים?

אני דואג לעשות עם כל בודק ישיבות אחד על אחד פעם בשבוע / שבועיים. אני משתדל להיות נגיש אליהם, להתעניין בהם ולהיות קרוב אל הבודקים ואל המפתחים, דבר שמאפשר לי לפתור בעיות בזמן. אני מאמין שאי אפשר לתת לעובד לעבוד כל הזמן כמו מכונה. חשוב לשמור על קשר בין הבודקים למפתחים.

אני מקפיד לוודא שכולם מרוצים מכולם ושהתהליך עובד לשני הצדדים.

נשמח שתשתף אותנו בהישג העיקרי של קבוצת הבדיקות

ההישג העיקרי היה המעבר לבדיקות אוטומטיות. ככה למעשה הצלחנו להגיע לכיסוי של למעלה מ-90% בכל המערכות שלנו ובכך שלושה שבועות של בדיקות הפכו ליום עבודה. בפלטפורמת הדפדפנים אנחנו עובדים על פורטל חדש לחברה ולזה אנחנו מתכננים לכתוב אוטומציה מלאה.

איך נראה יום העבודה שלך?

הכי כיף שיש!

אני קורא מיילים, משתדל להצטרף לישיבת בוקר של הקבוצות השונות של הבודקים והפיתוח.

מצטרף לפגישות על פיצ'רים חדשים ושולחנות עגולים.

אני מקפיד על פגישות אחד על אחד עם חברי הצוות שלי ועם צוותים אחרים.

דואג להיות במעקב מול קבוצת התמיכה ועוזר להם לשחזר באגים בעייתים במטרה להבינם ולאתרם בכדי לתקן אותם בהקדם על מנת שהלקוחות יהיו שבעי רצון.

אני תמיד מוודא שאין בעיות של כוח אדם בפרויקטים השונים ועוזר למנהלי פרויקטים להשיג את כוח האדם הנדרש למשימה.

כרגע אני גם כותב מסמכי בדיקות על מנת שקבוצת האוטומציה תוכל לכתוב עבורו את התשתיות המתאימות.

אני בעיקר דואג להיות עם אנשים. זו מבחינתי ההגדרה של מנהל טוב. עם הזמן נחשפתי לכלי אוטומציה מתקדמים כדוגמת סלניום וובדרייבר.

מה הן ההמלצות שלך לבודקים ומנהלים בתחום?

לא להישאר במקום. מתודולוגיות הולכות ומשתנות, לא תמיד זה נכון ומצליח אבל זה בהחלט חשוב להיות מעודכן עם מה שקורה סביבנו ולעבוד בצורה נכונה. אני ממליץ לקרוא הרבה, לראות סרטונים, לשתף מידע עם חברים נוספים בתחום ולראות לאן ואיך העולם הזה מתקדם. אצלי למשל, כולם רשומים לקבוצת המיטאפ של TestIl.

אני באופן אישי מאמין גדול בשיטה של Rapid Testing - שיטה של עבודה מהירה ועבודה בחלקי טסטים קטנים. בשיטה זו התרחישים מוגדרים כצ'ק בוקסים.

בעבר הבודק היה כותב את מסמכי הבדיקות מראש וברגע שהגרסה הייתה יוצאת הוא היה עובר על כל הטסטים הכתובים מראש ובודק לפיהם עד לסיומם. בשיטת Rapid Testing הבודק מכין מסמכים כלליים בלבד וכך נשאר לו למעשה מרחב גדול לבדיקת המוצר ושבירתו. במצב הזה הוא צריך להפעיל את הראש ואת כל החושים שלו שכן הוא אינו מוגבל לטסט שכבר נכתב. לכן זה מאפשר לו מקום להגדלת הראש של הבודק במקום קיבעון וככה הבודק מצליח להביא את יכולות הבדיקה שלו לידי ביטוי.

המלצה חשובה שלי היא להעריך את הבודקים לפי "מקרי התמיכה" (support cases), ככה אני יכול לבדוק הכי טוב איפה הפספוסים שלנו. אם בעבר היינו סופרים באגים ובודק שהיה מוצא סביב ה-60-70 באגים היה נחשב בודק מצטיין הרי שהיום צורת הבדיקות השתנתה. היום, ברוב המקרים הבודקים יושבים בתוך צוותי הפיתוח, רחוקים מראש הצוות ומהשגחתו ובהתאם לזאת גם צורת הערכתם השתנתה. הערכת הבודק מתבצעת בישיבות עם צוותי הפיתוח אך דבר זה אינו מספיק ויש צורך גם לבדוק את איכות הבדיקות של אותו בודק. במקרים רבים ניתן לקבל מידע ממחלקת התמיכה. ניתן לקבל מידע על איכות העבודה של הבודק על ידי מעקב אחרי הבאגים שמגיעים מהלקוחות בשטח ודווחו על הפיצ'ר עליו אותו בודק היה אחראי.

אני מאמין שאי אפשר לתת לעובד לעבוד כל הזמן כמו מכונה, חשוב לשמור על קשר בין הבודקים למפתחים

מה היית ממליץ לבודק תוכנה שנמצא בתחילת הקריירה שלו?

קודם כל להבין שחשוב להתפתח וכדי לעשות זאת חשוב לעבוד במקום שנותן לך אפשרות לשלב בין בדיקות ידניות לאוטומציה. חשוב לעבוד עם כלי בדיקות מוכרים בשוק, ללמוד שפת תכנות אחת לפחות ולהכיר מתודולוגיות שונות של בדיקות.

כאשר אני מגייס אנשים אני מחפש אנשים שאוהבים ורוצים להתפתח בתחום הבדיקות ושמוצאים עניין בתחום הזה. אני מחפש "כוכבים", אנשים שלוקחים אחריות ומגדילים ראש.

ממה היית ממליץ להימנע?

הרבה פעמים מגיעים אל קבוצת הבדיקות עם טענות שונות לגבי ביצוע הבדיקות והבאגים שעולים. מבחינתי זו טעות ישר להתגונן ולענות. במקום זה, צריך לקחת רגע לנשום ולנסות להבין מה קרה. הרבה פעמים קורה שהתרחיש נבדק והכל בסדר אבל רגע לפני שעלתה הגרסה, מתכנת הוסיף תיקון. אל תשלפו תשובה מהבטן - שבו, חיקרו ונסו להבין את מקור הבעיה.

ראיון עם מנהל בדיקות
אלברט אלרום, investing.com

פספורט קבוצתי – Investing.com

- **סוג המוצר הנבדק**
  פורטל פיננסי על פלטפורמת דפדפן, יישומי סלולר iOS & Android
- **גודל הקבוצה**
  5 בודקי תוכנה "היברידיים" העוסקים בבדיקות ידניות ואוטומטיות,
  7 אנשי אוטומציה, ר"צ תשתיות, מתכנתי Java קבועים וסטודנטים
- **וותק הבודקים**
  שילוב של בודקים ותיקים וגם בודקים חסרי ניסיון אחרי קורס
- **מבנה הקבוצה**
  צוות של שני בודקים לקבוצת הדפדפנים ושני בודקים לקבוצת הסלולר
  - ✔ בודקת אחת לקבוצת Revenue
  - ✔ צוות של תשתיות אוטומציה
  - ✔ מתכנתי אוטומציה קבועים
  - ✔ מספר סטודנטים שכותבים טסטים אוטומטיים
- **סוגי בדיקות**
  אנו מבצעים בדיקות ידניות בתחומי פונקציונליות ו-UI לפיצ'רים חדשים וכן בדיקות רגרסיה אוטומטיות וידניות
- **שיטת עבודה**
  - ✔ Rapid Testing במקום טסטים ארוכים משתמשים בצ'ק ליסט (בדיקה פוזיטיבית אחת שהפיצ'ר החדש מתנהג ע"פ התיכנון ובהמשך ניסיונות לשבור את הפיצ'ר ע"י בדיקות שליליות)
  - ✔ רגרסיה מסוג Golden Flow (בדיקה חיובית של 100% מהמסלול שימוש של 80% מהמשתמשים)
  - ✔ רגרסיה מסוג Silver Flows (עבור מסלולים חשובים נוספים אותם יבצעו המשתמשים במוצר)
  - ✔ ספרינט של שבוע בקבוצות Squad
- **כמות המוצרים הנבדקים וקצב שחרור גרסאות**
  - ✔ בקבוצת הדפדפנים - פיצ'ר נבדק ומיד עולה להפקה (Production)
  - ✔ בקבוצות הסלולר קצב השחרור הוא אחת לשלושה שבועות ובשאיפה לקצר לגרסה אחת לשבועיים

בחן את עצמך | טל פאר

כלל ידוע בבדיקות הוא שלא ניתן לבדוק הכל ועלינו לבחור את מקרי הבדיקה (test cases) שיתנו לנו את הערך המוסף הגבוה ביותר, כלומר אלה שימצאו פגמים (defects) משמעותיים במערכת או ידגימו שתכונה מסוימת של המערכת עובדת בצורה נכונה. כדי לבחור את מקרי הבדיקה האלה, על הבודק להשתמש בטכניקות בדיקה שונות. לדוגמה, באמצעות טכניקת ניתוח ערכי גבול (boundary value analysis) נתכנן בדיקות שיוודאו ששדות קלט מקבלים רק את הערכים הנכונים, לפי מפרט (specification) המערכת.

אולם לא תמיד ישנם מפרטים למערכת ולא תמיד המפרטים מספיקים. לשם כך אנו משתמשים בטכניקות בדיקה מקבוצת הבדיקות מבוססות ניסיון (experience based techniques). טכניקות אלה מתעלות את הניסיון והידע של הבודק כדי לעצב בדיקות בצורה יצירתית יותר.

בסילבוס של ISTQB® לרמת הבסיס (CTFL) מוצגות מספר טכניקות בדיקה מבוססות ניסיון ועל אחת מהן השאלה שלנו היום.

איזה מהמשפטים הבאים מתאר את טכניקת ניחוש שגיאות (error guessing) בצורה הטובה ביותר?

א. בטכניקה זו מנסה הבודק לשחזר שגיאות שעשה המפתח.

ב. טכניקה זו כוללת שימוש בידע של הבודק על כשלים שהופיעו בעבר במערכת הנבדקת ובמערכות דומות.

ג. טכניקה זו כוללת ניחוש של שגיאות שמשתמש יכול לעשות תוך שימוש במערכת.

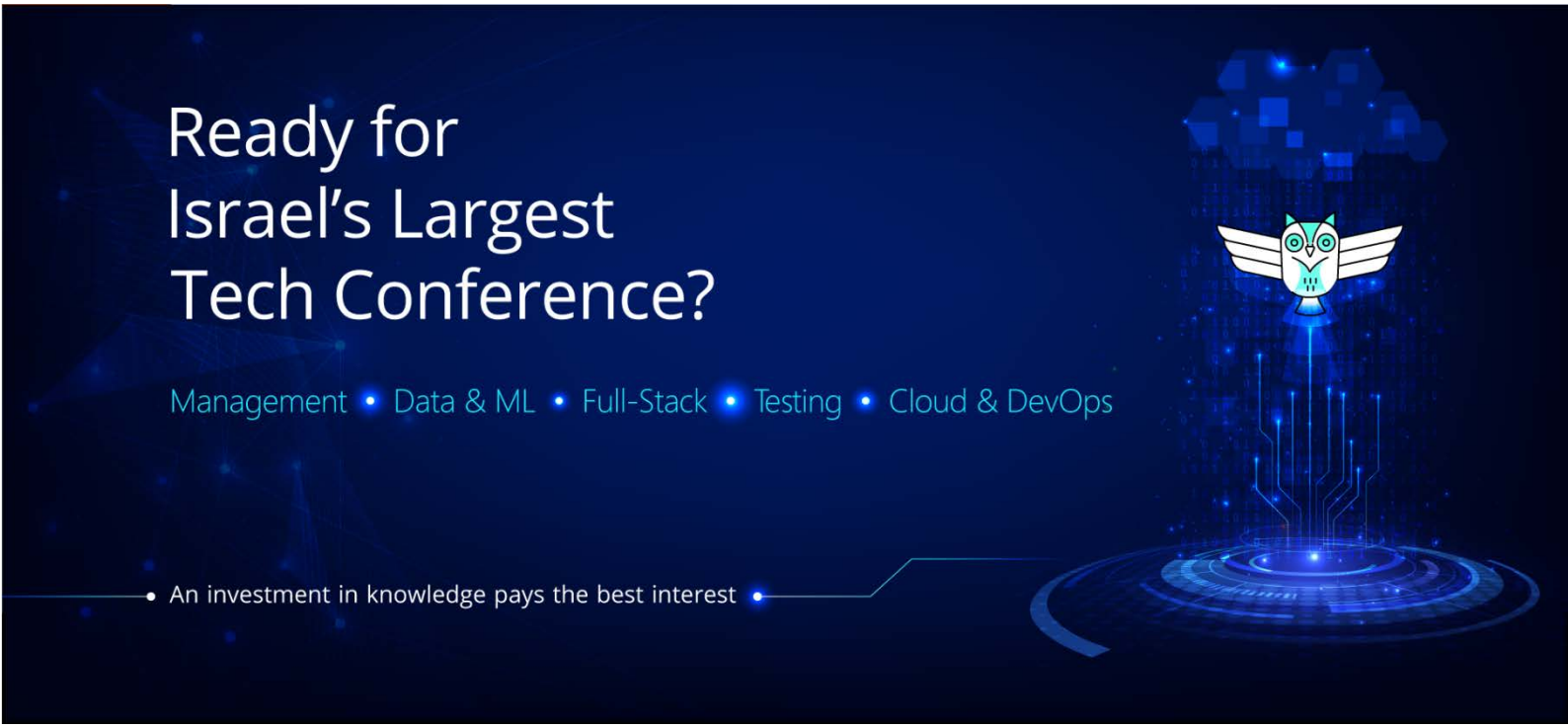
ד. בטכניקה זו מנסה הבודק לכתוב קוד מהיר כדי לראות איזה שגיאות יכול מפתח לעשות.

*לצפייה בתשובה המפורטת- דפדפו לעמוד 13

טל פאר

בעל ניסיון של יותר מ-20 שנים כבודק ומנהל בדיקות במגוון חברות וטכנולוגיות במודלי פיתוח שונים. כיום טל יועץ ומדריך בדיקות עצמאי.

טל חבר ב-ITCB® וגזבר הארגון העולמי ISTQB®.



מי לא אוהב לשחק במשחקים?

מאז שהיינו ילדים, כל מה שעניין אותנו זה להכיר משחקים חדשים, לשחק, להנות ובעיקר... לנצח! שיחקנו משחקים כמו מחבואים, "תופס את" (או בשמו היותר ידוע - תופסת), משחקי דמיון שונים, משחקי קופסה או משחקי מחשב מאתגרים, לשחק - זה תמיד כיף! למעשה, מחקרים מוכיחים שילדים מתפתחים על ידי משחק. חוקרים פרסמו מאמרים רבים המסבירים את החשיבות של המשחק להתפתחות תקינה של ילדים.

עולם המשחקים התפתח מאוד בשנים האחרונות והביא למצב שהשחקנים דורשים לשחק במשחקים מעוצבים, מהירים, מאתגרים, ובעיקר מספקים יותר

המשחק מפתח את המוח שלנו. נו.. בדרך כלל.
מצב דומה קיים גם אצל אנשים מבוגרים:
ניקח לדוגמה את Candy Crush, משחק פשוט כביכול.
אם נתבונן על הערך המוסף שנותן המשחק, נגלה שישנה משחקיות מסוימת, שיכולה לעזור לפתח את מהירות התגובה שלנו, לחדד את יכולות ההכללה ועוד פעולות מחשבתיות שיתרמו לפיתוח המוח שלנו וזאת בעקבות חישובים לוגיים מסוימים שהשחקן צריך לבצע בזמן נתון. כמובן שמטרת המשחק היא הנאה פשוטה, אך תוך כדי המשחק, אנו נדרשים להפעיל יכולות מסוימות התורמות לחידוד ושיפור החשיבה.

עם התפתחות הטכנולוגיה, המחשוב וכמובן עולם הסמארטפונים, עלתה הדרישה לייצור משחקים לכלל הפלטפורמות המקובלות בכדי לאפשר לנו ליהנות בכל רגע ובכל מקום.
היום כבר לא צריך להגיע הביתה ולהדליק את הפלייסטיישן (או את המגסון לילידי שנות ה-80) כדי להתחיל להנות, היום אפשר להגניב משחקון גם כשהולכים להתפנות בשירותים. אני לא עושה את זה כמובן.

עולם המשחקים התפתח מאוד בשנים האחרונות והביא למצב שהשחקנים דורשים לשחק במשחקים מעוצבים, מהירים, מאתגרים, ובעיקר מספקים יותר. המובילות הגדולות של התחום הזה הן כמובן חברות המשחקים, שמפתחות משחקים מדהימים כל פעם מחדש ובכך שוברות את תקרת הזכוכית של גירוי המשתמשים ע"י יצירת חוויה מרגשת, אמיתית ומהנה יותר.
החברות מייצרות אפליקציות איכותיות יותר התואמות את ציפיות המשתמשים ואף עוקפות אותן וכמובן מייצרות הכנסות לא רעות בכלל.

שומרי הסף של המוצרים הללו הם לא אחרים מאשר בודקי המשחקים. מיד לאחר יישום הרעיון לפיתוח של משחק מסוים יכנסו הבודקים לתמונה, בין אם זה בדיקה של משחק שלם או רק חלק מהמשחק שהבשיל לכדי בדיקות. תהליך הבדיקות יהיה מורכב ומאתגר, יכלול בתוכו המון דברים חשובים שפשוט צריך לבדוק ולוודא שפועלים על פי הרעיון המקורי ויבטיח שהחוויה שנוצרה היא ייחודית וקולעת למטרה.

סוגי הבדיקות

אז בואו נצלול קצת..

בשוק המשחקים קיים מגוון רחב של אפשרויות, ישנם משחקים גדולים ומורכבים עם עלילה דינמית שמתפתחת, תוכן ומשחקיות מורכבת. לעומתם ישנם משחקים פשוטים, רזים, כאשר תכולת הבדיקות שלהם תהיה נמוכה משמעותית מאפליקציות גדולות.
כמובן שהבדיקות למשחקים השונים צריכות להיות מותאמות לגודל האפליקציה ויידרש לנהל נכון את המשאבים של צוות הבדיקות כדי למקסם את היעילות של כיסוי הדרישות.

בדיקות פונקציונליות

הבדיקות העיקריות של המשחק שלנו נדרשות לוודא שמבחינה פונקציונלית, הכל עובד כשורה. בשלב הזה נדרש לכתוב ולהריץ כמה שיותר בדיקות, משום שזהו האזור המרכזי של האפליקציה, שעליה מתבססת חווית המשחק של המשתמש, ומכאן באופן ישיר - ההכנסות של האפליקציה. נשאף להבטיח שהאפליקציה שלנו מתפקדת בצורה טובה, ברורה, מהנה ובעיקר שתענה על הדרישות כפי שאופיינו.
הבדיקות הפונקציונליות מתחילות מתהליך ההתחברות לאפליקציה, הרשמה (Login), והדגמות ראשוניות לשחקן (Tutorials).
אם הגדרנו התנהגות מסוימת למשתמשים חדשים, זה המקום לכסות את דרישות התהליך. בזמן פעילות המשחק נרצה לוודא שכל הכפתורים באפליקציה פעילים ולחיצים (או שאינם פעילים - לפי הדרישה) וכמובן שמבצעים את הפעולות שלהם (actions) כגון, פתיחת מסכים נוספים, איסוף בונוסים או כל פעולה מוגדרת אחרת. נבדוק שכל מסכי הטעינה (loading screens), בר הטעינה או אנימציות המעבר עובדות כראוי. אם הוגדרה מוזיקת רקע במסכי הפתיחה, במשחק עצמו או בלחיצה על כפתור מסוים, נדרש לוודא שהם אכן עובדים וכמובן שניתן לכבות/להשתיק אותם.

לאחר מכן נבדוק את החלק העיקרי של המשחק:
בדרך כלל מדובר על אובייקט שהמשתמש צריך לנהל, בין אם זה לחיצה על האובייקט, הזזת האובייקט, החלקה, שינוי כיוון האובייקט ובדיקה שהוא לא חורג ממסלולו, פעולה שהאובייקט יכול לבצע, תנועה במרחב או כל פעולה שמתאפשרת לשחקן לבצע על האובייקט וכל שימוש אפשרי באובייקטים שמתווספים לסצנה.

כמובן שנדרש להוסיף את כל הבדיקות האפשריות הקשורות למשחקיות עצמה - כל משחק על פי הפונקציונליות הפנימית שלו. חשוב מאוד לציין שוב שהתפקיד של הבדיקות הן לוודא שהכל פועל כפי שאופיין, על פי דרישות מנהלי המוצר ומעצבי המשחק משחקים רבים משתמשים במעיין מטבע פנימי, כוכבים, יהלומים, מטבעות וכדומה, חשוב מאוד לבדוק שעבור פעולות מסוימות אנו מזכים/מחייבים את השחקן בסכום הנדרש. בדיקות אלו יכולות להיות מכוסות בצד המשתמש, השרת וכמובן בדיקות ברמת מסד הנתונים (DB Validation).

בנוסף, יש המון משחקים המשלבים בתוכם עולם שלם של תוכן מסביב למשחק עצמו, כלומר שאינו קשור לפונקציונליות המשחקית העיקרית: משחקי בונוס, עולמות / שלבים / תכונות חדשות שנפתחות לשחקן בהתאם להתקדמותו במשחק, קמפיינים פרסומיים ממוקדים, כל אלו צריכים להיבדק.

אופיר מור

בן 34, נשוי 3+

בעל רקע של 8 שנות נסיון בבדיקות תוכנה.

כיום מפתח משחקים ב- SciPlay.

בדיקות משחקים | אופיר מור

במשחקים מבוססי משתמש/שרת (C/S) ישנה יכולת לשנות מרחוק את התנהגות המשחק, הגדרות המשחק ועוד הרבה אפשרויות מעניינות שצריך לכסות בבדיקות. ביניהם נדרש לבדוק שהמידע המעודכן מופיע במשחק, שניתן לבצע עדכונים מרחוק על פי דרישות המוצר, ושהמשחק מתנהג כראוי גם כאשר משנים הגדרות לשחקנים בזמן שהם בסשן המשחק. בדיקה חשובה נוספת במשחקים שבנויים בארכיטקטורה זו היא בדיקת עדכוני שרת. כאשר אנו מעלים שרת חדש נהיה חייבים לבדוק שכל גרסאות הלקוח שבאוויר עדיין עובדות בצורה תקינה וכל הדברים שהתווספו או הוסרו מקוד השרת לא יפגעו בגרסאות הקיימות של הלקוח.

הידעתם..?
כיום, זמן ההמתנה הממוצע של גולשים לטעינת עמוד אינטרנט, ירד ל... 3 שניות בלבד

בדיקות ממשק משתמש

וואו... כמות ההשקעה הנדרשת כדי לעצב משחק היא פשוט אדירה.
החל ממעצבים, מומחי חוויית משתמש (UX), מנהלי מוצר ועוד הרבה מקבלי החלטות ששוקדים במרץ כדי להגיע לתוצאה המדהימה, המעניינת והמושכת ביותר שניתן.
בדיקות ממשק משתמש (UI) הן סופר חשובות להצלחת המוצר משום שבאמצעותן אנו בודקים למעשה את כל מה שהמשתמשים שלנו רואים וחווים. למעשה החלק העיקרי של הבדיקות הקשורות לחווית המשתמש הן בדיקות ממשק המשתמש, מכאן נבין את חשיבותן. הבדיקות באזור זה יתחילו ממסך הטעינה של המשחק (Splash Screen), ועד לפופ-אפ שמוודא שהשחקן בטוח רוצה לעזוב.

הבדיקות העיקריות יכללו:

בדיקות פקדים, כותרות, מסכים, אנימציות ולמעשה - כל דבר שאמור להופיע במשחק שלנו. שלל גירויים גרפיים מושכים את תשומת לב השחקן שלנו ואנו נהיה חייבים לוודא שהם תואמים לעיצוב המתבקש וכמובן שהם נראים טוב ומתפקדים טוב.

בדיקות אלו יכללו גם טעינת תמונות, פונטים וטקסטים דינמיים ומיקומם התקין על מגוון מכשירים. לגבי טקסטים, נוודא שאין שגיאות כתיב, שהטקסט הוזן בצורה תקינה ומופיע כראוי. בדיקה נוספת וחשובה לא פחות, היא בדיקה שלילית - כאשר האובייקט שניסינו לטעון נכשל ואובייקט ברירת המחדל נטען ומוצג במקומו.

לגבי אנימציות, לפעמים קשה לוודא שהמראה בפועל של אנימציה מסוימת - תואם לאפיון ולדרישות, לכן במקרה כזה, תמיד עדיף לפנות ישירות למקבלי ההחלטות בעניין, להציג להם את התוצאה על המכשיר ולקבל את אישורם.

כדי לנסות ולהגיע להתנהגות דומה ככל הניתן של השחקן נדרש לבצע בדיקות שימושיות (usability) שאמורות לכסות את כל האפשרויות שאנו מציעים לשחקן לבצע, בלי לצפות ממנו להתנהגות "הנכונה" - לחיצה על כפתורים שלא אמורים ללחוץ עליהם (לא זמינים / לחיצים), פעולות לאחר המתנה ארוכה, לחיצות כפולות וכו'. ניתן גם לבצע שימוש בחברות חיצוניות כדי להשלים את תהליך הבדיקות על ידי משתמשים אמיתיים.

בדיקות ביצועים

בדיקות ביצועים הן אחד מסוגי הבדיקות החשובים והמשפיעים ביותר וזאת משום שביצועי המשחק שלנו יעידו בדרך כלל, על איכות ויעילות הקוד שעומד מאחוריו. משתמשים מרגישים בביצועי המשחק שלנו כבר בשלב מאוד מוקדם של המשחק, מיד מהפעלתו ולכן נרצה לדאוג לחוויה טובה ומהירה ככל הניתן. הידעתם..? כיום, זמן ההמתנה הממוצע של גולשים לטעינת עמוד אינטרנט, ירד ל... 3 שניות בלבד.

בעידן המהיר של היום, משחקים קמים ונופלים על ביצועים, ישנם משחקים מסוימים, שמהירות התגובה של השחקן יכולה להכריע מערכה שלמה (ע"ע Clash Royal) ולכן לא נרצה שהמשחק שלנו יגמגם בזמן שהשחקן חייב להגיב במהירות. באופן כללי, שחקנים היום התרגלו למשחקים מהירים מאוד, שמצפים לתגובה שלהם ולא להפך.

כמובן שבדיקות הביצועים שלנו יהיו מותאמות לסוג המשחק, אך בכל אופן נכסה את המקרים הבאים:

- הפעלה ראשונית מהירה של המשחק
- בהנחה שהשחקן נדרש להוריד דברים מסוימים בהרצה הראשונה - מומלץ לבדוק את שאר ההפעלות (לאחר הטעינה הראשונית) שאמורות להיות מהירות יותר
- תנועה חלקה, מהירה, זורמת ותקינה של כל חלקי המשחק שלנו
- טעינה מהירה של אובייקטים גרפיים (סצנות, שלבים, פופאפים) ומעבר מהיר בין מסכים
- בדיקות ביצועים נוספות בהתאם לסוג המשחק (לדוגמה שימוש במצלמה, בג'יירו, אקסלרומטר וכו').

בדיקה נפוצה נוספת הינה מעקב על ביצועי האפליקציה בזמן שאנימציות מופעלות, משום שאנימציות מטבען לוקחות משאבים רבים מזכרון המכשיר, ישנם מכשירים שעלולים לסבול מתגובה איטית ולא חלקה של האנימציה, דבר שגורם לתחושה לא נעימה וזורמת של המשחק. בנוסף, נדרש לעתים להוסיף בדיקות כלליות על מכשירים ישנים ואיטיים אף יותר כדי לוודא שלא הגענו לתקרת הזיכרון הזמין של המכשיר, דבר שיכול להיות מורגש בטעינה איטית של תמונות, זמני מעבר ארוכים בין מסכים, ואיטיות כללית של האפליקציה. בבדיקות ביצועים יש כלל - מה שעובד טוב, נראה טוב ומתנהג טוב על מכשירים ישנים, כנראה יראה עוד יותר טוב על מכשירים מהשורה הראשונה.

דבר נוסף שצריך לקחת בחשבון הוא שאפליקציות נוספות פועלות ברקע וצורכות זיכרון מהמכשיר, ולכן נרצה להשאיר טווח ביצוע סביר בהתחשב בנתונים אלה.

ניתן לשקול אמצעים נוספים להשלמת הבדיקות: להעזר במחלקת הפיתוח (ייתכן שקיימות שיטות נוספות לבדיקות הביצועים ברמת הפיתוח), שימוש בכלי בדיקות ביצועים מותאמים על פי הצורך, שימוש בחברות חיצוניות וכו'.

בדיקות ניטור/אנליטיקס

בימינו, יש דבר אחד שכמעט כל חברות התוכנה כבר הבינו שהוא החשוב ביותר וזה מידע. כמה שיותר מידע על המשתמש, על הרגלי הצריכה וההתנהגות שלו - הרי זה משובח. חברות מסוימות כמו גוגל מצהירות באופן גלוי שהן עוקבות אחרי המשתמשים שלהן ועושות שימוש במידע שנאסף (לרוב עבור פרסום יעיל ומפולח).

חברות המשחקים מבצעות בדיוק את אותו הדבר - מבצעות מעקב אחרי כל הפעולות של המשתמש ואוספות מידע כללי על האפליקציה כל זאת בכדי ליצור משחק מהנה יותר, להגיע לחוויית משתמש טובה יותר וכתוצאה ישירה מכך - להגדיל את ההכנסות.

איסוף המידע יכלול לדוגמה את כמות ההתקנות של המשחק (Installs), כמות הרכישות, זמני המשחק (Session), המסכים שבהם המשתמש עבר, הפרטים שראה ועד לרמת דיווח הכפתור עליו בחר המשתמש ללחוץ מתוך כמה אפשרויות שניתנו לו.

כל פעולה נשלחת בזמן אמת למערכות שאוספות את המידע ומספקות פלטפורמה נוחה לאנליסטים / מנתחי המידע לבדיקה וניטור התנהגות ואופי המשתמשים. המידע שנשלח חייב להיות מדויק, משום שהחלטות גורליות יכולות להתקבל על סמך נתונים אלו. מכאן ניתן להסיק את חשיבות הבדיקות של כל פיסת מידע הנשלחת לצורכי בקרה. הבדיקות חייבות להבטיח שהמידע נשלח בזמן הנכון, שאין כפל של שליחות וכמובן... שהמידע תקין.

בדיקות משחקים | אופיר מור

בדיקות תאימות

האתגר הגדול ביותר של חברות המשחקים הוא להגיע לכמה שיותר משתמשים שישחקו באפליקציות שלהן, כאשר תמיכה בריבוי המכשירים והפלטפורמות מהווה אתגר משמעותי עבורן.ישנו מגוון עצום של מכשירים בשוק שמסוגלים להריץ משחקים שונים וחברות המשחקים שואפות להגיע לרמה שווה ככל הניתן של חווית משחק בכל המכשירים האפשריים, החל ממכשירים חדשים ולא מוכרים של חברה סינית ועד למכשירים היקרים והמהירים ביותר בעלי כמות מרשימה של זכרון זמין וכח עיבוד.האתגר להתאים את המשחק עבור כולם הופך להיות קשה הרבה יותר, כאשר חברות המשחקים צריכות ליישר קו גם עם כמות הרזולוציות השונות שישנן בשוק המכשירים.

בדיקות תאימות יכללו בין היתר בדיקה על מגוון מכשירים וטאבלטים, עד לרמת הכיסוי האפשרית:

בדיקות מיקום הטקסטים, התמונות, הכפתורים וכל אובייקט נוסף שעלול לחרוג מהמסך או להופיע בצורה לא תקינה כתוצאה מגודל מסך לא נתמך. בדיקות על מגוון מערכות הפעלה, דפדפנים ופלטפורמות נתמכות. בנוסף, כחלק מתהליך הבדיקות ניתן לשכור את שירותיהן של חברות חיצוניות שנותנות מענה למגוון רחב יותר של מכשירים. כל זאת, שוב, כדי לוודא שחווית המשחק תהיה טובה ככל האפשר עבור כלל המשתמשים בכל העולם.

בדיקות סלולר

בעולם הסלולר ספציפית, ישנן בדיקות נוספות שנדרש לבצע כדי לדמות את המצב האמיתי של המשחק שלנו, במידה מרובה ככל שניתן:

קבלת שיחות, הודעות, כיבוי הטלפון (למי לא נגמרה הסוללה באמצע המשחק...) וכו' הן רק חלק מהבדיקות שאסור לפספס.
היום משתמשים נכנסים לאפליקציות כמה פעמים ביום ולכן תהליך החזרה למשחק חייב להיבדק, בין אם זה חיבור מחדש (Re-login) וחזרה לאותו מקום בדיוק שהמשתמש עזב, כמובן, על פי ההתנהגות הרצויה שאופיינה.

עדכוני גרסה- עדכוני גרסה הם דבר שבשגרה, ישנם משחקים שמאלצים לשדרג את גרסת המשחק (Force Update) וישנם שמעדיפים לעשות זאת בגדר המלצה, בכל אופן, תפקידנו יהיה לסמלץ שדרוג שכזה מגרסאות קודמות לגרסה החדשה ולוודא שהכל עובר כשורה.

בימינו, יש דבר אחד שכמעט כל חברות התוכנה כבר הבינו שהוא החשוב ביותר וזה מידע

התראות – משחקים רבים בוחרים להשתמש בשרותי התראות כדי לפתות את השחקן לחזור ולשחק במשחק שלנו (engagement), על ידי מתן בונוסים, משימות לזמן קצוב או סתם לתזכר ולשמור על הגחלת חמה. בדיקות אלה יכללו את קבלת ההתראות בפלטפורמות השונות, הטקסט המופיע בהתראה, העיצוב ותדירות ההתראות, בנוסף - אם ישנה פונקציונליות מסוימת בלחיצה על ההתראה נרצה לוודא את תקינותה.

משחק במצב מאוזן\מאונך - רוב המשחקים בוחרים להציג את המשחק שלהם בצורה מסוימת לרוחב (horizontal) או לאורך (vertical), אך יש משחקים שמשלבים את אופן התצוגה שמופיעה על המכשיר בהתאם להתנהגות \ בחירת השחקן, במצב כזה נרצה לוודא שהמשחק נראה טוב בשני המצבים ושהפונקציונליות לא נפגעת, כמו כן נרצה לוודא שהמעבר בין המצבים נראה טוב.

בדיקות רכישה

תכלס, בדיקות רכישה הן הבדיקות הכי חשובות במשחק. בבדיקות אלו נבטיח שמנגנון התשלום עובד כראוי ושהחבר'ה מלמעלה יהיו מרוצים. אז למה הן מופיעות רק לקראת סוף המאמר?
א'- כי הן די פשוטות.
ב'- אני מאמין, שתחילה ראוי לוודא שהאפליקציה שלנו מתפקדת בצורה מספקת ומהנה ומבטיחה חוויה ששווה לשלם עבורה.

בבדיקות רכישה נפוץ מאוד לבצע רכישות אמיתיות, ולבדוק את הצלחת הרכישה מול גוגל, פייסבוק, אפל, פייפאל, או כל פלטפורמת תשלום אפשרית שהמשחק שלנו מציע.
כמו כן מומלץ לערוך בדיקות רכישה מדגמיות בכל גרסה חדשה שמעלים לאוויר.

ישנם משחקים שמודל הכנסות שלהם מבוסס פרסומות, במשחקים שמטמיעים שרותים מסוג זה נרצה לוודא שהפרסומות מופעלות בהתאם לדרישות, בתדירות הנכונה, בתזמון הנכון, שהפונקציונליות של הפרסומת תקינה ושאם יש בעיה בטעינת הפרסומות - חווית המשתמש שלנו לא תפגע.

סיכום

אנחנו אוהבים לשחק ואפילו יותר אוהבים לשחק במשחקים שאנו מרגישים שהושקעה מחשבה רבה בתהליך התכנון, העיצוב, הפיתוח והבדיקות שלהם!

כמו שאנו יודעים, בדיקות הן חלק בלתי נפרד מתהליך הפיתוח, אך עם זאת - אין מוצר שחף מתקלות. גם במאמר זה (שכתבתי בהנאה גדולה) כנראה פספסתי דבר או שניים.

אבל היי, אני מקווה שהיה פה כיסוי לא רע..!

גישת ניחוש שגיאות

בבדיקות תוכנה אנו חותרים תמיד להגיע לכיסוי מקסימלי של דרישות הלקוח. בדרך להשגת המטרה אנו מכירים את הטכניקות הידועות שכוללות:

- טכניקות קופסה שחורה המבוססות על המפרטים הטכניים (Specification based)
- טכניקות קופסה לבנה המבוססות על מבנה הקוד של התוכנה (Structure based)
- טכניקה מבוססת ניסיון (Experienced Based) שנחשבת גם לטכניקת קופסה שחורה כשאחת השיטות המוכרות בה היא **בדיקות המבוססות על ניחוש שגיאות**

מהי השיטה?

שיטה זו מבוססת למעשה על ניחוש אינטליגנטי של מיקומי תקלות אפשריות שניתנות לחיזוי.

ה"ניחוש" מתבסס על היכרות מוקדמת עם המערכת או היכרות עם מערכות ממוחשבות דומות.

יכולת הניחוש נרכשת בהתאם לפרמטרים הבאים:

- היסטוריה של תקלות מגרסאות קודמות
- ניסיון של צוות הבדיקות במערכות דומות אחרות
- ניסיון של צוות הבדיקות כמשתמשי קצה של המוצר
- בעיות רגרסיה שנצפו בעבר ועלולות להופיע שוב
- בעיות ידועות בתשתית, בחומרה או בקוד של המוצר שניתן לנחש את הופעתם שוב
- הבנה לגבי הפסיכולוגיה של משתמשי הקצה ומעקב אחר צורת החשיבה שלהם

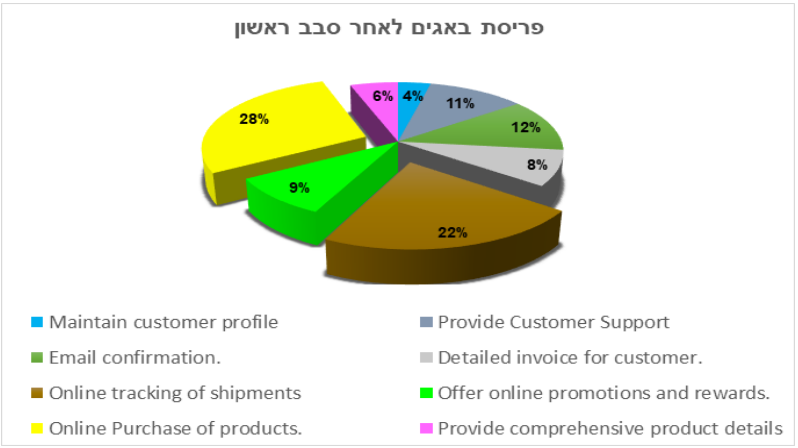
כל אלו יסייעו להשתמש בשיטה, להצביע על מקומות בעייתים ובדרך לאפשר גם לצאת מפה לתכנון הבדיקות (STP).

קובי יונסי

בעל תואר ראשון בלוגיסטיקה וכלכלה מאונ' בר אילן. מרצה ומכשיר בודקי תוכנה כבר למעלה מעשור. מוביל את תחום בדיקות התוכנה במספר מוסדות ביניהם: הטכניון - היח' ללימודי חוץ, מכללת עזריאלי להנדסה ירושלים, המכון להכשרה חרדית וגופים נוספים. מאמן אנשים כיצד לחשוב בדיקות ומגלה להם את הדרך אל חשיבה מחוץ לקופסה.

כיצד נבצע את הבדיקות בשיטה זאת?

- ניתן להיעזר ברשימת בדיקות (Checklist) שהוכנה מבעוד מועד ומבוססת על פונקציות שנמצאו בהן באגים בסבבים קודמים. רשימה כזאת תכלול היסטוריית תקלות כולל גורם ושורש התקלה, מתוכה נאתר רשימת נושאים מומלצים לבדיקה.
- מומלץ לאפשר למומחים של המערכת מצד הפיתוח, הבדיקות או המוצר לקבוע את תכולת הבדיקות הרלוונטית שכן קיימת להם היכולת לזיהוי והכרות של המקומות הבעייתיים של המערכת - תקלות חוזרות, בעיות רגרסיה ידועות, צווארי בקבוק וכד'.
- דו"ח סיכונים של המערכת, ישמש לזיהוי ואיתור של פונקציות חשובות שמתועדפות בבדיקות על ידי בעלי העניין (מנהל הפרויקט, הלקוח או גורמים נוספים). נהוג להישען גם על ידע מעשי של הלקוחות עצמם שהושג בזמן בדיקות קבלה, זאת על ידי פנייה ישירה אליהם, במקרים רבים הלקוח הוא סוכן המידע הטוב ביותר שלך כיוון שהוא מכיר היטב את נקודות הכשל במוצר והוא שולט בתהליך העסקי מקצה לקצה.



דוגמה

במקרה שלפנינו, בשלב של תכנון בדיקות לגירסה 2 של המוצר נוכל להשתמש בפריסת הבאגים ההיסטורית של גירסה 1 ולהתמקד באזורים בהם נצפו תקלות בצפיפות גבוהה כבסיס לניחוש שגיאות.

כלומר הפונקציות הבאות יקבלו קדימות בבדיקות:

א. רכישת מוצרים אונליין (Online Purchase of products)

ב. מעקב מקוון אחרי הזמנות (Online tracking of shipments)

יתרונות השיטה

- יתרון מובהק הוא שהשיטה פשוטה יחסית ומתבססת על ניסיון העבר ככלי לזיהוי אזורים בעלי סיכון.
- השיטה מאתגרת את המוצר ומאפשרת להגיע לאזורי בדיקה שאולי היינו מפספסים בשיטות פורמליות יותר.
- השיטה מאפשרת להגיע לתקלות שהיינו מוצאים רק בבדיקות מקיפות יותר ולכן היא מסייעת באיתור מוקדם של באגים ועל כן מקצרת ומייעלת את סבב הבדיקות וכך חוסכת במקרים רבים זמן וכסף (עלות תיקון).

האנציקלופדיה לבדיקות | קובי יונסי

- שיטת ניחוש שגיאות יכולה לשמש בסיס מצויין לבדיקות חוקרות (Exploratory Testing) ולהוות גם בסיס לכתיבת מקרי בדיקה (Test Cases) מסודרים כפי שאנו מבצעים מול מפרטים טכניים.

חסרונות השיטה

- לא ניתן להשתמש בטכניקה זו במידה והבודקים בצוות אינם מנוסים מספיק בסביבת המוצר זאת מאחר והיא נשענת בעיקר על ניסיון הבודק.
- לא ניתן להשתמש בטכניקה זו כאשר המוצר הוא חדש לגמרי על המדף ואין מאחוריו היסטוריה של תקלות כי אז לא נוכל לנחש.
- באמצעות שיטה זאת בלבד לא נוכל להגיע למינימום של כיסוי דרישות ולכן גם לא נוכל לערוב למוצר איכותי כמה שניתן.
- קשה לנהל את הבדיקות בשיטה זו, הכיסוי אינו ניתן למדידה ואין מטרות ברורות, קיים קושי לשחזר תקלות שנמצאו בצורה זאת שאיננה מתועדת באופן מלא.

לסיכום,

מדובר בגישה פשוטה המתבססת על ניסיון העבר והיכולת לזהות מראש מקומות מועדים לסיכון. הגישה אינה פורמאלית אך בעלת חשיבות רבה באיתור מקדים של נקודות בעייתיות.

ניתן להשתמש בשיטת ניחוש שגיאות בכל שלב בבדיקות כשיטה משלימה לשיטות הפורמאליות ולא כשיטת בדיקות בפני עצמה. נהוג לעשות שימוש בטכניקה זאת לאחר שוידאנו שהתכולה המלאה של הבדיקות (מבוססות האיפיון) נלקחו בחשבון וכוסו על ידי צוות הבדיקות ורק אז להקצות זמן לבדיקות אלו כבדיקות משלימות לתהליך.

בחן את עצמך | טל פאר

הפתרון לשאלה

השאלה שלנו מתייחסת לפרק 4.4.1 בסילבוס שמדבר על טכניקה שנקראת "ניחוש שגיאות" (error guessing). טכניקה זו מתבססת על ידע קודם של הבודק כדי לחזות איפה יתכן ונעשו שגיאות שיגרמו הופעה של פגמים (defects) כשלים (failures). הידע של הבודק יכול להגיע ממקומות שונים, לדוגמה:

- ידע על כשלים שהתרחשו בעבר, במערכת הנבדקת או במערכות דומות. הבודק מכיר כשלים וכותב מקרי בדיקה (test cases) בניסיון לשחזר את הכשלים במערכת הנבדקת. אם הצליח, ידווח באג ויתוקן. יתכן, אפילו, שיתברר שזו רגרסיה במערכת.

  אם לא, מה טוב, הוכחנו שהמערכת "נקייה" באזור הזה וכשלים שקרו לא יקרו שוב.

- ידע על שגיאות נפוצות, למשל שגיאות שעושים מפתחים. גם במקרה זה יכתוב הבודק מקרי בדיקה שינסו לשחזר את הכשלים משגיאות אלה.

ניתן להשתמש בטכניקה זו בצורה שיטתית, ליצור רשימה של שגיאות, כשלים ופגמים אפשריים, לכתוב מקרי בדיקה, רשימות בדיקה (checklist) או אמנת בדיקות (charter test) עבור בדיקות חוקרות (exploratory testing).

עוד על ניחוש שגיאות בעמוד 12 בטור אנציקלופדיה של בדיקות.

הבה נבחן את התשובות השונות לנו:

א. על פי תשובה א' בודק שמשתמש בטכניקה זו צריך שיהיה לו ניסיון בפיתוח. למרות שניסיון וידע בפיתוח יכולים לעזור בניחוש שגיאות אפשריות, זו אינה דרישה הכרחית ולכן זו אינה התשובה הנכונה.

ב. הרעיון המרכזי בניחוש שגיאות הוא שהבודק מנסה לנחש שגיאות שנעשו במהלך חיי הפיתוח ולשחזר כשלים על בסיס שגיאות אלה ועל בסיס כשלים שנתקל בהם בעבר. לכן זו התשובה הנכונה.

ג. ניחוש שגיאות אינה טכניקה של שגיאות שימושיות (usability) ולכן זו אינה התשובה הנכונה.

ד. למרות שניחוש שגיאות שעשויות להיעשות בזמן תכנון היא חלק בסיסי מטכניקה זו, ניסיון לכתוב קוד מהיר כדי לראות איזה שגיאות ייעשו הוא לא הכרחי ולא תמיד מעשי ולכן זו אינה התשובה הנכונה.

טיפים וטריקים בבדיקות סלולר חלק שלישי

סלביק פשנין

בדיקות פונקציונליות בעולם המכשירים הניידים טומן בתוכו נדבכים רבים ומגוונים, בחלק השלישי של המאמר אצור מעין "רשימת מכולת" של תכנים שנדרש לשלבם וצריכים להיות בתת המודע שלנו בעת ביצוע הבדיקות.

1. **הסרה והתקנה של היישומון** - בעת התקנת היישומון והסרתו אנו נבדוק שכלל הקבצים מותקנים\מוסרים מהמכשיר הנייד שלנו ובעת התקנת היישומון מחדש אנו נבדוק פעם נוספת את הקבצים בכדי לבדוק שאין התנגשויות של מידע ו\או קבצים.

2. **הרשאות** - כל תוכנה דורשת הרשאות מסוימות מהמכשיר הנייד, אנו נוודא שההרשאות שהיישומון שלנו דורש (כגון, הרשאות למצלמה,מיקרופון, GPS, וכו') באמת נדרשות לתפעול תקין של היישומון שלנו ושאין דברים מיותרים או לחלופין דברים שנדרשת אליהם הרשאה והיא לא מוצגת.

3. **מחוות** - המכשיר הנייד שלנו תומך במחוות שונות כגון - החלקה, לחיצה, לחיצה כפולה, לחיצה חזקה (תלוי בגרסת ה-IOS שברשותכם), החלקה במספר שונה של אצבעות, סיבוב, וכו'. נדרש לוודא שהיישומון שלנו תומך בנדרש, לא פעם ראיתי יישומון קורס בשילוב של מספר מחוות כשהן אחת אחרי השנייה, אין במקרה הזה "כלל אצבע" אלא פשוט לנסות שילובים שונים של מחוות.

4. **תקשורת נתונים** - רצוי לבדוק את תפקוד היישומון בשילוב שונה של ביצועי רשת כגון: WIFI, GPRS, 3G, LTE, מצב טיסה. לכל מצב יכולה להיות התנהגות שונה ביישומון.

5. **הפרעות בעת שימוש** - בעת שאנו עובדים עם היישומון יכולות להיות הפרעות שונות כגון: הודעות, שיחה נכנסת, התראות שונות ושיחות וידאו. מצבים אלו יכולים להשפיע על תפקוד היישומון שלנו. כמו כן במכשירים ניידים שונים ישנם הגדרות של התראות, נדרשת לבדוק שיש התאמה בין מה שמוגדר במכשיר לבין מה שהיישומון שלנו מבצע.

6. **סוללה** - לסוללה ישנן השפעות רבות על תפקוד היישומון כפי שציינו במאמר מספר 2, כמו כן, גם בבדיקות פונקציונליות כדאי לבדוק את ביצועי היישומון שלנו בעת סוללה מוטענת במלואה, במצב טעינה, סוללה שרמתה נמוכה מ-10%.

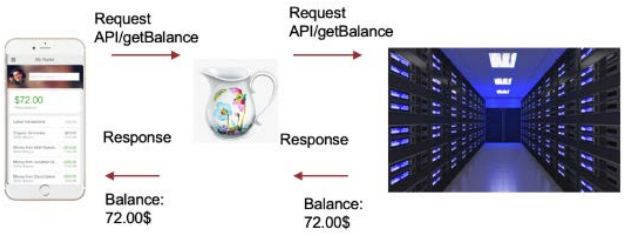
7. **הפעלה ברקע** - כדאי לבדוק נתוני מכשיר כאשר אנו שולחים את היישומון לעבוד ברקע המכשיר או שמחזיר את היישומון לקידמה, סגירתו והפעלתו מחדש.

8. **תאריכים וזמן** - שינוי של אזורי הזמן, תאריכים וזמן מכשיר יכולים להשפיע על תפקוד היישומון.

9. **TPA (תוספים)** - אם אנו משתמשים במקלדת ביישומון כדאי לבדוק תאימות למקלדות שונות שאותן ניתן להוריד דרך חנויות האפליקציה השונות, ההמלצה היא להוריד 2-1 שונות עם דירוג מירב ההורדות בחנויות האפליקציה.

10. **UI/UX** - תאימות וממשק המשתמש. לכל מערכת הפעלה iOS או Android יש מערכת כללים והגדרות לאיך יישומון אמור להתנהג ולהיראות על מנת להיות מותאם למערכות ההפעלה השונות.
[קישור למערכת iOS](#) / [קישור למערכת Android](#)

11. **שימוש ב-Proxy Charles** - כלי נוסף שאני ממליץ לשלב במהלך הבדיקות הוא Charles שהוא למעשה Proxy. דרך כלי זה ניתן לדמות בקלות שימושים שונים ומורכבים של משתמשים ביישומון. קל להבין את תפקידו ושימושו בדוגמה הבאה:

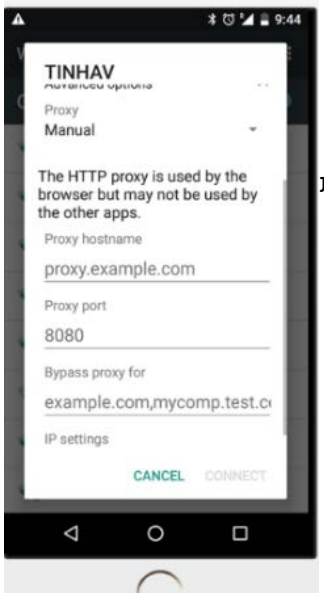


- נניח שיש לנו יישומון של חשבון בנקאי, בעת פתיחת היישומון יוצאת בקשת API/getBalance לשרת והשרת מחזיר "תשובה" כלשהי של מצב העובר ושב בחשבון. Charles למעשה יושב בין השרת לצד לקוח ומאזין לשני הצדדים.

באמצעות כלי זה ניתן:

1. לבצע עריכה של בקשות ותשובות בין צד הלקוח לבין השרת, כך שניתן לדוגמה לקחת את התגובה מהשרת של מצב העובר ושב שלנו ולשנות את הערכים בשני הצדדים.
2. שינוי/ויסות מצבי תעבורת רשת, כך ניתן לבדוק את עובדת האפליקציה שלנו במצבי רשת של LTE, 4G, מעבר למצב טיסה ועוד.
3. חזרה על מספר בקשות, למצב של שליחה לשרת מספר בקשות של לחיצת כפתור לדוגמה על קבלת סטטוס מצב חשבון עובר ושוב ניתן לראות כיצד מכשיר הנייד שלנו יגיב במצבים אלו, שימושי לבדיקות עומסים לדוגמה.

על מנת לאפשר את Charles על המכשיר הנייד:



1. צריך להתחבר במחשב ובמכשיר הנייד לאותה הרשת.
2. במכשיר הנייד לפתוח את הגדרות > הגדרות רשת > הגדרות פרוקסי > להגדיר את ה-IP של המחשב עם פורט 8888.
3. לאפשר את תעבורת הפרוקסי במחשב שלכם.
4. על מנת לאפשר תעבורת HTTPS נדרש להתקין את [האישור](#) על המכשיר הנייד.

12. **נקודות נוספות** - כדאי לבדוק ולזכור את התמיכה בשפות שונות, שימוש במצלמה, שימוש בחיישנים שונים שנמצאים במכשיר הנייד, ביצועים שונים (שדיברנו עליהם בפרק 2) ואינטגרציה עם כלים שונים דרך חנויות היישומונים של גוגל או אפל.

13. **חנויות היישומונים** - כדאי גם לזכור את משך הזמן שלוקח להעלות יישומון לחנויות וזה משתנה מאוד בין שתי מערכות ההפעלה השונות - גוגל כשעה וחצי לעומת אפל שלוקח כשבוע.

*בפרק הבא בסדרה נדבר על טרנדים עכשוויים בעולם המכשירים הניידים.

סלביק פשנין

ראש צוות QA בחברת Wix.com ומוביל טכנולוגי בסניף באר שבע כמומחה לבדיקות ואוטומציה. סלביק הקים את צוות הבדיקות בבאר שבע וכיום אחראי על בדיקות במוצרים עם יותר מ-100 מיליון משתמשים.

מרצה ושופט באירועים טכנולוגיים בארץ ובעולם ונהנה מאתגרים חדשים ובשיתוף הידע והנסיון שצבר.

ראיון עם נחום דימר, מייסד cloudbeat

שלום נחום, ספר לנו קצת על עצמך ואיך הגעת לתחום הבדיקות?

אני בן 39, מהנדס תוכנה במקצועי. התחלתי את הקריירה שלי כמפתח תוכנה בחיל האוויר אי שם בסוף שנות ה-90, לתחום הבדיקות הגעתי במקרה. אחרי כ-7 שנים בהן עבדתי כמפתח תוכנה, החלטתי לנסות את מזלי בתחום אחר, התקבלתי לתפקיד מהנדס pre-sale בחברת RADVIEW, אשר נחשבה באמצע שנות ה-2000 לאחת היצרניות המובילות של כלי בדיקות. התחלתי לעבוד עם WebLOAD, כלי לבדיקות עומסים שפיתחה החברה, ומהר מאוד התחלתי להתמחות בתחום זה. כמהנדס תוכנה, רכשתי הבנה עמוקה במערכות אשר אותן בדקתי - יתרון ענק אשר עם השנים הפך אותי לאחד המומחים המובילים בתחום זה.

מה הביא אתכם להקים את קלאודביט?

ב-12 שנים האחרונות ליוויתי וייעצתי להרבה ארגונים בארץ ובחו"ל והקמתי עשרות פרוייקטים בתחום בדיקות אוטומציה וביצועים. עם השנים, ראיתי את הפער הבלתי נתפס בין קבוצת הפיתוח לקבוצת הבדיקות, הן ברמה המתודולוגית והן ברמה הטכנית. הבנתי שחסרה פלטפורמה שמגשרת על הפער שבין DevOps ותהליכי פיתוח מודרניים לבין עולם הבדיקות שברובו עדיין מתבסס על בדיקות ידניות. בשלב זה החלטתי להקים את קלאודביט (CloudBeat), כשהצטרפו שותפיי רומן אובסייצב וערן דמלין. ערן בעצם היה הלקוח השני שלנו שהחליט להצטרף מיד אחרי זה למיזם. אנו 3 שותפים עם ניסיון מאוד עשיר ומגוון יכולות בתחום הבדיקות, אשר יחד הצלחנו ליישם בחברה את כל מה שלמדנו במהלך השנים, לרבות הטעויות שלנו וגם של אחרים. כיום, זוהי אחת הפלטפורמות המתקדמות בעולם ליצירה, הרצה, ניהול וניתוח של בדיקות אוטומטיות בעולמות של DevOps ואג'ייל.

מה לדעתך עתיד עולם הבדיקות ואיך אתה רואה את קלאודביט משתלבת בו?

עולם הבדיקות משתנה בקצב מסחרר, יחד עם כל תהליך פיתוח התוכנה אשר עובר מהפכה בשנים האחרונות. מתהליכי פיתוח של חודשים רבים והוצאת גירסה פעם בשנה עד שנתיים, אנו נדרשים כיום לפתח, לבדוק, לארוז ולהוציא גירסה לפחות פעם בשבועיים ואף לעיתים בקצב מהיר יותר. זהו שינוי דרמטי אשר משפיע על הצוותים הטכנולוגיים: פיתוח בדיקות גם יחד. בדיקות ידניות כפי שהכרנו הופכות אט אט להיות פחות עד לא רלוונטיות בפיתוח ובניה של גירסאות/מערכות בעולמות של DevOps ו-CI/CD. תובנה זו היתה לנו ברורה עוד לפני הקמת המיזם והבנו שעלינו ליצור פתרון שיתן מענה לשינוי, גם ברמה האישית וגם ברמת הארגון. חברת קלאודביט נוסדה במטרה לספק פתרון / גשר לפער שבין עולם הבדיקות המסורתי לבין עולם ה-DevOps. כבר היום, Oxygen Framework בדיקות קוד-פתוח שאנו פיתחנו, מסייע לאלפי בודקים ידניים להשתלב בעולם הבדיקות המודרני ולהתחיל הרבה יותר בקלות את הקריירה שלהם בבדיקות אוטומטיות.

אייל זילברמן

מייסד חברת Qualitest, חברת הבדיקות הגדולה בעולם. בעל ניסיון כבודק, מנהל בדיקות ויועץ לארגונים בתחום הבדיקות. פרסם עשרות מאמרים מקצועיים ונואם בכנסים ישראליים ובינלאומיים. בין המקימים של מגזין בדיקות התוכנה הישראלי הראשון "חושבים בדיקות" וחבר בצוות ה-AB של ITCB® - ארגון ההסמכה הישראלי לבדיקות. מייסד ושותף במספר חברות היי-טק בתחום הבינה המלאכותית ובינה עסקית.

איפה אתה רואה את קלאודביט בעוד 5 שנים?

אנו צופים שבשנים הקרובות יותר ויותר חברות יטמיעו תהליך Continuous Testing לצד פיתוח אג'ילי. כמות ומורכבות של בדיקות אוטומטיות תלך ותגדל. הקושי הגדול בשנים הבאות לא יהיה איך לכתוב בדיקה אוטומטית אלא איך לנתח את התוצאות של אלפי ועשרות אלפי בדיקות שרצות ברמה יומית בסביבת DevOps. אנו מאמינים שקלאודביט תהיה פלטפורמה מובילה בתחום זה, שתעזור לחברות להריץ כמויות גדולות של בדיקות בצורה יעילה ויציבה ותספק ניתוח אינטגרטיבי ואגרגטיבי טכני ועסקי אשר ישפר ויקצר משמעותית את תהליכי הבדיקות לרבות ובמיוחד הזמן הנדרש להן.

אתה גר בפולין, איך אתה רואה את אופן ניהול החברה מרחוק? האם יש שוני לעומת עבודה בישראל?

עת שהקמנו את החברה, קיבלנו החלטה על מודל עבודה של עבודה מרחוק. בנינו מראש חברה גלובאלית שיודעת לעבוד עם שווקים מרוחקים וליעל עלויות תפעול. כיום, הצוות שלנו מבוזר בארבע מדינות ובשלושה אזורי זמן שונים. לפעמים זה מאתגר לא לשבת מול המפתח או מנהל מוצר ולהסביר לו הכל "כתף אל כתף" אבל הריחוק גורם לנו לעבוד מתודולוגי יותר ולחשוב לעומק על הדברים. עבודה מרחוק היא השקעה המצריכה ניהול נכון, שיפור תהליכים מתמיד והכי חשוב הון אנושי המתאים לאופי עבודה זה. באופן כללי, התברכנו בצוות מדהים אשר לו רקע טכנולוגי, מומחיות, ניסיון לצד תרבויות שונות. מחד, זו תעוזה ויצירתיות של צוות ניהול ישראלי ומאידך, גישה המביאה עימה עבודה מסודרת ומדוקדקת של הצוותי פיתוח שלנו בפולין ואוקראינה.

מה הטיפים שיש לך לתת לקהילת הבדיקות וליזמים בקהילה?

אני אצטט את סטיב ג'ובס במה שהפך להיות למוטו שלי עם השנים: "***Stay hungry, stay foolish***". אף פעם אל תעצרו ותסתפקו במה שהושג. תמציאו את עצמכם מחדש ותלמדו להנות מזה.

לבודקים שבינינו ולכל אלו שקוראים כתבה זו אגיד: אל תפחדו משינויים. כל שינוי הוא לטובה. תתקרבו לטכנולוגיה, תשקיעו extra mile במה שאתם עושים. בעולם של מחר, אין יותר מקום לחלוקה לבודקים ידניים ובודקי אוטומציה. כל בודק הוא טכנולוג, כל בודק הוא חוליה מקשרת בין פיתוח לגירסה שעוד שניה צריכה לעלות לאוויר. תלמדו, תשקיעו ותהיו חלק מהעתיד!

נחום דימר

מנהל ושותף בחברת CloudBeat – פלטפורמת בדיקות חכמה ליצירה, הרצה וניתוח בדיקות אוטומטיות בסביבת DevOps.

בעל נסיון רב כיזם, מרצה ומומחה בתחום בדיקות ביצועים, עומסים ואוטומציה.

עזר לבנות מערכי בדיקות לעשרות חברות בארץ ובחו"ל.

ארץ, עיר, חי, צומח ו-QA | טליה יורמן

קצת על עצמי

שמי טליה יורמן, אני בת 48, במקור כפר סבאית, נשואה מעל 20 שנה לאמיר, חיפאי שהכרתי במקרה במסיבה בת"א. גרנו בחיפה כ-10 שנים וב-13 השנים האחרונות אנחנו גרים בישוב הקהילתי 'שמשית' שבעמק יזרעאל. יש לנו שלושה ילדים: אביב בת 12, גל בן 16.5 ושחף בת 19.

אני אוהבת לרקוד, ולצאת להליכות בטבע. בהשכלתי אני מהנדסת כימיה, למדתי ב'שנקר' והתמחתי בתחום הכימיה הטקסטילית, אולם לקראת סיום התואר הבנתי שלטקסטיל בארץ אין עתיד וחיפשתי עבודה ככימאית בכל תחום.

מיד עם סיום 4 שנות הלימוד התקבלתי לעבודה בחברת "דר' פישר" המייצרת מוצרי קוסמטיקה ותרופות, עבדתי במעבדת בקרת האיכות של המפעל.

בזמן הזה הכרתי את אמיר ולכן לאחר שנה התפטרתי והחלטתי לעזוב את המרכז ולעבור לגור איתו בחיפה.

לאחר כשנה שבה עבדתי כתועמלנית בחברת התרופות הישראלית "רפא" ושיווקתי את מוצריה בצפון הארץ החלטתי לשנות כיוון.

באותה תקופה, החלה "בועת ההייטק" והמדינה עודדה מהנדסים מתחומים שונים לעשות קורסי הסבה להנדסת מחשבים. לאחר מבדקים וסינונים התקבלתי לקורס הסבת מהנדסים לתוכנה של הטכניון במלגה של משרד התמ"ת. זה היה קורס מואץ של 3 סמסטרים בשנה, שעשה השלמה לבעלי תואר הנדסי כלשהו לתחום התוכנה.

הגודל כן קובע

מיד לאחר הקורס התקבלתי לאמדוקס חיפה כבודקת תוכנה, עם אפשרות לעבור לפיתוח בעתיד... אולם התאהבתי במקצוע הבדיקות ולמדתי להכיר את חייו של המפתח ולכן החלטתי להישאר בתחום הבדיקות.

בעבודה כבודקת תוכנה מצאתי ביטוי לביקורתיות ולפרפקציוניסטיות שלי ויכולתי לייצר איכות.

אמדוקס היתה בית-ספר נהדר לתחום, עברתי שם קורס בדיקות ולמדתי לבדוק מערכת מורכבת, כולל מסד נתונים, אינטגרציות ובדיקות תהליכי E2E.

אני בהחלט ממליצה לבודק מתחיל להתחיל במקום עבודה גדול יחסית שיש בו מחלקת בדיקות מנוסה, ששם ניתן לקבל הדרכות וללמוד את התחום לעומק תוך כדי תנועה.

במקביל ל-7 שנותי שם, למדתי בטכניון תואר שני כללי עם התמחות באבטחת איכות ואמינות.

התואר לא התמקד בתוכנה ודווקא נהניתי לקחת קורסים כלליים בטכניון בעיקר בתחום הביו-רפואה.

לאחר שנולדו לי שני ילדי החלטנו לעבור מן העיר ל"כפר", ועברנו לישוב קהילתי בשם 'שמשית', עקב המעבר החלטתי שהגיע הזמן לעזוב את עבודתי באמדוקס.

התחלתי עבודה חדשה בחברת מרקיורי, כבודקת תוכנה של מערכת CRM שפיתחה חברה קטנה בתפן ("תפנסופט") שנקנתה על ידה, כעבור מספר חודשים נקנתה מרקיורי ע"י HP שהחליטה כי המוצר אותו פיתחה החברה מתפן יפסיק להימכר ואין לו עתיד פיתוח ובעצם עתיד החברה התערער. במקביל לתקופה בתפן, קו"ח שלי התגלגלו גם ל'רפאל' ולאחר תהליך סיווג של עוד כחודשיים התקבלתי לעבודה ב'רפאל' במכון לשם (משגב) דרך חברת קוואליטסט. ברפאל' עבדתי כבודקת תוכנת מערכת שו"ב (שליטה ובקרה) במחלקת הטילים. בשנה בה הייתי עובדת קוואליטסט, עברתי דרכם את הסמכת ISTQB®.

לאחר כשנה ברפאל יצאתי לחופשת לידה של ילדתי השלישית, וממנה לא חזרתי.

לקחתי חופשה ארוכה מהתחום, גידלתי את ילדיי ועבדתי מהבית בשיווק מוצרי בריאות וגם למדתי הדרכה של ספורט/מחול 'ניה' וניסיתי לעבוד כעצמאית בתחום.

אני חושבת שלהיות בודק טוב זו אומנות

בנוסף עבדתי במשרה חלקית בהדרכה בביה"ס היסודיים בתוכנית המפתחת מיומנות חשיבה ע"י משחקי חשיבה.

לאחר כשנה הבנתי שאני זקוקה לעבודה עם יותר אתגר אינטלקטואלי וחזרתי לחפש עבודה בתחום אותו ידעתי הכי טוב ואהבתי - בדיקות תוכנה, אבל הפעם החלטתי שאני מעוניינת להתמקד בתחום של מוצרים רפואיים.

לפעמים משהו טוב מגיע בחבילות קטנות

לאחר מספר חודשים של חיפוש וראיונות, בינואר 2012 התקבלתי לעבודה זמנית (דובר על חצי שנה...) בחברת Given Imaging ('גיוון') ביקנעם, (כיום חלק מ- Medtronic).

מכיוון שהייתי אחרי הפסקה ארוכה מהתחום הסכמתי לעבודה זמנית ולו רק כדי "להכניס שוב רגל" לתחום והפעם כמו שרציתי, בחברה למוצרים רפואיים, החלטה שנמצאה כנכונה, שהרי הזמני הפך לקבוע ואני שם כבר מעל 8 שנים!

המוצר של Given סיקרן אותי מאוד ופתח בפני עולם שלם של מוצרים רפואיים ואפשר אפילו לומר מצילי חיים, שכן החברה מפתחת גלולת מצלמה שמצלמת תמונות לאורך מערכת העיכול לשם דיאגנוזה של מחלות מעיים כולל סרטן המעי הגס. המוצר הוא מערכת שלמה המשלבת חומרה ותוכנה, התוכנה מאוד מורכבת ומשתמשי הקצה הם בעיקר אחיות ורופאים (למידע נוסף על המוצר ניתן להכנס לקישור).

ארץ, עיר, חי, צומח ו-QA | טליה יורמן

מוצר רפואי חייב לעמוד בדרישות רגולציה מחמירות כמו תקני ISO ו- FDA, מה שהופך את הבדיקות ותיעודן לעניין מאתגר שכן על מסמכים כמו STR, STD, STP להיות מדויקים ומקיפים ולהכיל את כל דרישות התקנים. פעם בכמה זמן החברה עוברת ביקורת של גופים חיצוניים ושל ה-FDA לוודא שהיא עובדת נכון ע"פ כל התקנים וכעובדים בתחום אבטחת האיכות אנחנו שותפים להכנות למבדקים הללו ולרוב אנחנו גם נבדקים בעצמנו.

הפיתוח ב-Given עובד בשיטה האג'ילית, כך שאנחנו הבודקים מעורבים בשלבי פיתוח המוצר ושותפים להתהוותו.

בבדיקות של מוצר רפואי, תפקידו של מהנדס אבטחת איכות היא לא רק בדיקות, אנו אחראים גם על דו"חות המוכיחים כיסוי דרישות, על ניהול סיכונים, על עבודה ע"פ תקנים ואפיון תהליכי עבודה.

לאחר שנה במעמד של "עובדת זמנית" הפכתי להיות עובדת קבועה ולאחר כשלוש שנים נוספות קודמתי להיות מובילה מקצועית. מאז הובלתי בדיקות של מוצרי תוכנה חדשים בתחום הרפואה אשר בנויים על פלטפורמות: דפדפן, ענן וסלולר. התפתחות מוצרי החברה והטכנולוגיות גרמה גם לי להתפתח, שהרי בדיקות למוצרים הללו דורשות בנוסף לבדיקות פונקציונליות גם לחשוב על בדיקות אבטחה, עומסים, GUI, דפדפנים, מכשירים שונים וכו'.

עולם התוכנה המתפתח במהירות מייצר אתגרים רבים לנו הבודקים, לעיתים תסריט בדיקה אחד צריך לרוץ על מטריציית סביבות/מכשירים רבים ושונים, דבר שמייצר מקרים ובאגים שונים.

לפני כשלוש שנים, החברה נקנתה ע"י חברת 'מדטרוניק' האמריקאית, ובעצם כיום אנו חלק מקונצרן גלובלי המחייב אותנו לעבוד בשיטתיות ובאחידות עם שאר החברה, ישנם כלים רבים להדרכה ולתיעוד שיש להכיר וללמד וזאת בנוסף לעבודה השוטפת.

בדיקות תוכנה זו אומנות

כמי שנמצאת בתחום כבר שנים רבות, אני חושבת שלהיות בודק טוב זו אומנות.

אם מסתכלים על ה"מיקרו" - על הבודק לשים לב לכל הפרטים הקטנים, לא להתפשר על כל שדה וכל כפתור. אם מסתכלים על ה"מאקרו" - על הבודק להיות בעל ראיה מערכתית ולראות את המוצר בכללותו, כבעיני המשתמש. לחשוב מחוץ לקופסה ולמצוא את המקרים הבעייתים, לחשוב על מקרי בדיקה לא טריוויאליים, כאלו שהמפתח לא חשב עליהם, או לא התייחס אליהם בקוד. והכי טוב זה למצוא את המקרים האלו כבר בשלב האפיון ולהיות ביקורתיים מול הדרישות והמפתחים.

אני חושבת שתפקידו של הבודק הידני, המוח והעיניים שלו, לעולם ישארו חשובים ונצרכים.

עם זאת אני רואה חשיבות בבדיקות אוטומטיות אך לא מאמינה שיכולות להחליף את הבודקים הידניים!

המלצתי לכל בודק בתחילת דרכו הוא למצוא את החברה/התחום/המוצר שאליו הוא מתחבר.

אני מאמינה שגם כשאתה "רק בודק", אתה צריך להאמין ולאהוב את המוצר אותו אתה בודק!

מאז שאני שותפה בתהליכי פיתוח המוצר כבר משלבי איפיון המוצר מצאתי עניין בתחום חוויית המשתמש (הנדסת אנוש) ולכן החלטתי ללמד את התחום. בפברואר האחרון סיימתי קורס "אפיון חווית משתמש" ביחידה ללימודי חוץ של אוניברסיטת חיפה. אני רואה בידע הזה עוד תוספת להבנה שלי כבודקת ואת ראיית המשתמש, צרכיו ותרחישיו ומקווה להצליח לשלב ידע זה בתפקידי בעתיד.

גם כשאתה "רק בודק", אתה צריך להאמין ולאהוב את המוצר אותו אתה בודק!

לסיכום

בדיקות תוכנה זהו מקצוע עם אחריות וסיפוק רב. לא חשוב אם "התגלגלתם" למקצוע כמוני או אם בחרתם בו מלכתחילה. אם אתם לא מתפשרים על איכות, ואתם רוצים להיות משמעותיים לחברה שבה אתם עובדים, כנראה שאתם במקום הנכון!

אני בעצם עוסקת בתחום כבר כמעט 20 שנה ולא משעמם לרגע.

גם ההפסקה שעשיתי באמצע מהתחום בכלל, היתה נכונה וחשובה לי כדי להבין שאני רוצה לחזור!

חשיבות מקצוע הבדיקות ואבטחת האיכות בחברות הולכת ומתחזקת, והוא נחשב משמעותי מאד בעיקר בחברות המפתחות מוצר קריטי (כמו בתחום הרפואי או בתחום הפיננסי).

זהו תפקיד שיש בו הרבה אפשרויות להתפתחות, יחד עם למידת מוצרים חדשים בחברה או עבודה בחברות שונות.

אל תתפשרו, אני מאמינה שכל אחד יכול למצוא את מקום העבודה בו יתחבר למוצר ולאופי החברה, ואז יוכל לתרום הכי הרבה, לתת ולקבל.

בהצלחה!

מחפש צרות | מיכאל שטאל

בדיקות מטמורפיות

עולם ההייטק, כך לפחות מקובל לחשוב, נע ומשתנה במהירות. רעיונות חדשים מופיעים בקצב גבוה, ומה שהיה מיוחד ושונה לפני שנה הוא כבר כמעט חדשות ישנות היום. ומצד שני... מישהו שמע על טכניקת בדיקות שנקראת "בדיקות מטמורפיות"? אם לא שמעתם, אין פלא. הרעיון חדש מהניילון. הוא הופיע לראשונה במאמר של Tsong Yueh Chen, Shing Chi Cheung, and Siu Ming Yiu ממש לאחרונה... בשנת 1998.

די מדהים שהטכניקה הזו לא ידועה יותר ואף לא מופיעה בחומרי הדרכה לבודקים - אפילו לא בחומר מתקדם. אין גם התייחסות לרעיון ב-ISO 29119 (תקן לבדיקות תוכנה), אם כי זה כנראה עומד להשתנות. אבל אל דאגה. גם אם לא הכרתם את המושג, הרי שעם קריאת מאמר זה מצבכם (לפחות בעניין הזה) ישתפר לאין ערוך.

לא מספיק הטכניקות הקיימות?

הרבה מהמערכות שאנו בודקים הן דטרמיניסטיות ומאפשרות לנו לכתוב מקרי בדיקה חד-משמעיים. לדוגמה: נתונה תוכנה שמחשבת פונקציה y=f(x). עבור כל קלט x אני יכול לחשב בסימולטור או בתוכנה מקבילה את התוצאה הצפויה ולהשוות למה שהמערכת הנבדקת מחשבת. למעשה, כל טכניקות הבדיקה הבסיסיות בנויות על ההנחה שיש לנו "אורקל בדיקות" (test oracle) שמולו ניתן להשוות את התוצאה של כל בדיקה.

אבל יש לא מעט מערכות שבהן המצב לא כל כך פשוט. במערכות אלה, התוצאה הצפויה ניתנת לחישוב מראש, אבל רק ברגע נתון; או רק במאמץ חישובי או לוגיסטי לא מבוטל. כשנתוני המערכת או סביבתה ישתנו, התוצאה תשתנה גם היא. תוכנות המשתמשות בבסיסי נתונים גדולים הן דוגמה טובה לזה. חישבו על מערכת חיפוש טיסות. כל זמן שהבדיקות נעשות על מערך נתונים שהוכן במיוחד לצורך הבדיקות, הכל טוב. אנחנו יודעים מה הכנסנו לבסיס הנתונים ועל כן מה התוצאות הצפויות לשאילתה זו או אחרת. אבל הנסיון מראה שכשמשחררים תוכנה כזאת לשימוש בעולם האמיתי מתגלים באגים חדשים שנובעים ממצבים שלא דמיינו שייוצרו בבסיס הנתונים. הפתרון הוא לעשות גם בדיקות "in production" - או לפחות על העתק של בסיס הנתונים האמיתי. הבעיה אז היא שקשה מאוד להגדיר את התוצאות הצפויות. כיוון שכל הזמן הנתונים משתנים (אנשים תופסים מקומות; אנשים מבטלים הזמנות; חברות מוסיפות טיסות) הרי חיפוש שיבוצע ברגע נתון ייתן תוצאות מסוימות ואותו חיפוש יניב תוצאות שונות כשבסיס הנתונים של הטיסות ישתנה.

> **"ישנם מקרים בהם ניתן לנסח יחס צפוי בין תוצאה של בדיקה אחת ובין תוצאות של בדיקות אחרות"**

בעיה אחרת היא העלות של יצירת מקרי הבדיקה. גם כשיש לנו סביבת בדיקות שבה אנו שולטים לגמרי, לא יעיל לכתוב בדיקות רגילות שמורצות תמיד על אותו סט נתונים כיוון שחייבים לוודא שאלגוריתם החיפוש עובד נכון גם כשבסיס הנתונים משתנה. נניח שכתבתי בדיקה המחפשת טיסה לניו יורק לשני אנשים, במחלקת תיירים. ידוע לי שכרגע יש בבסיס הנתונים שתי טיסות המתאימות לתנאים שנתתי (תאריך, נקודת יציאה וכו'). אם הניסוח של הבדיקה הוא "קשיח" ("תוצאה צפויה: שתי טיסות") הרי שלפני הרצת הבדיקה אצטרך לארגן את בסיס הנתונים כך שיכיל רק את אותן שתי אפשרויות, אחרת הבדיקה תכשל. כמובן שאצטרך לכתוב עוד הרבה בדיקות מפורטות, ולפני הרצת כל אחת מהן לדאוג לעדכן את נתוני הטיסות באופן דטרמיניסטי. זה לא יעיל. וכל זה עוד במערכת שקל להבין את הגורמים שמעורבים בה ומשפיעים על התוצאה. במערכות יותר מסובכות, בהן יש יחסים לא לינאריים בין נתונים, קלטים ותוצאות, המצב קשה עוד יותר.

ישנן גם תוכנות שמלכתחילה קשה להגדיר עבורן תוצאה צפויה. נניח שאנו חוקרים התנהגות של חומר כלשהו בתנאים קיצוניים. פיתחנו אלגוריתם שמתאר את ההתנהגות ואנו רוצים לוודא אם הוא נכון. לא נשאר אלא להיכנס למעבדה, להריץ ניסויים ולהשוות למה שהאלגוריתם נותן. את האלגוריתם נממש כמובן בקוד, שאותו צריך לבדוק. אבל ברגע שתהיה אי התאמה בין האלגוריתם לתוצאות הניסוי, איך נדע אם הבעייה היא שיש באג בקוד, או שהאלגוריתם אינו נכון? אם הניסויים עולים כסף רב, נרצה גם למצוא דרך לבדוק את האלגוריתם עם מינימום הרצות ניסויים במעבדה. איך נבדוק את הקוד של האלגוריתם כשאין לנו "תוצאות צפויות"?

בדיקות מטמורפיות מספקות פתרון מסוים לבעיות שהעליתי. טכניקה זו מאפשרת לנו להריץ הרבה מאוד בדיקות שניתן לייצר באופן אוטומטי, ושאינן דורשות הגדרה מראש של התוצאה הצפויה.

הטכניקה

ישנם מקרים רבים בהם ניתן לנסח יחס צפוי בין תוצאה של בדיקה אחת ובין תוצאות של בדיקות אחרות, גם כשאני לא יודע אם התוצאות עצמן נכונות. נחזור לדוגמא של חיפוש טיסות ונראה שגם אם לא ידוע לי התוכן של בסיס הנתונים, אני יכול לומר משהו אינטליגנטי על התוצאות הצפויות, ברגע שהרצתי בדיקה אחת.

קודם כל אעתיק את בסיס הנתונים מ-production לעותק שעליו אבצע את הבדיקות. אריץ את החיפוש של טיסה לניו יורק עם שני אנשים ואקבל תוצאה מסוימת (נניח: שלוש טיסות אפשריות). כיוון שאני לא יודע מה מכיל בסיס הנתונים, אני לא יכול לפסוק אם תוצאה זו היא נכונה. אבל: כיוון שידועה לי התוצאה הזאת, הרי שאוכל לומר משהו על התוצאה הצפויה כשאריץ את הבדיקה שוב, בשינוי מה:

השינוי	התוצאה
חיפוש טיסה לשלושה אנשים	אותו מספר טיסות כמו בבדיקה הראשונה, או פחות
חיפוש טיסה לאדם אחד	אותו מספר טיסות כמו בבדיקה הראשונה, או יותר
חיפוש טיסה לערים ברדיוס של 100 ק"מ מניו יורק	אותו מספר טיסות כמו בבדיקה הראשונה, או יותר

מיכאל שטאל

ארכיטקט בדיקות תוכנה באינטל, ישראל
עוסק בעיקר בבדיקות מערכות משובצות מחשב. במסגרת תפקידו, מיכאל מגדיר שיטות בדיקה ומתודולוגיות עבודה, עוסק בהדרכה ולפעמים אפילו מרשים לו לבדוק משהו (שזה הכי כייף).
מיכאל מציג תכופות בכנסים בארץ ובחו"ל ומלמד בדיקות תוכנה בפקולטה למדעי המחשב באוניברסיטה העברית.
ניתן לראות חלק מהמצגות והמאמרים שלו באתר www.testprincipia.com

מעתה אפשר להריץ את הבדיקה הראשונה עם איזה נתונים שארצה מבחינת תאריך או מצב בסיס הנתונים של הטיסות. למרות שכלל איני יודע אם התוצאה נכונה, הרי שהתוצאה הצפויה של שלושת הבדיקות הבאות נשארת אותו דבר. כלומר, נפטרנו מהתלות הקשיחה במצב של בסיס הנתונים. בצורה זו ניתן להגדיר עוד הרבה **יחסים מטמורפיים** בין תוצאות של בדיקה אחת, לתוצאות הצפויות מבדיקות שמשנות במשהו את הבדיקה הראשונית (מכאן השם "בדיקות מטמורפיות" – מהמילה "מטמורפוזה" שמשמעותה, בין השאר, שינוי צורה).

בדיקות מטמורפיות מאפשרות לייצר כמות רבה של בדיקות, על מערכות מסובכות, בלי צורך להשקיע זמן רב בחישוב התוצאות הצפויות או "ארגון" מערכת הבדיקה לפני כל הרצה. הנה עוד כמה דוגמאות שמופיעות במאמרים המתארים את הטכניקה:

בדיקת פונקציה sin(x): על כל הרצה של בדיקה עבור ערך X מסוים, אפשר להריץ גם:

- sin(-x): התוצאה צריכה להיות זהה
- sin(x+Π): התוצאה צריכה להיות זהה, אבל שונה בסימן
- sin(x+2Π): התוצאה צריכה להיות זהה

הערה: כשמדובר בתכונה בסיסית של החישוב או של התוכנה, כמו במקרה של חישוב סינוס, יש שקוראים לזה "בדיקות מבוססות תכונות" (property based testing). הרעיון דומה לבדיקות מטמורפיות, ובחלק מהמאמרים הדוגמה של סינוס מופיעה כבדיקה מטמורפית.

מימוש

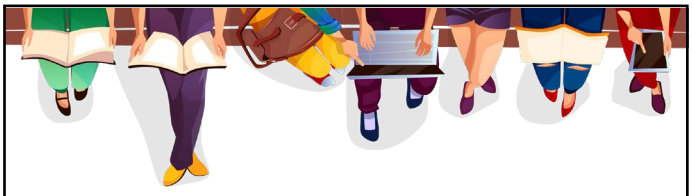
החלק הקשה (המאתגר ומעניין!) במימוש בדיקות מטמורפיות הוא הגדרת היחסים המטמורפיים. זה משהו שאין ברירה אלא לשבת ולשבור את הראש. אין אלגוריתם או אוטומציה שיעשו את זה בשבילכם ולעתים זה בכלל לא טריוויאלי. במאמר המקורי מ-1998 יש כמה דוגמאות ליחסים מטמורפיים בחיפוש בינארי - וזה לא רעיונות שעולים באופן טבעי ומיידי כשחושבים על זה. אתן כאן דוגמה למקרה שבו כתבנו בדיקות שרק עכשיו, בדיעבד, אני מבין שהיו בדיקות מטמורפיות.

בדקנו תוכנה של זיהוי עצמים בסרט וידיאו. הקלט לאלגוריתם הזיהוי היה סרט וידיאו, שבו הופיע חפץ. האלגוריתם היה אמור לזהות את החפץ ואת מיקומו בתמונה.

המצלמה נעה מימין לשמאל, כך שמיקום החפץ השתנה בכל תמונית (frame), 30 - פעמים בשנייה, שזה קצב התמוניות בסרט וידיאו. על מנת שתהיה "תוצאה צפויה" , עברנו על הסרט, ובאופן ידני סימנו איפה נמצא החפץ בכל תמונית עשירית (כן, בעסה של עבודה). אבל אז הבנו שאפשר לבדוק עוד משהו, בלי השקעת המאמץ ההזוי של סימון מיקום החפץ בתמוניות נוספות. כיוון שהמצלמה נעה באופן חלק מימין לשמאל, הרי שאנו מצפים שבכל תמונית המיקום של החפץ יהיה טיפה יותר ימינה מזה שבתמונית הקודמת. בצורה כזאת הוספנו עוד עשרות בדיקות, במאמץ מזערי. מחקרים מראים שבעזרת בדיקות מטמורפיות אפשר לגלות באגים שמצליחים לחמוק מבדיקות המבוססות על אורקל בדיקות.

מטריקות כיסוי

יחד עם הגדרה של טכניקת בדיקות מקובל לתת גם מדד כיסוי: כמה טוב מקרי הבדיקה שהגדרנו מכסים את כל הבדיקות שהטכניקה מגדירה. למשל, עבור הטכניקה של בדיקת מחלקות שקילות, כיסוי של 100% משמעותו שמכל מחלקת שקילות בדקנו לפחות נציג אחד. עבור בדיקות מטמורפיות המצב לא פשוט כל כך. בתור התחלה, אין גבול עליון ברור למספר היחסים המטמורפיים שאפשר להגדיר. בדוגמה של חיפוש טיסות נתתי שלושה יחסים. ברור שאפשר להגדיר עוד המון יחסים אחרים. ועבור כל יחס שהגדרנו, אפשר להריץ מספר אין סופי של בדיקות, כשכל פעם משנים את הבדיקה הראשונה, זו שיחסית אליה משווים את תוצאות הבדיקות הבאות. ואכן אין כרגע (ומן הסתם גם לא יהיה) מדד כיסוי לטכניקה זו. כן הגיוני להגדיר ציפיית מינימום שעבור כל יחס מטמורפי נריץ לפחות בדיקה אחת.



חומר לקריאה בתקופת הבידוד החברתי

למי שרוצה ללמוד עוד על בדיקות מטמורפיות אני ממליץ לקרוא את Metamorphic testing. במאמר זה מסכם הכותב, הלל וויין (Hillel Wayne) את הנושא באופן יפה (וזאת תהיה חזרה טובה על החומר שהבאתי כאן). לקראת סוף הכתבה יש קישורים לשורה של מאמרים בנושא, למי שרוצה להעמיק. המאמר המרכזי שעליו מצביע הכותב הוא: Metamorphic Testing: A Review of Challenges and Opportunities שווה לקרוא. כמו כן, אשמח לשמוע על ניסיונות ותוצאות של יישום בדיקות מטמורפיות. שלחו אלי למייל – או (עדיף!) פרסמו בפורום בדיקות תוכנה בפייסבוק.

כלי לבדיקות - Qase | ניצן גולדנברג

בשל מצב הקורונה בארץ ובעולם יותר ויותר חברות מוצאות עצמן במצב כלכלי קשה ולכן מחפשות פתרונות בעלות מינימלית.

הכלי שאותו אני הולך לסקור במאמר זה נקרא Qase והוא חינמי לחלוטין.

הכרות

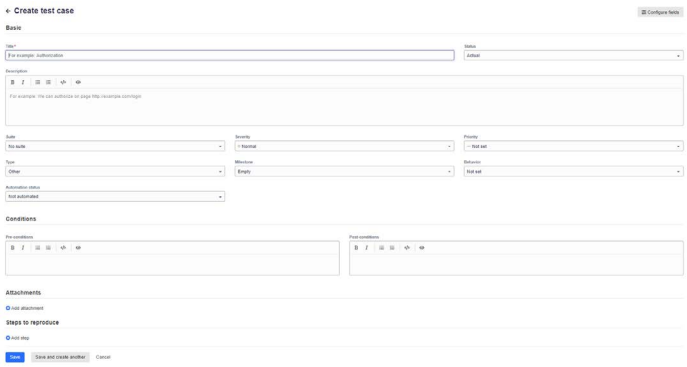
Qase הינה מערכת ניהול תסריטי בדיקות בענן למפתחים ולבודקים. המערכת מותאמת לעבודת יחיד או צוות, זאת בכדי להגדיל משמעותית את יעילות הבדיקות ולנהל את מקרי הבדיקות (Test Cases), להרכיב תוכניות בדיקה (Test Plans) ולבצע ריצות בדיקה (Test Runs) בכל דרך אפשרית.

התכונות (Features) המרכזיות של המערכת הם:

1. מאגר של תסריטי בדיקה
2. ביצוע ריצות בדיקה
3. הזמנת חברי צוות להשתמש במערכת ובפרויקט
4. אינטגרציה (Integration) עם מערכות ניהול תקלות
5. בדיקות API
6. עבודה מול מספר פרויקטים בו זמנית

מאגר תסריטי בדיקות

ניתן לאחסן ולנהל את כל מבחני הבדיקה ומקרי הבדיקה במקום אחד. מבחני הבדיקה (Test Suites) מאפשרים לארגן את כל תסריטי הבדיקה לקבוצות לפי שיקול דעתו של הבודק ולפי הצרכים של הבדיקה. ניתן להגדיר פרמטרים כגון: חומרה, עדיפות, תיעדוף, תנאים מוקדמים, תנאים לאחר התנאים, צעדים לשחזור ועוד.



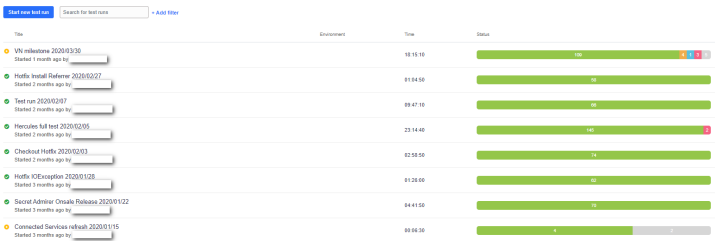
ביצוע ריצות בדיקה

ניתן ליצור תוכניות בדיקה שונות ולהוסיף להן תסריטי בדיקה ומבחני בדיקה קיימים או חדשים. ניתן להריץ את תוכניות הבדיקה ולעקוב אם תסריטי הבדיקה עברו, נכשלו או חסומים מהמסך ריצות מבחן עצמו.

ניתן להשתמש באשף חכם אשר ידריך את המשתמש בתוכנית הבדיקה ויסייע לבדוק את כל מקרי הבדיקה בבת אחת. מעקב אחר זמן הריצה יציג מידע מפורט על זמן אשר הוקדש לכל מקרה.

כמו כן, ניתן לשתף בקלות את דוח הבדיקה הכולל מידע מפורט על כל תסריט בדיקה בריצה למנהל או ללקוח בכמה לחיצות בלבד.

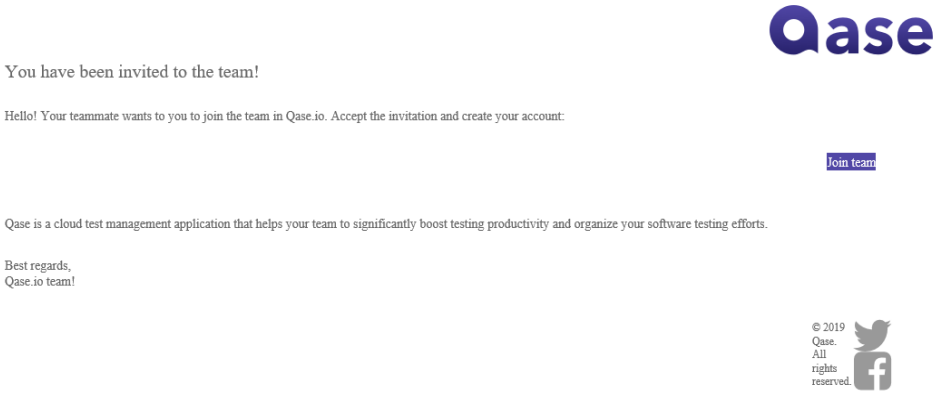
בתמונה ניתן לראות את מספר התסריטים אשר עברו, חסומים, דולגו, נכשלו ולא נבדקו לפי צבעים לכל סטטוס.

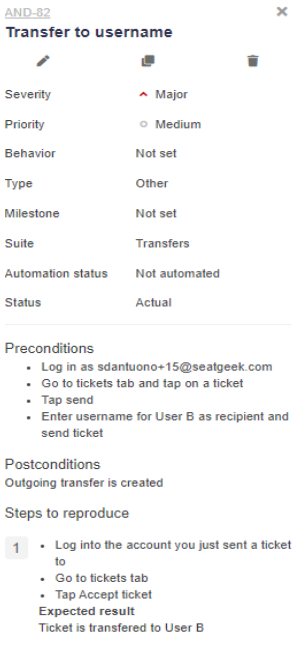


הזמנת חברי צוות השתמש במערכת ובפרויקט

ניתן להזמין את חברי הצוות לפרויקטים והם יכולים לגשת לפרויקט על בסיס בקרת גישה לפי תפקידים. הצוות יכול לעבוד יחד על בדיקות הריצה, כתיבת מבחני בדיקה, תסריטי בדיקה והכנת תוכניות בדיקה.

בקרת גישה מבוססת תפקידים, תעזור להגדיר הרשאות לקבוצות משתמשים שונות.





ניצן גולדנברג

מזה 5 שנים בתחום בדיקות התוכנה, תפקיד נוכחי מהנדס בדיקות בחברת SeatGeek, המוביל הראשי של קבוצת המיטאפ TestIL, מרצה בקורסים לבודקי תוכנה וחבר בצוות המייעץ AB של ITCB®.

התממשקות (Integration) עם מערכות ניהול תקלות

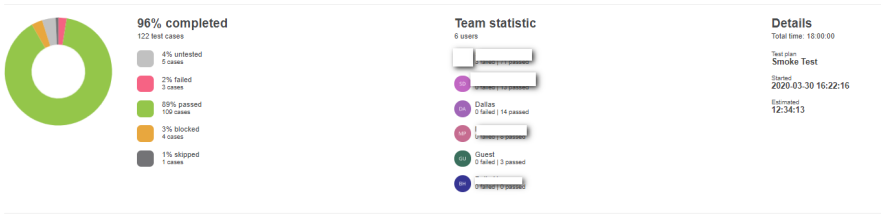
ניתן לפתוח תקלות (Bugs) ישירות במערכות ניהול תקלות המועדפות והנפוצות בשוק. הגשת דוח תקלות לצוות הפיתוח או המנהלים עם הוראות מפורטות כיצד לשחזר את התקלות.

המערכת משתלבת עם:

- ✔ Jira integration
- ✔ Redmine integration
- ✔ YouTrack integration
- ✔ Slack integration
- ✔ Github integration

דוחות

לאחר ביצוע הריצות, ניתן לייצר דוחות ריצה לכל פרויקט בנפרד

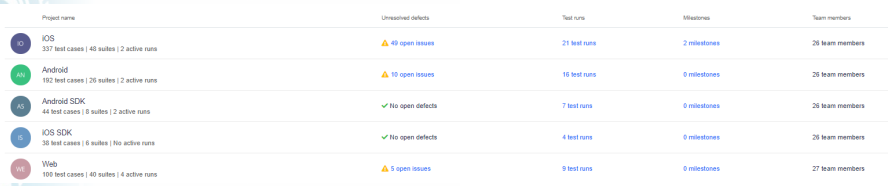


בדיקות API

ניתן להשתמש ב-REST API כדי לשלב את הבדיקות האוטומטיות (שימוש בכלי בדיקות חיצוניים) ושליחת תוצאות הבדיקות ל-Qase.

עבודה מול מספר פרויקטים בו זמנית

ניתן לפתוח במערכת מספר פרויקטים בו זמנית ולעבוד על כל פרויקט בנפרד.



יתרונות וחסרונות

היתרונות הבולטים של הכלי

1. ממשק משתמש נוח וקל לתפעול
2. עבודה בענן
3. המסלול הבסיסי חינמי לחלוטין (קישור לעוד מסלולים)
4. ייבוא/ייצוא תסריטי בדיקה מ:
   TestLink, TestRail, JSON, XML,
   Test Management for Jira SquashTM & LeanTesting

החסרונות הבולטים של הכלי

1. אין קהילה עשירה - כאשר יש קהילה גדולה לכלי מסוים אז תיקוני באגים נעשים מהר ויש מענה מהיר לשאלות. ל-Qase אין קהילה גדולה של משתמשים
2. אין אפשרות ליצור קשר עם נציגי החברה טלפונית או לפני התחברות למערכת. אם הייתי מעוניין לשאול שאלות או להתייעץ עם נציגי החברה אז אין אפשרות בכך לפני שמתחברים למערכת (במסלול החינמי)
3. הדוחות מאוד מינימליים - הדוחות לא נותנים למשתמש יותר מדי מידע לגבי הריצות

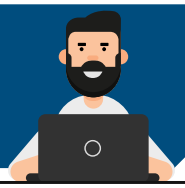
לסיכום

במידה ואתם מעוניינים לנסות את הכלי ולראות אם הוא מתאים לצרכים שלכם, הכלי ניתן לשימוש ניסיון ללא הגבלה.

לאחר שהתנסתי בשימוש הכלי, הרגשתי שזה אחד מכלי ניהול הבדיקות הבולט יותר בשוק הנוכחי. Qase הוא כלי פשוט, יצרני ויעיל המספק ממשק קל לשימוש אשר הופך את חיי המשתמש נוחים יותר ומזניקים אותו ישר קדימה.

זכרו תמיד, Qase שימושי לניהול מבחני בדיקות, אך בסופו של יום זו העבודה הקשה, החכמה והאיכותית שלכם, שבסופו של דבר מציבה אתכם בראש.

ציונים:
מענה לצרכים של החברה: 8/10
נותן למשתמש חווית שימוש טובה: 9/10
תמיכה וקהילה: 3/10
סה"כ: 6.6



בדיקות נגישות מקלדת

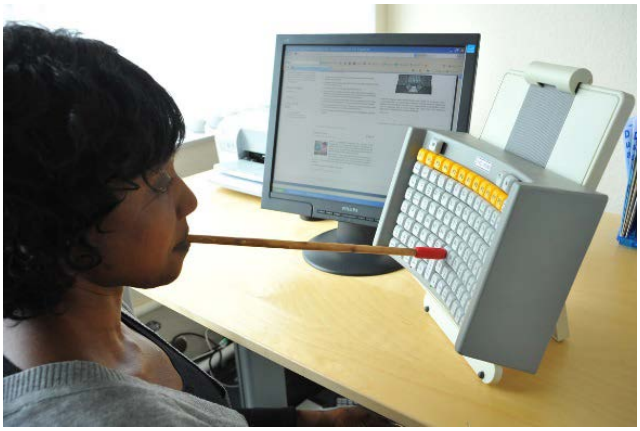
בטור זה נעסוק בבדיקות נגישות מקלדת. נבין את חשיבות הנושא, נכיר כמה סוגי משתמשים, נלמד מה בדיוק בודקים ונבחן מספר כלים שיכולים לעזור לנו.

מי צריך מקלדת?

ישנם סוגים רבים של משתמשים אשר אינם יכולים להשתמש בעכבר.

ניתן לחלק אותם לשתי קבוצות עיקריות:

1. משתמשים עם מגבלה מוטורית אשר מונעת מהם להשתמש בעכבר: זה יכול להיות רעידות בידיים, שיתוק ברוב הגוף או היעדר גפיים. משתמשים אלו נעזרים בטכנולוגיות מסייעות שונות. לדוגמה: יש הנעזרים בכפתורי לחיצה גדולים שהם מפעילים עם הזזות ראש, ויש כאלה עם מקלדת מיוחדת שהם מפעילים בעזרת מקל בפה. בכל מקרה, פתרונות שונים אלו כולם מבוססים על אותו ה-API של מקלדת רגילה ועל כן בבדיקות שלנו נסתפק בבדיקות בעזרת מקלדת.



2. משתמשים עיוורים אשר אינם מסוגלים לראות את סמן העכבר, משתמשים במקלדת (בין אם רגילה או מקלדת ברייל), יחד עם תוכנת קורא מסך. לבדיקות אלה נשתמש במקלדת יחד עם קורא מסך.

למה זה חשוב?

כפי שראינו, יש מגוון רחב של משתמשים אשר מושפעים מאוד מרמת נגישות המקלדת של המוצר. יתרה מזאת, כיוון שבנגישות מקלדת אנחנו מדברים על ביצוע פעולות בתוכנה ולא רק תפיסה או הבנה של התוכן, אלו בדרך כלל בעיות חמורות יותר.

מבחינת תעדופי באגים, אני ממליץ להתייחס לבאג שמונע ממשתמש מקלדת לבצע פונקציונליות בסיסית של המוצר, בדיוק באותה חומרה כאילו היה זה באג שמונע ממשתמש עכבר לבצע את אותה פונקציונליות.

מה בודקים ואיך?

ניווט

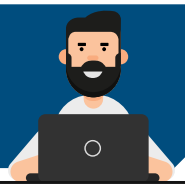
1. **יכולת ניווט לכל רכיב** - כאשר אנו מנווטים עם מקש ה-Tab, נוודא שיש אפשרות להגיע לכל רכיב אינטראקטיבי ולהפעיל אותו בעזרת המקלדת בלבד. שימו לב, מדובר רק ברכיבים אינטראקטיביים כגון קישורים, כפתורים, שדות בטפסים ופקדים שונים.
2. **פוקוס מסומן באופן ברור** - סימון הרכיב עליו נמצא הפוקוס, בין אם על ידי מסגרת פשוטה מסביבו, או שינוי עיצובי אחר. היעדר הגדרות עיצוביות לפוקוס מקלדת מקשה מאוד על ההתמצאות בעמוד, ובפרט על ביצוע יתר הבדיקות. שימו לב שמדובר כאן בנגישות עבור משתמשי מקלדת אשר מסוגלים לראות את הסימון ואינו רלוונטי מבחינת משתמשים עיוורים עם קורא מסך.
3. **יכולת ניווט החוצה מכל רכיב** - יש לוודא שאין מה שנקרא "מלכודת מקלדת" - מצב שבו נכנסת על ידי Tab לאזור שאי אפשר לצאת ממנו יותר בעזרת המקלדת בלבד.
4. **סדר ניווט הגיוני** - סדר הניווט צריך להיות תואם את סדר הקריאה של השפה. למשל, בעברית מימין לשמאל ומלמעלה למטה. שגיאה של סדר ניווט עם כפתור ה-Tab יכולה להצביע על אחד מהשניים:
   - הרכיב לא נמצא במקום הנכון מבחינת ה-Document Object Model (להלן DOM) ואז החזרה שלו תפתור זאת בקלות.
   - נעשה שימוש ב-tabindex עם ערך חיובי על מנת להקדים את הרכיב בסדר הניווט. במקרה זה יש רק להסיר את ה-tabindex.
5. **אין עצירות מיותרות** - כיוון שעבור המשתמשים, הפעולה המקבילה ל-Tab עלולה להיות כרוכה בקושי פיזי מהותי, יש לכבד זאת ולוודא שכל עצירה מוצדקת. דוגמה לשתי בעיות שכיחות הן כאשר יש עצירה על רכיב שאינו אינטראקטיבי כלל (אך הוסיפו לו "tabindex="0") או כאשר יש מספר רכיבים עם

אלון פרידמן ויסברד

אחראי בדיקות בצוות הנגישות של Wix.

בעל רקע של 15 שנים בבדיקות תוכנה.

נשוי + ילד + חתולה.



אותו קישור בזה אחר זה. יש לצמצם כפילויות אלה על מנת לאפשר ניווט יעיל ומהיר.

6. **מעקף בלוקים** - בעמוד שיש בו תפריט ראשי בתחילת העמוד, רצוי מאוד לתת קישור המאפשר למשתמש לדלג על כל קישורי התפריט. קישור כזה צריך להיות הראשון שאליו מגיע משתמש מקלדת אשר פתח את העמוד. הקישור יכול להיות מוסתר עד לרגע שהוא מקבל פוקוס.

פונקציונליות

7. **יכולת תפעול מלאה** - לאחר שוידאנו את אפשרויות ההגעה לרכיבים השונים, יש לבדוק את כל הבדיקות הפונקציונליות בעזרת מקלדת. למשל, יש לבדוק שהקישורים נפתחים עם Enter ושהכפתורים עובדים עם Enter או Spacebar. עבור רכיבים מותאמים אישית ניתן להיעזר במדריך שיטות הכתיבה של WAI-ARIA המפרט את ההתנהגות המקלדת המומלצת ברכיבים השונים.

8. **ניהול העברת פוקוס** - יש לוודא שהפוקוס עובר לאן שהוא אמור. למשל, אם פותחים modal, אז יש להעביר מיידית את הפוקוס לתוכו ולוודא שלאחר מכן, הפוקוס יהיה "נעול" בפנים עד לסגירתו על ידי כפתור סגירה או על ידי שימוש ב-Esc. ב"נעול" הכוונה ש-Tab מהרכיב האינטראקטיבי האחרון ב-modal יוביל אל הרכיב הראשון ב-modal, ולהפך: Shift+Tab מהרכיב הראשון אל האחרון. גם הגדרות אלו ניתן למצוא במסמך שיטות הכתיבה של WAI-ARIA הנ"ל.

תוכן נוסף

9. **גילוי תוכן נוסף** - יש לבדוק שיש אפשרות להצגת כל המידע בעזרת המקלדת בלבד. לצורך בדיקה זו, יש להשתמש בעכבר לזיהוי מקומות בהם יש צורך בריחוף עכבר או גלילה על מנת לקבל את כל התכנים.

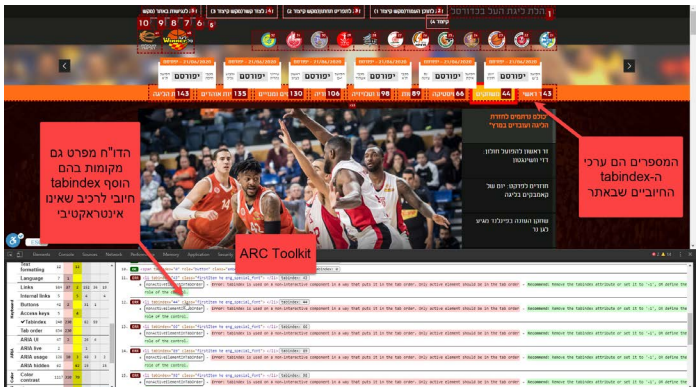
בדיקה עם קורא מסך

10. את כל הבדיקות הנ"ל יש לבצע פעם נוספת עם קורא מסך (רצוי NVDA על Windows). הסיבה לכך היא שלעתים קרובות יש התנגשות בין הגדרות המקלדת של המוצר לבין הגדרות המקלדת של קורא המסך, והדבר מוביל לכך שדברים שעבדו ללא קורא מסך לא יעבדו כשהוא מופעל (במיוחד עם NVDA).

כלי עזר בבדיקות מקלדת

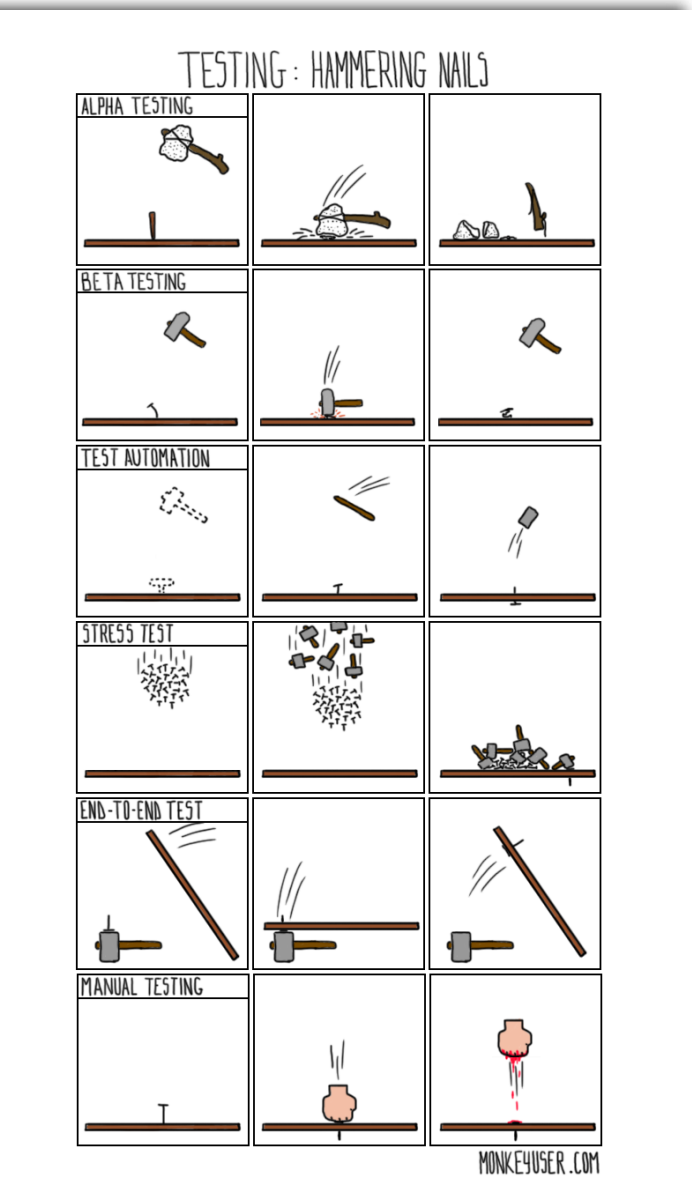
בגדול, את בדיקות המקלדת עושים בעיקר בבדיקות ידניות בעזרת מקלדת, אך הכלים הבאים יכולים לעזור:

1. התוסף aXe Developer tools (להורדת aXe לפיירפוקס, aXe לכרום) למשל יכול לזהות ולהתריע על קיום של רכיבים עם tabindex גדול מ-0.
2. כלי כמו Microsoft Accessibility Insights for Web יכול לעזור לצייר את סדר הטאבים.
3. לבסוף, תוסף נוסף לכרום בשם ARC Toolkit אשר בין היתר מאפשר: לצייר את סדר הטאבים, לזהות רכיבים שאינם אינטראקטיביים בסדר הטאבים וגם לסמן את הערך של כל הרכיבים בעלי הגדרת tabindex (ראו צילום מסך בהמשך).

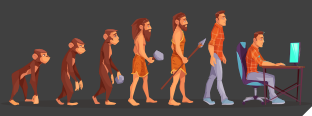


לסיכום

בטור זה דיברנו על בדיקות עם מקלדת, ראינו איך זה משפיע על סוגים שונים של משתמשים, למדנו את הנושאים השונים שדורשים בדיקה, והכרנו מספר כלי עזר (למרות שהכלי המרכזי היה ונשאר המקלדת עצמה).



אבולוציית התשתית בהרצת סלניום, מהרצה מקומית ועד לדוקר בענן | יואב לוינסון



מבוא

סלניום הינה ספריית קוד המיועדת לבדיקות ממשק משתמש בצורה אוטומטית והיא הפופולרית ביותר בתעשיה כיום. העניין הוא שבעוד שאנשים בוחרים בסלניום ככלי מרכזי לתשתית הבדיקות שלהם, הם לא שוקלים ומתכננים את השלב הבא בתהליך שהוא להחליט "כיצד לעשות בו שימוש", כלומר - כיצד לפרוס אותו, כיצד לבצע באמצעותו הרצות. החלטות אלו הן משמעותיות לא פחות מאשר ההחלטה באיזו תשתית נבחר. במאמר זה אציג שלל דרכי פריסה ושימושים בסלניום כאשר לכל שיטה יש יתרונות וחסרונות. אין שיטה אחת מתאימה באופן מושלם לכלל הארגונים ולכן כדאי שנכיר היטב את השיטות המרכזיות וכך נוכל לבחור מה הכי מתאים לארגון שלנו.

נקודת ההתחלה – הרצה מקומית

נתחיל בשיטה הפשוטה ביותר למימוש – הרצת הבדיקות ופתיחת הדפדפנים על המחשב שלנו, מחשבו של המפתח. אם תרשו לי לצלול רגע לשורת קוד בודדת (בשפת ג'אווה לצורך הדוגמה), הרי משפט המפתח הוא:

```
WebDriver driver = new ChromeDriver();
```

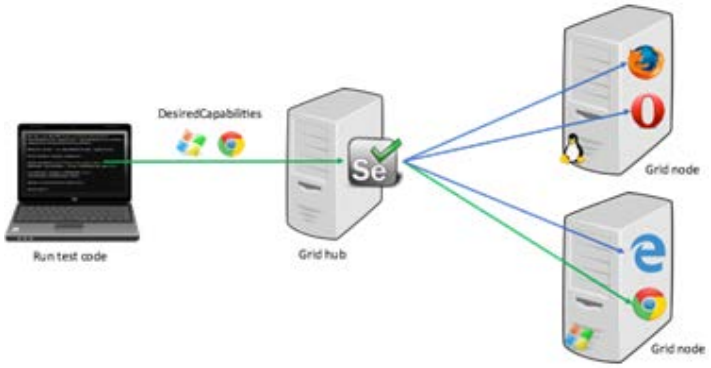
כלומר אנו גורמים לסלניום לפתוח את הדפדפן הנבחר (כרום, פיירפוקס, אדג' וכו') על המחשב שלנו, מהמקום בו הקוד שלנו רץ (תהה השפה בה נכתב אשר תהה) והוא מדבר ישירות מול הדפדפן, ללא גורמי תיווך ביניים, ללא שליחת הודעות למחשבים מרוחקים - מה שגורם לכך שההרצה בפועל מאוד מהירה. נוסף על כך היתרון המרכזי הוא הנראות - אנו יכולים לראות בקלות בכל רגע נתון מה הבדיקה מבצעת היות והדפדפן פתוח לנו מול העיניים. החיסרון הראשון שנבחין בו כאן הוא חוסר היכולת להריץ כמות גדולה באמת של בדיקות. השיטה עובדת נהדר כשמדובר במספר צנוע יחסית של בדיקות אבל אם מדובר בחבילה של מאות בדיקות, תהליך ההרצה עלול לקחת שעות ובזמן זה המחשב שלנו מושבת כי כל פעולה שלנו עלולה להפריע לתהליך הבדיקות. אמנם קיימת אפשרות של הרצה מקבילית, כלומר להריץ מספר בדיקות שונות בו זמנית אבל חשוב להבין שמחשב בודד מוגבל במשאבים ולכן קיימת הגבלה משמעותית על כמות הדפדפנים שניתן להריץ במקביל ולכן בשורה התחתונה מדובר בפתרון שאינו ניתן להרחבה (בתעשייה עושים שימוש במילה סקלבילי ומתכוונים למילה האנגלית scalable). חסרונות נוספים הינם חוסר יכולת לבדוק גרסאות שונות של דפדפנים. נכון, ניתן להתקין על המחשב האישי שלנו גם כרום וגם פיירפוקס ולבדוק את שניהם, אבל כיצד נתמודד עם דרישה לבדוק גרסאות שונות של אותו הדפדפן? למשל כרום בגרסאות 79 וגם 80? כיצד נתמודד עם דרישה לבדוק דפדפן ספארי (שרץ כיום רק על מחשב מק) במידה ויש לנו מחשב מפתח מבוסס ווינדוס?

היתרונות והחסרונות של **שיטת הרצה מקומית**:

יתרונות	חסרונות
הרצת בדיקות מהירה	פתרון שאיננו מתאים לכמות בדיקות גדולה
נראות פשוטה לתהליך הבדיקות	אין תמיכה בגרסאות שונות של מערכות הפעלה ודפדפנים
תצורת תשתית קלה לניהול	הפעלת הריצה דורשת מעורבות ידנית

צעד אחד קדימה Selenium Grid-

החבר'ה של סלניום מצאו פתרון יצירתי לסוגיית התמיכה בגרסאות שונות של מערכות הפעלה ודפדפנים והחליטו לקרוא לו: Selenium grid. הרעיון הוא ליצור רשת של מחשבי הרצה כאשר כל אחד יכול להיות מבוסס מערכת הפעלה שונה ולהכיל התקנות של דפדפנים שונים בגרסאות ספציפיות. הארגון מכין אוסף של מחשבים כאלה, אפשר לדמות זאת לתפריט של שלל המנות המוצעות במסעדה כלשהי וברגע האמת כאשר בדיקה מתחילה לרוץ, הלקוח (הקוד שלנו) ניגש למלצר של אותה המסעדה (שבפועל מחוץ לעולם הדימוי שלנו הוא מנגנון בשם hub בסלניום גריד שלנו) ומבקש ממנו מנה כלשהי מהתפריט (מחשב המתאים לצרכיו. למשל, אני מעוניין להריץ בדיקה על מחשב עם מערכת הפעלה חלונות 10 וספציפית דפדפן כרום גרסה 80). במידה ויש לנו את המנה הזאת בתפריט, המלצר (ה-hub), מבצע את הקישור וההתאמה בין הלקוח (קוד הבדיקה שלנו) לבין המנה שהזמין (המחשב הייעודי המתאים לו). ולמעשה כל פעולה שנבצע מתוך הקוד שלנו, תועבר למלצר והוא יעביר אותה למחשב הרלוונטי. אותו העיקרון תופס גם בכיוון ההפוך כאשר הדפדפן מחזיר תשובה לקוד שלנו. להלן תרשים הממחיש את עיקרון הפעולה:



ניתן לראות בתמונה כיצד הקוד שלנו מבקש מחשב ווינדוס ודפדפן כרום וה-hub מעביר את כל הפעולות לביצוע אל המחשב הייעודי.

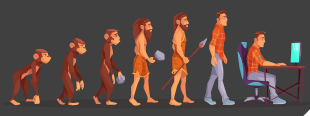
ובכן, המעבר לסלניום גריד דורש מאיתנו להרים שלל מכונות (פיזיות או וירטואליות) ובנוסף מחשב hub אליו כל אותם מחשבים יתחברו ויציעו את עצמם כפלטפורמה להרצה. נוסף על כך עלינו לבצע שינוי קל בקוד פתיחת הדפדפן שלנו. עבור המקרה שציינתי לעיל:

```
// what is the address of our hub (replace hubAddress with the actual hub address)
String hubURL = "http://hubAddress:4444/wd/hub";
// define what OS, browser and version we need
DesiredCapabilities capabilities = DesiredCapabilities.chrome();
capabilities.setPlatform(Platform.WIN10);
capabilities.setVersion("80");
// open the remote browser
WebDriver driver = new RemoteWebDriver(new URL(hubURL), capabilities);
```

יואב לוינסון

אחד המייסדים וה-CTO של חברת "פרקטיס", חברת הייעוץ וההדרכה להיי-טק. עוסק באוטומציה מזה 15 שנה כמפתח, ראש צוות ויועץ לסטארטפים, חברות הייטק וארגונים ביטחוניים בארץ ובעולם.

אבולוציית התשתית בהרצת סלניום, מהרצה מקומית ועד לדוקר בענן | יואב לוינסון



היתרון בשיטה זו ברור – הוא מאפשר לנו לכסות את כל מטריצת הבדיקות מבחינת סוגי מערכות הפעלה, סוגי דפדפנים וגרסאות. היות והדפדפנים נפתחים בפועל על מחשבים שאינם מחשבי המפתח, אנו גם חוסכים בצריכת משאבים במחשב של המפתח, מה שמאפשר לנו להריץ כמות גדולה של בדיקות באופן מקבילי. אחד החסרונות הבולטים בשיטה זו היא המורכבות של תהליך הרמת הסביבה, התקנת והגדרת הגריד על המכונות ובעיקר לגרום לכל המכונות לעבוד היטב יחדיו.

אם כך, בואו נסכם את היתרונות והחסרונות המרכזיים של **שיטת סלניום גריד**:

יתרונות	חסרונות
הרצת הבדיקות	הרצת הבדיקות איטית יותר בשל הצורך להעביר כל בקשה דרך מספר תחנות
תמיכה מצויינת במערכות הפעלה שונות ודפדפנים שונים	אין נראות נוחה לתהליך ריצת הבדיקות אלא אם נתחיל להתחבר מרחוק למחשבים המריצים
ניתן לשלב בקלות מנגנון CI כגון ג׳נקינס בכדי ליזום הרצות באופן אוטומטי (כגון nightly)	תצורת תשתית הדורשת מספר רב של מחשבים – כבד מבחינת משאבים וכן ניהול המכונות ועדכונן
	תהליך הרמת סביבת הגריד מסורבלת ומאתגרת

העתיד הוא במכולות, דוקר נכנס לתמונה

טכנולוגיית דוקר מלווה אותנו מאז 2013 וכיום, קשה לדמיין תהליך עבודה נכון ומסודר שלא עושה שימוש בכלי נהדר זה. למי שלא מכיר אסביר בקצרה במה מדובר - זוהי יכולת מובנית במערכת ההפעלה של לינוקס (אבל קיימות התאמות עבור מק וחלונות) להריץ יישומון מסויים באופן מבודד לחלוטין, כאילו (אבל רק כאילו) מדובר במחשב נפרד לחלוטין. נשמע קצת כמו מכונה וירטואלית? אולי, אבל שימו לב להבדל העצום הבא - ההרצה היא של האפליקציה והתלויות שלה בלבד (כגון ה-JVM של Java). כלומר אין צורך להריץ בפועל מערכת הפעלה שלמה נוספת ונפרדת. ההשלכה היא מאוד משמעותית - משאבי המערכת מוקצים אך ורק לקוד שמעניין אותנו ולא לתהליכים זוללי משאבים של מערכת הפעלה וירטואלית שרצה לה ברקע. הפער מתגלה כמשמעותי כשאנו מבינים שדפדפן כרום ממוצע באוטומציה צורך כ-2 ג'יגה זיכרון ובכדי להריץ מערכת הפעלה מודרנית מומלץ לתת לה לפחות 4 ג'יגה זיכרון משלה. כלומר במקום דפדפן בודד ומערכת הפעלה וירטואלית אחת שבפועל מיותרת לנו, יכולנו בעזרת אותם משאבים בדוקר להריץ 3 דפדפנים! משמעותי ביותר! ובגלל שהדפדפנים רצים באופן מבודד זה מזה בזכות דוקר, הם כלל לא מפריעים אחד לשני! שימו לב שמבחינת צריכת משאבים דנו עד כה רק בסוגיית הזיכרון ולא צללנו לסוגיות כגון זמן המעבד שמערכת ההפעלה הוירטואלית צורכת רק בשביל לעלות ולתפקד מבלי קשר לדפדפן עצמו, או על עדכונים שמערכת ההפעלה זקוקה לה באופן שוטף, או המקום בדיסק הקשיח שהיא תופסת... כלומר שילוב של יכולות דוקר ושל גריד בעצם עוזרים לנו להפחית משמעותית את כמות המכונות להן נזדקק וזה משפיע באופן ישיר על העלויות וכאב הראש של תחזוקת המכונות הללו.

איזה כייף שהחבר'ה של סלניום מכירים ביעילות של דוקר ומפיצים לנו באופן רשמי הפצות ייעודיות של מכולות דוקר עם סלניום עבור הדפדפנים המרכזיים! ובעזרת הפשטות של דוקר, אפשר תוך דקות ספורות להרים סביבה שכזו מבלי להסתבך יותר מדי! ומבחינת הקוד שלנו? אין שינוי מהותי מאשר עבודה עם גריד רגיל!

נסכם את היתרונות והחסרונות המרכזיים של **שיטת דוקר**:

יתרונות	חסרונות
הרצת הבדיקות הפכה סקלבילית	הרצת הבדיקות איטית יותר בשל הצורך להעביר כל בקשה דרך מספר תחנות
ניתן לשלב בקלות מנגנון CI כגון ג׳נקינס בכדי ליזום הרצות באופן אוטומטי (כגון nightly)	אין נראות נוחה לתהליך ריצת הבדיקות אלא אם נתחיל להתחבר מרחוק למחשבים המריצים
תצורת תשתית יעילה מבחינת משאבים וקלה הרבה יותר לניהול	תמיכה רק במערכת הפעלה לינוקס עליה ניתן להריץ רק את הדפדפנים המובילים
תהליך הרמת הסביבה פשוט ומהיר	

כשהמספרים הולכים וגדלים - הרצת גריד בענן

מכירים את הביטוי: "אין דבר כזה ענן, מדובר במכונה של מישהו אחר"? ובכן היתרון בכך טמון בעובדה שאם המכונה היא אכן של מישהו אחר אזי תמורת סכום כסף מסוים, גם כאב הראש של רכישת הציוד, ההתקנה, התחזוקה והקינפוג יכול להיות של מישהו אחר! לעבודה עם ענן באופן כללי יש יתרון של סקלביליות מאוד גדולה ובהתאם לצורך. למשל, אם ביום-יום אנחנו מעוניינים להריץ בדיקות הדורשות שלושה מחשבים שונים עבור הדפדפנים שלנו ורק פעם בשבועיים בעת הורדת הגרסה נרצה להריץ מסה גדולה יותר הדורשת למשל עשרה מחשבים, ללא הענן היינו צריכים להחזיק באופן שוטף את אותם עשרה מחשבים. בעידן הענן אנחנו יכולים לבחור בכל רגע נתון כמה ואילו מכונות אנחנו רוצים ובלחיצת כפתור לשחרר אותן לחופשי מבלי לשלם עליהן עד אשר נזדקק להם שנית.

חשוב לציין שמעבר לבחירה של ספק שירותי הענן עימו נרצה לעבוד, קיימות גם טכניקות שונות להרמת תשתיות סלניום בסביבות הענן של כל ספק ולכל אחת מהן יתרונות וחסרונות.

במקרה זה ניקח את הדוגמה הקודמת של הרצה באמצעות דוקר וניקח אותה לענן עם שדרוג קל:

בדוגמה זו נעבוד עם כלי CI כגון ג׳נקינס שתומך בעיקרון של שרת מרכזי קבוע (קרוי master) ושרתי slave הנוצרים בענן באופן אוטומטי כאשר אנו רוצים להריץ job כלשהו ונמחקים מיד לאחר מכן. אנו יכולים להגדיר מראש כיצד נראה שרת slave ומה מותקן עליו. נגדיר ששרת זה יכיל את כל הנתונים הרלוונטים בכדי להיות מסוגל להריץ ולקמפל את קוד האוטומציה שלנו וכן יידע להרים את רכיב ה-hub של סלניום גריד ודפדפנים שונים לפי הצורך - כל זאת על בסיס דוקר כפי שראינו בדוגמה הקודמת ועל אותה מכונת slave.

אבולוציית התשתית בהרצת סלניום, מהרצה מקומית ועד לדוקר בענן | יואב לוינסון

התהליך ייראה כך:

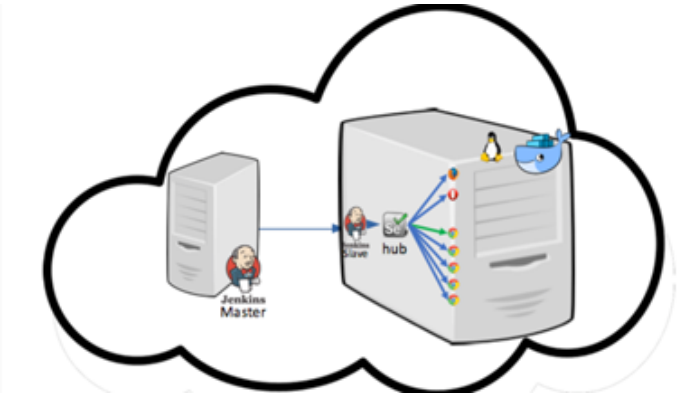
1) שירות ה-CI שלנו (בפועל שרת ה-master הקבוע) יחליט שהגיע הזמן להריץ job מסוים

2) שרת ה-master ייגש לספק הענן וירים שם שרת slave זמני לפי ה-template שהגדרנו מראש

3) שרת ה-slave יתחיל להריץ את הjob: ימשוך, יקמפל ויריץ את קוד האוטומציה העדכני ביותר

4) ההרצה כוללת התחברות לשירות ה-hub **המקומי** והפניית הבדיקות לאחד מדפדפני הדוקר **המקומיים**

5) כאשר הבדיקה תסתיים, ה-slave יעלה אל שרת ה-master באופן אוטומטי את תוצאות הבדיקות ושרת ה-slave ימחק ויוחזר לידי ספק הענן.



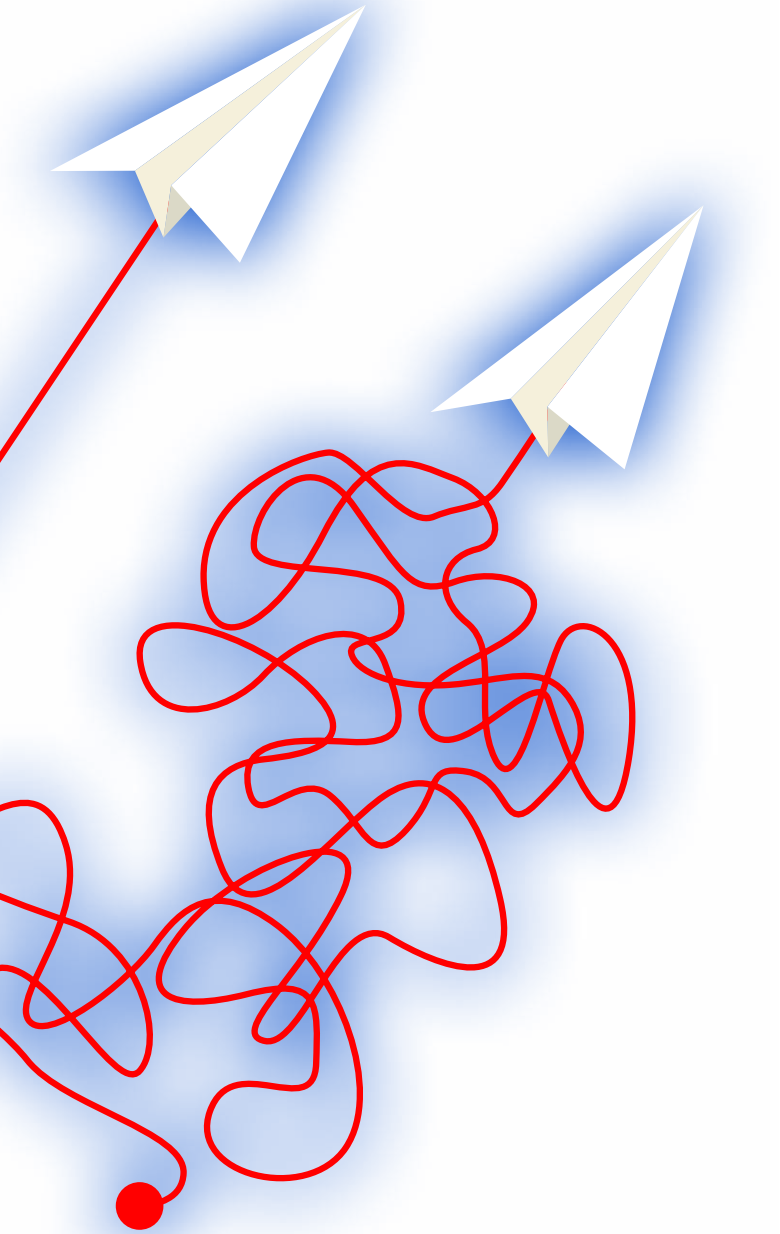
היתרון בהרצה שכזו בענן הוא סקלביליות משמעותית של הרצת הבדיקות שלנו - יכולת הכרחית בעידן ה-CI/CD אליו הרבה מאוד חברות שואפות וזאת כיוון שנוכל לפרק את האוטומציה שלנו ל-job-ים שונים ב-CI וכל job ירוץ על מכונה זמנית משלו. הקוד שלנו והדפדפן יפתחו באותה מכונה, דבר שחוסך מאיתנו שליחת הודעות מרובות הלוך ושוב דרך שרתים שונים. זה הופך את הבדיקה שלנו לעוד יותר מהירה! ומה לגבי חסרונות? ובכן, בשיטה זו יש עלויות נוספות של שירותי הענן שחשוב לקחת בחשבון וכן טרם פתרנו את הסוגיה בה אין לנו יכולת לראות באופן פשוט מול העיניים את ריצת הבדיקה. יש לכך פתרונות אבל נדון בכך בפעם הבאה.

אם כך, נסכם את היתרונות והחסרונות של **שיטת הרצת גריד בענן**:

יתרונות	חסרונות
הרצת הבדיקות מאוד סקלבילית	אין נראות נוחה לתהליך ריצת הבדיקות אלא אם נתחיל להתחבר מרחוק למחשבים המריצים
הרצת הבדיקות מהירה	תמיכה רק במערכת הפעלה לינוקס עליה ניתן להריץ רק את הדפדפנים המובילים
ניתן לשלב בקלות מנגנון CI כגון ג'נקינס בכדי ליזום הרצות באופן אוטומטי (כגון nightly)	המחיר גדל משמעותית עם נפח הבדיקות
תצורת תשתית יעילה מבחינת משאבים וקלה הרבה יותר לניהול	תהליך הרמת הסביבה איננו פשוט ומהיר

לסיכום

מעבר לבחירת הכלי שישמש אותנו כתשתית אוטומציה, למשל סלניום, חשוב גם לתכנן היכן ובאיזה אופן ירוצו הבדיקות שלנו. קיימות שיטות שונות לכך, ולכל אחת היתרונות והחסרונות שלה. ברוב המקרים תהליך האוטומציה שלנו יעבור סוג של אבולוציה - נתחיל מפשוט - מהרצות מקומיות ובהתאם לדרישות ולנפח האוטומציה וההרצות שילכו ויגדלו, נתקדם בסולם האבולוציה שהוצג כאן עד לשלב הכי נכון עבורנו. לא תמיד חייבים לשאוף להגיע לרמה הגבוהה ביותר, לעיתים דווקא רמות מסוימות תחתיה עדיפות ונכונות לפרויקט שלנו יותר. בהצלחה!



מה הסקרים אומרים?...

יאן ברון

מנהל בדיקות בחברת iMDSoft ישראל.

בעל תואר M.Sc במדעי המחשב

והסמכה בינלאומית של ISTQB® FL.

עם 35 שנות ניסיון בהנדסת תוכנה וניהול בדיקות.

תחומי עניין

מתודולוגיה וכלי פיתוח לאזורים שונים בבדיקות, בדיקות אוטומציה, ניהול מחזור חיי מוצר: מתודולוגיה וכלים.

התנדבות

ISTQB® – ראש ועדת סקרים, ITCB®, ועד המנהלים,

SiGIST ישראל – יו"ר תכנית הכנסים.

הפעם אני רוצה להציג בפניכם מעט ממצאים מהמהדורה ה-11 של דו"ח האיכות העולמי מאת Sogeti בהקשר עם Micro Focus.
התובנות התקבלו מראיונות עם 1725 מנהלי מערכות ומובילי טכנולוגיה בכירים מ-32 מדינות ו-10 תעשיות.
את הגרסה המלאה של הדו"ח תוכלו למצוא כאן.

Fig 1 Executive management objectives with QA and testing on a scale of 1-7, where 7 = very important

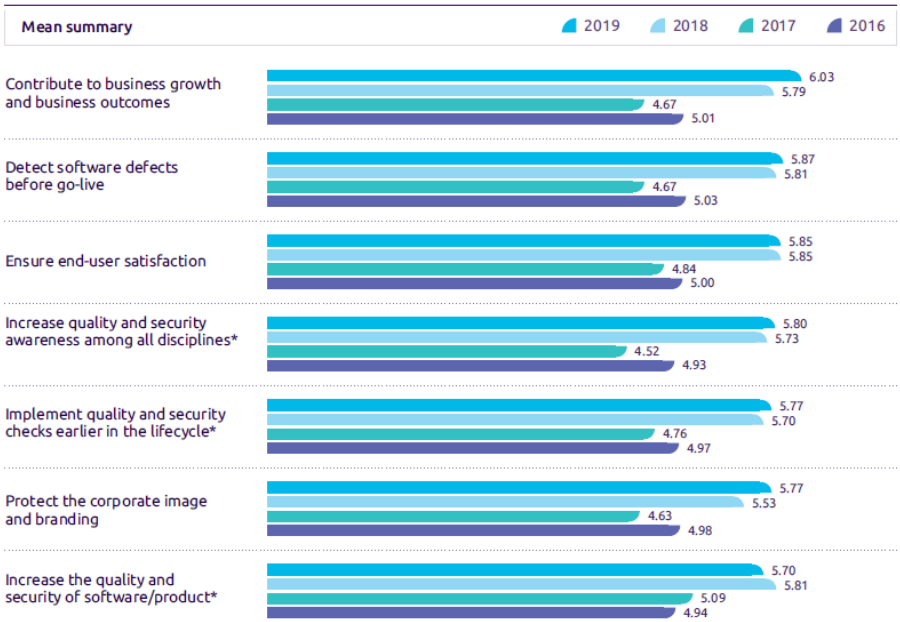
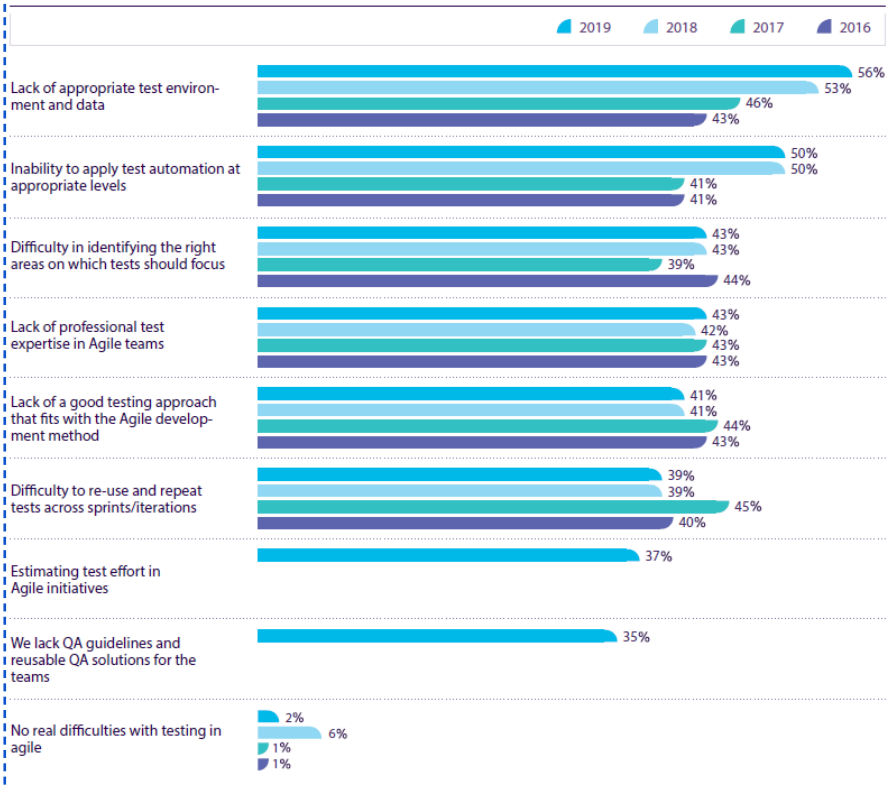


Fig 2 Challenges currently faced in applying testing to agile developments



מה הסקרים אומרים?...

Fig 3 Likely use of specific approaches to speed up and optimize testing in agile and DevOps developments

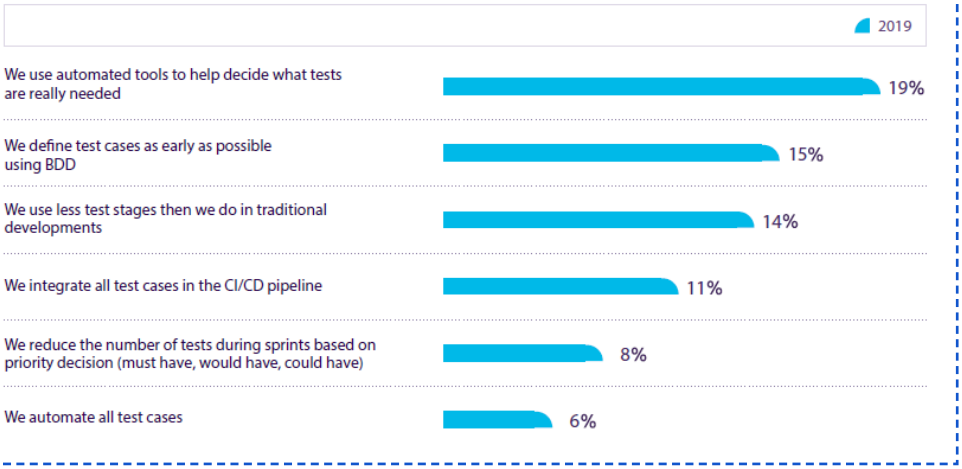


Fig 7 Main challenges in achieving desired level of test automation

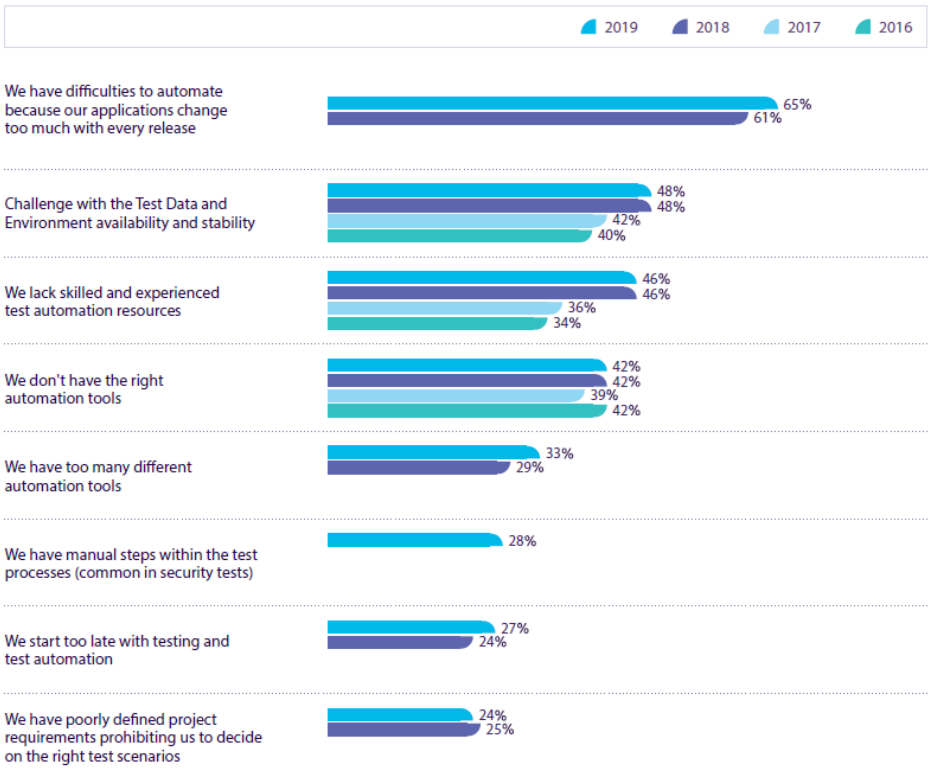


Fig 8 Projected business interest in automation techniques in the coming year

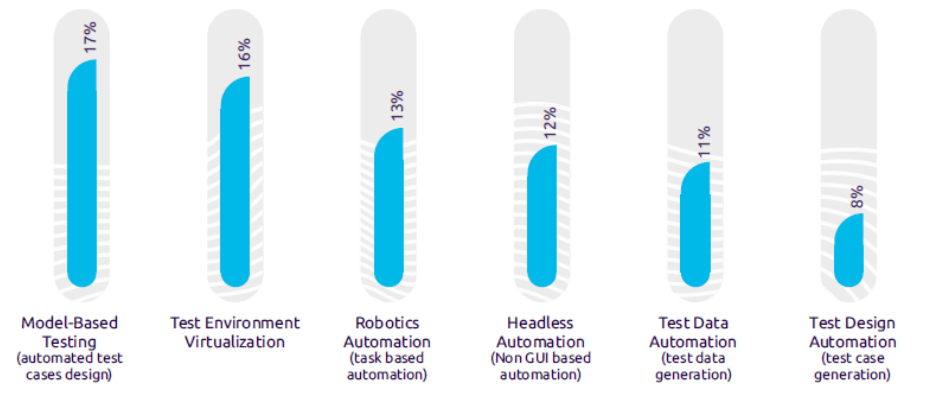
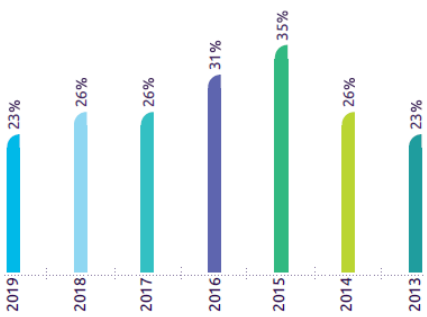


Fig 14 Proportion of total IT budget allocated to QA and testing (including testing processes, tools, and resource costs)



נגיסה מ-TestIL
מפגשים לבודקי תוכנה

בגיליון הזה החלטתי לשנות קצת את הכיוון של סקירת האירועים בשל מצב הקורונה בארץ ובעולם ולתת סקירה של מקורות לימוד אונליין - פודקאסטים, ווובינרים ערוצי סלאק ועוד.

ניצן גולדנברג

מזה 5 שנים בתחום בדיקות התוכנה, תפקיד נוכחי מהנדס בדיקות בחברת SeatGeek, המוביל הראשי של קבוצת המיטאפ TestIL, מרצה בקורסים לבודקי תוכנה וחבר בצוות המייעץ AB של ITCB®.

ערוצי סלאק

MOT (Ministry Of Testing)

אחת הקהילות הגדולות לבודקי תוכנה. הקהילה הוקמה בשנת 2007 וכיום מונה אלפי בודקים ברחבי הגלובוס.

יש לקהילה מרכז מידע ענק בשם The Dojo הכולל מאמרים מקצועיים הנכתבים ע"י בודקים מכל העולם, פודקאסטים, הרצאות וקורסים. רוב התוכן פתוח רק למנויים משלמים, אך גם באזור החינמי יש לא מעט.

ל-MOT יש ערוץ סלאק הפתוח לכלל הבודקים. הערוץ משלב בתוכו שלל רחב מאוד של נושאים. ערוץ "רשמי" (קישור) וערוץ של "קהילת הבדיקות" תחת הכותרת Testers.chat שנקרא גם Testing Community וגם בשמו הישן - testers.io.

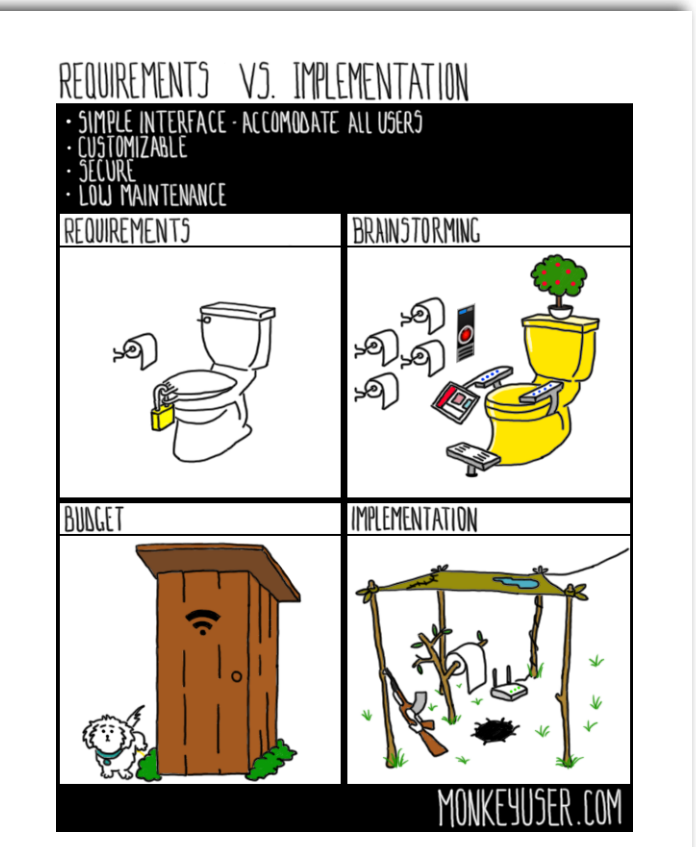
פודקאסטים

- **ABTesting**
  פודקאסט דו-שבועי בו מדברים Alan Page ו-Brent Jensen על בדיקות, פיתוח תוכנה ומגוון נושאים צדדיים. לפודקאסט יש גם ערוץ סלאק בו הקהילה יכולה להרחיב את הדיון ולשלוח שאלות למנחים.
- **abstracta**
  חברה למתן שירותי בדיקות שמעלה פודקסטים מעניינים.
- **Testing one on one**
  פודקאסט של Rob Lambert & Joel Montvelsky שמדברים על כל מה שקשור לעולם הבדיקות.
- **Qualitest - The testing show**
  בכל פרק (שיוצא פעם בחודש), פאנל של בודקי תוכנה מנוסים (חלקם קבועים, וחלק מתחלפים) דן בנושא - מנושאים ספציפיים כמו בדיקות אבטחה או ביצועים ועד לשינויים ארגוניים ותכנון קריירה לבודקי תוכנה.
- **Testers Island Discs**
  פודקאסט בהנחיית Mark Winteringham - כאן יש לנו הזדמנות לשמוע על בודקי תוכנה רבים ולהכיר אותם ואת נושאי הבדיקות שמעסיקים אותם תוך האזנה לקטעים מחמישה שירים שהם בחרו.
- **Testing Podcast**
  אתר שמרכז בתוכו מגוון פודקאסטים מתחום הבדיקות, בתוכם גם חלק גדול מהפודקאסטים ברשימה זו.
- **Test&Code**
  פודקאסט על פייתון, עם דגש מסויים על בדיקות.
- **RBCS**
  סדרת פודקאסטים מאת רקס בלאק על מגוון נושאים רחב.
- **STP (Software Test Professional)**
  שיחות עם מרצי STPCon, בדרך כלל סביב הנושא עליו הם ידברו בכנס הקרוב. אמנם המטרה המרכזית היא לעניין את המאזינים להגיע לכנס, אך התוכן עומד היטב בפני עצמו.
- **Test Guild**
  תחת המותג הזה נמצאים שלושת הפודקאסטים של Joe Calantonio - TestGuild, פודקאסט כללי על בדיקות, בעיקר כלי אוטומציה שונים. PerfGuild - פודקאסט סביב בדיקות ביצועים, וSecurityGuild פודקאסט על בדיקות אבטחה. רוב הפרקים הם פרקי ראיון עם מומחה בתחום מסויים, מה שמאפשר היכרות טובה עם הנושא או הכלי המדוברים.

פודקאסטים

- **TestBash**
  MOT מקיימים מיטאפים בכל העולם ובעקבות המצב כיום המיטאפים מתקיימים און ליין כוובינרים.
- **EuroSTAR Huddle**
  אתר תוכן שנוצר סביב כנסי EuroStar ו-UKStar, האתר מציע תוכן מגוון - הרצאות, וובינרים, ספרונים אלקטרוניים, מאמרים ועוד.
- **יוני פלנר**
  מדריך בתחום האוטומציה, מרצה בכנסים בארץ ובעולם, מטמיע ומייעץ בתחום האוטומציה.
  ניתן למצוא מאמרים, הדרכות, וובינרים ומיטאפים.
- **מטריקס**
  חברת שרותי טכנולוגיות אשר הוקמה בשנת 2001. ליאור כף סמנכ"ל טכנולוגיות, בדיקות ואוטומציה מארגן כל זמן מה וובינרים ומיטאפים בנושאי בדיקות תוכנה לכלל קהילת הבודקים.
- **ALMtoolbox**
  החברה בניהולו של תמיר גפן מעלה כל זמן מה וובינרים ומיטאפים בתחום של כלי בדיקות וכלי פיתוח. הנושאים שהועלו היו הכרות עם GitLab, Xray ועוד שלל נושאים מעניינים הקשורים לכלים בעולם ההייטק.

אם יש לכם עוד רעיונות למקורות מידע, אירועים או פודקסטים אתם מוזמנים לשלוח לנו מייל למערכת המגזין ונשמח לשתף בגליונות הבאים: Info.testingworld@gmail.com.



עורכי הגיליון

טל פאר

בעל ניסיון של יותר מ-20 שנים כבודק ומנהל בדיקות במגוון חברות וטכנולוגיות במודלי פיתוח שונים. כיום טל יועץ ומדריך בדיקות עצמאי.

טל חבר ב-ITCB® וגזבר הארגון העולמי ISTQB®.

עלי טאהא

מהנדס B.sc, מוביל טכני לאוטומציה בחברת SeatGeek בעל 6 שנות ניסיון בבדיקות והקמת תשתית אוטומציה. לשעבר ראש מחלקת אוטומציה בחברת STMS,בעל ידע מעמיק בתקשורת ובמודל 7 השכבות.

אסתר צבר

מהנדסת (M.Sc.) בעלת 23 שנות ניסיון בפיתוח ובדיקות תוכנה, מתוכן 11 שנים בניהול QA בחברות ECI ו BMC - ובנוסף חברה ב-Advisory Board של ITCB® הארגון הישראלי להסמכת בודקי תוכנה. בתשע השנים האחרונות – יזמית ומנהלת של AQA המכשירה ומשלבת אנשים עם תסמונת אספרגר בעבודה בהייטק כבודקי תוכנה.

תמרה מוסונובה

בודקת תוכנה ואינטגרציה בחברת Varonis.
בעלת תואר בתקשורת וקולנוע והסמכות פיתוח אפליקציות Web של מיקרוסופט, כך משלבת חשיבה יצירתית ואנליסטית בעבודה. בונה אתרים ועורכת סרטים כתחביב. שחקנית כדורשת בליגה ארצית, תופסת כדורים ובאגים מקצועית.

שי ביטון

בעל 12 שנות ניסיון בפיתוח אוטומציות ובדיקות.

עובד כיום ב-Qwilt.

רוביק סביאנץ

בודק תוכנה, נמצא בתחום כ-3 שנים את דרכו התחיל בחברת CARAMBOLA בה הוא נמצא גם היום.בשנה וחצי האחרונות מחזיק את כל מערך הבדיקות של החברה כבודק יחיד. בזמן הפנוי - ספורט, טיולים ומחשבים.

משה מאמיה

בעל 17 שנות ניסיון כמהנדס, מתוכן מעל עשר שנות ניסיון ניהולי, מתמחה בפיתוח אוטומציה ובבדיקות ביצועים. עובד מעל 5 שנים ב-HP כמנהל קבוצות QA ו-DevOps. חבר מייעץ למועצת מנהלים של ISTQB® ומרצה בפקולטה להנדסת תוכנה במכללת SCE.

אפרת וינברג

עוסקת בבדיקות תוכנה קרוב ל-20 שנה. עבדה במספר ארגונים בתפקידי בדיקות וניהול בדיקות. בשנים האחרונות עוסקת בפיתוח והוראת קורסים בבדיקות תוכנה ונושאים נוספים.

עמית ורטהיימר

בודק תוכנה ב-Deep Instinct.

שירה נוסבוים

הייטקיסטית ואמא במשרה מלאה, בדקות הבודדות שנשארות ביום בלוגרית אפייה.

בעלת תואר ראשון במדעי המחשב ובכימיה, מעל 10 שנות ניסיון כמפתחת תשתיות אוטומציה וכלים אוטומטיים בחברות גדולות, בסטארטאפים שונים בתעשייה במגוון תחומים.

מובילת תחום, מרצה ומפתחת קורסי אוטומציה.

בשבילה החיים זה לא מספיק.

רחל ברוך

הנדסאית תוכנה, לפני 3 שנים לאחר הפסקה של שנים מעולם ההייטק חזרה לעולם התוכנה בכל הכוח. נהנית להיות בצד הבודק עם חשיבה של מפתחת. אין יום שהיא לא לומדת משהו חדש בעולם התוכנה.