**Code Retreat**

תרגיל שבו הם מתחלקים לזוגות/שלישיות וכותבים את המשחק Game of Life בדרכים שונות. התרגיל מתחלק לשלבים, כאשר בכל שלב יש הגבלות שונות שמכריחות אותם לעבוד בדרך מסוימת. בין כל שלב לשלב מכים בגונג.

ב-retrospective שאחרי כל session שואלים שאלות מכווינות ומגיבים למה שהחניכים אומרים. הרבה ממה שעובר כאן זה כתגובה לחניכים, פחות כהכוונה נוקשה. בסוף זורקים קצת עצמות לקראת השלב הבא (תוך ניצול הרמות להנחתה באדיבותם של החניכים החכמים).

מטרות המופע:

1. להתנסות בדרכי עבודה שונות
2. לפתוח דרכי חשיבה שונות לפיתוח תוכנה

**הקדמה (כרבע שעה):**

Code Retreat:

* מיני "היסטוריה"
  + אפשר למצוא באתר המצורף
* מטרות
  + יום שלא כותבים בו כדי לסיים
  + לכתוב קוד בשביל האימון
  + לשים דגש על עקרונות ולנסות כל מיני דברים מעניינים כדי 'לצאת מהקופסה'

4 החוקים של דיזיין פשוט (עובד, DRY, ברור וקטן).

מה זה Game of Life

הם חייבים לעבוד ב-pair programming לכל אורך התרגיל (בכל הסבבים).

הם יכולים לבחור לעצמם את שפת העבודה (Java, Python, C,C++,JavaScript ), ואת מתודולוגיית העבודה (TDD למשל).

בין כל שלב לשלב מוחקים את הקוד. לא מותירים לו זכר! לא עובדים עם git.

**שלב ראשון (כ-40 דקות):**

החניכים כותבים את המשחק Game of Life בסגנון חופשי. לא מדריכים אותם.

**חיתוך על השלב הראשון (כרבע שעה):**

שואלים אותם מה כל אחד עשה, בדגש על:

* Grid vs Cell – איך מייצגים אותם? באחריות מי לבדוק את השכנים?
  + מי השתמש במחלקות לעומת פרימיטיבים
  + על כל דיזיין – איך ידעו שזה מה שצריך? יתרונות וחסרונות לכל דבר
* מי כתב באיזו שפה
* מי עשה טסטים (מי עשה TDD)
* מי תיעד
* דברים אחרים מעניינים שאולי ניסו?

**שלב שני (כ-40 דקות):**

בשלב זה שמים דגש על primitive obsession – כלומר, אסור לאף פונקציה/מתודה שלהם להחזיר טיפוס פרימיטיבי. בנוסף, ניתנת המלצה לעשות שתי מחלקות לכל תא – תא חי ותא מת.

**חיתוך על השלב השני (כ-10 דקות):**

שואלים מה כל אחד עשה:

* מה חשבו על הדיזיין החדש?
  + האם יהיה יותר גמיש לשינויים? Brian's Brain – הגירסה עם תא זומבי: האם היה קל לשנות את החוקים?
* שוב הסוגייה של האם תא צריך להכיר את השכנים שלו
* האם שינו שפה? בכל מקרה להמליץ להם לנסות שפות חדשות.
* האם כתבו טסטים? בכל מקרה להמליץ להם לכתוב טסטים...

**שלב שלישי (כ-40 דקות):**

כאן שמים דגש על פונקציות קצרות (3 שורות בפייתון, 4 בג'אווה), ומחלקות קצרות (עד 3 פונקציות).

ויתרנו להם על מגבלת if (קצת כי שכחנו, קצת כי זה שלב קצת מוקדם וגם ככה היה להם די מפסטן). בנוסף, פינג-פונג (אחד כותב טסט, השני גורם לו לעבור וכותב את הטסט הבא, ומתחלפים).

**חיתוך על השלב השלישי (כ-10 דקות):**

שואלים:

* מה יצא בהינתן המגבלות?
* מי עשה מערך? מי עשה מחלקות?
* מאיזה טסט התחילו? דוגמאות.
* באמצע השלב להזכיר להם את ארבעת העקרונות -- מה גילו? איך היה מצבם ביחס לכללים?

מדברים על TDD באופן כללי: הגדרה (שלושת האיסורים), איך מתהווה דיזיין תוך כדי TDD.

**שלב רביעי (כ-40 דקות):**

אחד כותב טסטים, והשני מממש. אסור לדבר. מי שמממש צריך לכתוב את הדבר הכי פשוט ומרושע כדי לגרום לטסט לעבור.

**חיתוך על השלב הרביעי (כ-10 דקות):**

שואלים:

* איך היה? האם הדיזיין התכנס?
* איך היה לתקשר באמצעות קוד?
* שמות לטסטים: דוגמאות מהחניכים
  + רוצים משהו שמתאר את ההתנהגות
* כמה זמן לקח בין שהקוד היה ירוק, לבין זה שהוא עוד פעם היה ירוק (מחזור TDD אחד של red->green->refactor)? דיבור על time to green -- הכנה לשלב הבא.

זו בעצם אנלוגיה לחיים האמיתיים, רק בצורה מוגזמת. אפשר להגיע לקוד קיים, להרוס אותו ושעדיין הטסטים יעברו, רק כי הטסטים לא טובים. קשה לכתוב טסטים טובים.

הדגמה קצרה של hg לקראת babysteps בשלב הבא.

**שלב חמישי (כ-40 דקות):**

Babysteps: עובדים באיטרציות TDD של שלוש דקות. אם לא מספיקים להגיע לירוק, עושים revert.

**חיתוך (כ-5 דקות):**

שואלים:

* למי יצא לעשות revert-ים? למה?
* האם הצליחו להגיע יותר לקראת משהו עובד לעומת פעמים קודמות

**שלב שישי (כ-40 דקות):**

Freestyle עם המלצות לשיגועים: לוח אינסופי, טופולוגיות, ריבוי מימדים, לוח משושה

**סיכום (כ-20 דקות):**

איך היה בהשוואה לסיבוב הראשון?

מעגל:

* מה למדתי?
* מה הפתיע אותי?
* מה אעשה אחרת?

**הערות נוספות:**

נראה שהם לא מצליחים אף פעם להגיע למשהו עובד. אבל הם חושבים הרבה בדרך, וזה טוב.