

# Εισαγωγή στον οδηγούμενο από συμπεριφορές σχεδιασμό και τις ιστορίες χρηστών

(Τεχνολογία Ανάπτυξης  
Λογισμικού ως Υπηρεσίας  
§7.1)



David Patterson

© 2013 David Patterson & David Patterson  
Licensed under [Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License](#)



1

1

## Γιατί αποτυγχάνουν τα έργα λογισμικού;

- Δεν κάνουν αυτό που θέλουν οι πελάτες
- Ή τα έργα καθυστερούν
- Ή ξεπερνούν τον προϋπολογισμό
- Ή δύσκολη συντήρηση και εξέλιξη
- Ή όλα τα παραπάνω
- Πώς ο Ευέλικτος κύκλος ζωής προσπαθεί να αποφύγει την αποτυχία;

2

2

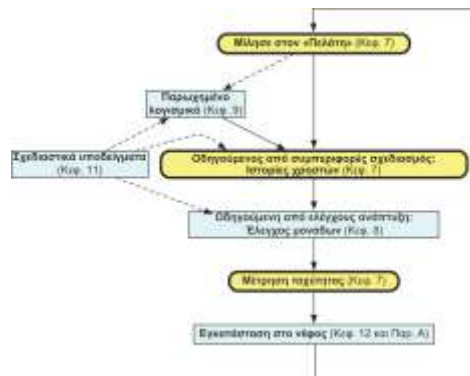
## Ανασκόπηση ευέλικτου κύκλου ζωής

- Στενή και συνεχής συνεργασία με τους ενδιαφερόμενους για την ανάπτυξη απαιτήσεων, ελέγχων
  - Χρήστες, πελάτες, προγραμματιστές, προγραμματιστές συντήρησης, χειριστές, διευθυντές έργων, ...
- Συντήρηση λειτουργικού πρωτοτύπου κατά την εγκατάσταση νέων χαρακτηριστικών σε κάθε επανάληψη
  - Συνήθως κάθε 1 ή 2 εβδομάδες
  - Αντί για 5 μεγάλες φάσεις, διάρκειας μηνών η καθεμία
- Έλεγχος με τους ενδιαφερόμενους για το τι ακολουθεί, για να επικυρώνεται η δημιουργία του σωστού προϊόντος (αντί να επαληθεύεται)

3

3

## Ευέλικτη επανάληψη



4

4

## Σχεδιασμός οδηγούμενος από συμπεριφορές (BDD)

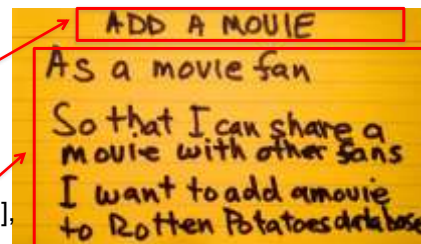
- Η BDD κάνει ερωτήσεις σχετικά με τη συμπεριφορά της εφαρμογής *πριν και κατά τη διάρκεια της ανάπτυξης* για να περιορίζεται η ελλιπής επικοινωνία (miscommunication)
  - Επικύρωση αντί επαλήθευσης
- Οι απαιτήσεις καταγράφονται ως *ιστορίες χρηστών*
  - Λιτές περιγραφές του τρόπου χρήσης της εφαρμογής
- Η BDD επικεντρώνεται στη *συμπεριφορά* της εφαρμογής αντί στην *υλοποίηση* της εφαρμογής
  - Η οδηγούμενη από ελέγχους ανάπτυξη ή TDD (μελλοντικές ενότητες) ελέγχει την υλοποίηση

5

5

## Ιστορίες χρηστών

- 1-3 προτάσεις σε καθομιλουμένη γλώσσα
  - Χωρά σε μια κάρτα 3" x 5"
  - Γράφεται από/με τον πελάτη
- Μορφή «Connextra»:
  - Όνομα χαρακτηριστικού
  - Ως ένας [είδος ενδιαφερόμενου],  
Έτσι ώστε [να πετύχω κάποιο σκοπό],  
Θέλω να [κάνω κάποια εργασία]
  - 3 φράσεις πρέπει να βρίσκονται εκεί, σε οποιαδήποτε σειρά
- Ιδέα: η ιστορία χρήστη μπορεί να τυποποιηθεί ως *έλεγχος αποδοχής πριν* γραφεί ο κώδικας



6

6

## Ιστορίες χρηστών

---

- The Connextra template highlights the who, the what and the why from the user's perspective:
  - As a <someone>
  - I want to <do something>
  - So that <some result or benefit>
- Several people began sharing this template:
  - As a [role]
  - I want [feature]
  - So that [benefit]

New alternative

- As a [role]  
I want [feature]  
So that [business benefit]

7

7

## Ιστορίες χρηστών

---

New alternative

- As a [role]
- I want [feature]
- So that [business benefit]

The easy way out:

- In order to <achieve some value>
- as a <type of user>
- I want <some functionality>

Whether it was the intent or not, I believe that these variations made it easier for teams to think they were working with User Stories when in fact they were simply capturing product-features and business-benefits like old-style software “requirements” – only dressed in a User Story like template.

8

8

## Ιστορίες χρηστών

---

User stories are about the user – hence the name. It's a way of working, not a way of writing requirements. This is an area we all need to get better at explaining.

## Ιστορίες ΧΡΗΣΤΩΝ

9

9

## Γιατί κάρτες 3x5;

---

- (από την κοινότητα Διεπαφών Χρήστη)
- Μη απειλητικές => συμμετέχουν όλοι οι ενδιαφερόμενοι στην ανταλλαγή σκέψεων
- Εύκολες στην επανατοποθέτηση => όλοι οι ενδιαφερόμενοι συμμετέχουν στον καθορισμό προτεραιοτήτων
- Επειδή οι ιστορίες είναι σύντομες, μπορούν να αλλάξουν εύκολα κατά την ανάπτυξη
  - Καθώς συχνά προκύπτουν νέες ιδέες κατά την ανάπτυξη

10

10

## Οι διαφορετικοί ενδιαφερόμενοι περιγράφουν τη συμπεριφορά διαφορετικά

---

- *Δείξε ποιοι από τους φίλους μου θα πάνε σε μια εκδήλωση*
  - Ως ένας κινηματογράφος φίλος
  - Έτσι ώστε να απολαύσω την εκδήλωση με τους φίλους μου
  - Θέλω να δω ποιοι από τους φίλους μου στο Facebook παρακολουθούν μια εκδήλωση
- *Δείξε τους φίλους στο Facebook ενός θαμώνα*
  - Ως ένας διευθυντής γραφείου εισιτηρίων
  - Έτσι ώστε να προσελκύσω έναν θαμώνα να αγοράσει ένα εισιτήριο
  - Θέλω να της δείξω ποιοι από τους φίλους της στο Facebook θα πάνε σε μια εκδήλωση

11

11

## Εκκρεμότητες προϊόντος

---

- Τα πραγματικά συστήματα έχουν εκατοντάδες ιστορίες χρηστών
- *Εκκρεμότητα*: Ιστορίες χρηστών που δεν έχουν ολοκληρωθεί
  - (Θα δούμε ξανά την έννοια της εκκρεμότητας –Backlog– με το Pivotal Tracker)
- Θέτουμε προτεραιότητες ώστε τα πιο πολύτιμα στοιχεία να έχουν υψηλότερη προτεραιότητα
- Τα οργανώνουμε έτσι ώστε να ταιριάζουν με τις κυκλοφορίες λογισμικού στον χρόνο

12

12

## Σχετική έννοια: Αιχμή

- Σύντομη διερεύνηση τεχνικής ή προβλήματος
  - Π.χ. σε αλγορίθμους συστάσεων
  - Πειραματισμός, συγγραφή κώδικα, οτιδήποτε χρειάζεται
- Περιορίζουμε τον διαθέσιμο χρόνο
- Μετά την ολοκλήρωση, *πετάμε τον κώδικα*
  - Τώρα που ξέρουμε την προσέγγιση που θέλουμε, γράφουμε τον κώδικα σωστά!

13

13

Ποια πρόταση σχετικά με την BDD και τις ιστορίες χρηστών είναι ΛΑΘΟΣ;

1. Η BDD είναι σχεδιασμένη να βοηθά με την επικύρωση (δημιουργία του σωστού πράγματος) επιπρόσθετα της επαλήθευσης
2. Οι ιστορίες χρηστών θα πρέπει να περιλαμβάνουν πληροφορίες για τις επιλογές υλοποίησης
3. Οι ιστορίες χρηστών στη BDD παίζουν τον ίδιο ρόλο με τις σχεδιαστικές απαιτήσεις στο μοντέλο σχεδιασμού και τεκμηρίωσης
4. Η ίδια λειτουργικότητα της εφαρμογής θα μπορούσε να συμπεριληφθεί σε πολλές ιστορίες χρηστών σύμφωνα με τις απόψεις πολλών ενδιαφερομένων

14

14

# ΤΕΛΟΣ

15

15



Σημεία, ταχύτητα, και  
το Pivotal Tracker  
(Τεχνολογία Ανάπτυξης Λογισμικού ως  
Υπηρεσίας §7.2)  
David Patterson

© 2013 David Patterson & David Patterson  
Licensed under [Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License](#)



16

16




## Παραγωγικότητα και εργαλεία

- Δεν θέλουμε να αποφύγουμε τη μεγάλη προσπάθεια σχεδιασμού στο Ευέλικτο μοντέλο; Τότε, πώς μπορούμε να εκτιμήσουμε τον χρόνο χωρίς κάποιο σχέδιο;
- Μπορούν οι ιστορίες χρηστών να χρησιμοποιηθούν για τη μέτρηση της προόδου σε ένα έργο;
- Τι θα πρέπει να κάνει ένα εργαλείο για να βοηθά την απόδοση στο Ευέλικτο μοντέλο;

17

17

## Μέτρηση της παραγωγικότητας

- Μια μέτρηση της παραγωγικότητας ομάδας: υπολογισμός του μέσου πλήθους ιστοριών/εβδομάδα;
 
  - Όμως μερικές ιστορίες είναι πιο δύσκολες από άλλες
- Βαθμολόγηση κάθε ιστορίας χρήστη εκ των προτέρων σε μια απλή κλίμακα ακεραίων
  - 1 για απλές ιστορίες, 2 για μέτριες ιστορίες, 3 για πολύ σύνθετες ιστορίες
- **Ταχύτητα:** μέσο πλήθος σημείων/εβδομάδα

18

18

## Περισσότερα για τα σημεία

- Αφού αποκτηθεί κάποια πείρα, χρησιμοποιείται συχνά η κλίμακα Fibonacci: 1, 2, 3, 5, 8
  - (Κάθε νέος αριθμός είναι το άθροισμα των 2 προηγούμενων)
  - Στην Pivotal Labs, το 8 είναι εξαιρετικά σπάνιο
  - Συμβουλή: όταν ξεκινάτε, αν  $\geq 5$  τότε **διασπάστε την ιστορία!**
- Ψήφος ομάδων: υψώστε δάχτυλα, πάρτε τον μέσο όρο
  - Αν υπάρχει μεγάλη ασυμφωνία (2 και 5), συζητήστε περισσότερο

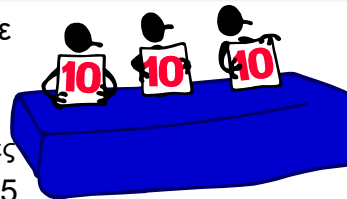


19

19

## Περισσότερα για τα σημεία

- $\geq 5 \Rightarrow$  διαιρέστε την ιστορία χρήστη σε απλούστερες ιστορίες, ομαδοποιήστε σε *έπη* (epics)
  - οι εκκρεμότητες δεν είναι πολύ απαιτητικές
- Δεν έχει σημασία αν η ταχύτητα είναι 5 ή 10 σημεία ανά επανάληψη
  - Όσο η ομάδα είναι συνεπής
- Η ιδέα είναι να βελτιώσετε την αυτοαξιολόγηση και να προτείνετε το πλήθος των επαναλήψεων για ένα σύνολο χαρακτηριστικών



20

20

## Επίδειξη: Pivotal Tracker

- Υπολογίζει την ταχύτητα της ομάδας, διαχειρίζεται τις ιστορίες χρηστών: Current, Backlog, Icebox



21

21

## Pivotal Tracker

- Καθορίζει προτεραιότητες στις ιστορίες χρηστών τοποθετώντας τις στα τμήματα Current, Backlog, Icebox
- Όταν ολοκληρώνονται, τις μετακινεί στο τμήμα Done
  - Οι προγραμματιστές πατούν στο κουμπί Finish, στέλνουν την ιστορία στον Ιδιοκτήτη Προϊόντος
  - Ο ιδιοκτήτης δοκιμάζει την ιστορία και την αποδέχεται ή απορρίπτει
- Μπορεί να προσθέσει λογικά σημεία κυκλοφορίας ώστε να προσδιορίσει πότε θα συμβεί μια κυκλοφορία
  - Υπόλοιπα σημεία/Ταχύτητα
- Έπος (epic)** (με δικό του τμήμα)
  - Συνδυάζει σχετικές ιστορίες χρηστών
  - Ανεξάρτητη διάταξη από τις ιστορίες χρηστών



22

22

## Pivotal Tracker: Χαρακτηριστικά και μικροδουλειές

---

- **Χαρακτηριστικά (Features)**
  - Ιστορίες χρηστών που παρέχουν επαληθεύσιμη επιχειρηματική αξία στον πελάτη
    - «Προσθήκη πλαισίου 'Συμφωνώ' στη σελίδα πληρωμής»
  - Αξίζει σημεία & πρέπει να εκτιμηθεί
- **Μικροδουλειές (Chores) & Σφάλματα (Bugs)**
  - Ιστορίες χρηστών που είναι αναγκαίες, αλλά δεν παρέχουν άμεση, προφανή αξία στον πελάτη
    - «Να δούμε γιατί είναι αργή η συλλογή ελέγχων»
    - «Αναδόμηση του υποσυστήματος πληρωμών»
  - Καθόλου σημεία

23

23

## Πίνακας κυβερνοχώρου ομάδας

---

- Το Tracker επιτρέπει την επισύναψη εγγράφων σε ιστορίες χρηστών (π.χ., διεπαφή χρήστη χαμηλής πιστότητας – LoFi UI)
- Wiki με αποθετήριο GitHub
- Google Documents: ομαδική δημιουργία και εξέταση σχεδίων, παρουσιάσεων, υπολογιστικών φύλλων, και εγγράφων
- Campfire: υπηρεσία που βασίζεται στον Ιστό για διαδικτυακές αίθουσες συζητήσεων προστατευμένες με κωδικό πρόσβασης

24

24

Ποια πρόταση σχετικά με τα σημεία, την ταχύτητα και το Tracker είναι ΑΛΗΘΗΣ;

1. Κατά τη σύγκριση δύο ομάδων, αυτή με την υψηλότερη ταχύτητα είναι η πιο παραγωγική
2. Όταν δεν γνωρίζετε πώς να προσεγγίσετε μια δεδομένη ιστορία χρήστη, απλώς βαθμολογήστε την με 3 σημεία
3. Με το Tracker, οι προγραμματιστές επιλέγουν τις ιστορίες χρηστών και τις επισημαίνουν ως «Αποδεκτές» όταν τελειώνουν
4. Το Tracker βοηθά στον ορισμό προτεραιοτήτων και την παρακολούθηση των ιστοριών χρηστών και της κατάστασής τους, υπολογίζει την ταχύτητα, και προβλέπει τον χρόνο ανάπτυξης λογισμικού

25

25



## 7. Ιστορίες και επίπεδα

- «Η υποδιαίρεση της δουλειάς σε ιστορίες βοηθά όλα τα μέλη της ομάδας να κατανοήσουν την εφαρμογή και να έχουν περισσότερη αυτοπεποίθηση όταν κάνουν αλλαγές σε αυτή»
- «Το Tracker μάς βοήθησε να ορίσουμε προτεραιότητες σε χαρακτηριστικά και να εκτιμήσουμε τη δυσκολία»
- «Διαιρέσαμε την εφαρμογή σε επίπεδα [διεπαφής, υποδομής, JavaScript, κ.λπ] και ήταν δύσκολο να συντονιστούμε για να κάνουμε τα χαρακτηριστικά να λειτουργήσουν»
- «Ήταν δύσκολο να εκτιμήσουμε αν η δουλειά χωρίστηκε δίκαια...δεν είμαστε σίγουροι αν η ικανότητά μας να εκτιμούμε τη δυσκολία βελτιώθηκε με τον χρόνο»

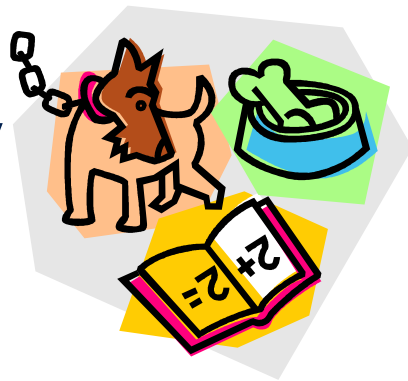
26

# ΤΕΛΟΣ

27

27

«Έξυπνες»  
ιστορίες χρηστών  
(Τεχνολογία ανάπτυξης  
Λογισμικού ως  
Υπηρεσίας §7.3)



David Patterson

© 2013 David Patterson & David Patterson  
Licensed under [Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License](#)



28

28

## Δημιουργία ιστοριών χρηστών

---

- Πώς ξέρετε αν έχετε μια καλή ή κακή ιστορία χρήστη;
  - Σωστό μέγεθος;
  - Όχι πολύ δύσκολη;
  - Αξίζει;

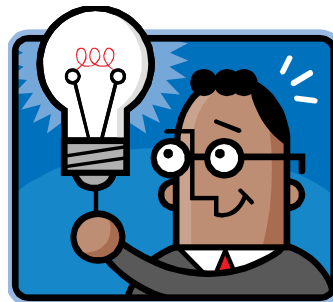
29

29

## «Έξυπνες» (SMART) ιστορίες

---

- Συγκεκριμένη (**S**pecific)
- Μετρήσιμη (**M**easurable)
- Επιτεύξιμη (**A**chievable)  
(ιδανικά, υλοποιείται σε 1 επανάληψη)
- Σημαντική (**R**elevant)  
(«τα 5 γιατί»)
- Περιορισμένου χρόνου  
(**T**imeboxed)  
(ξέρεις πότε να την παρατήσεις)



30

30

## Συγκεκριμένη και μετρήσιμη

- Κάθε σενάριο είναι ελέγξιμο
  - Υπονοεί γνωστή καλή είσοδο και αναμενόμενα αποτελέσματα
- Αντιπαράδειγμα: «διεπαφή φιλική προς τον χρήστη»
- Παράδειγμα: Given/When/Then
  1. Δεδομένων (Given) μερικών συγκεκριμένων αρχικών συνθηκών,
  2. Όταν (When) εκτελέσω τη συγκεκριμένη ενέργεια X,
  3. Τότε (Then) θα συμβούν ένα ή περισσότερα συγκεκριμένα πράγματα



31

31

## Επιτεύξιμη

- Ολοκληρώνεται σε 1 επανάληψη
- Αν δεν μπορεί να προκύψει το χαρακτηριστικό σε 1 επανάληψη, προκύπτει ένα υποσύνολο ιστοριών
  - Ο στόχος είναι πάντα να έχουμε λειτουργικό κώδικα στο τέλος της επανάληψης
- Αν  $<1$  ιστορία ανά επανάληψη, χρειάζεται βελτίωση η εκτίμηση σημείων ανά ιστορία



32

32



## Σημαντική: «Επιχειρηματική αξία»

---

- Ανακάλυψη επιχειρηματικής αξίας, ή απόρριψη της ιστορίας:
  - Προστασία των εσόδων
  - Αύξηση των εσόδων
  - Διαχείριση του κόστους
  - Αύξηση της αξίας του εμπορικού ονόματος
  - Το προϊόν γίνεται θαυμάσιο
- Μπορείς να συμπεριλάβεις ιστορίες που δεν έχουν προφανή επιχειρηματική αξία;

33

33

## 5 λόγοι για να βρούμε τη σημασία

---

- *Δείξε τους φίλους του θαμώνα στο Facebook*  
 Ως διευθυντής γραφείου εισιτηρίων  
 Έτσι ώστε να μπορώ να προσελκύσω έναν θαμώνα να αγοράσει ένα εισιτήριο  
 Θέλω να της δείξω ποιοι φίλοι της στο Facebook πηγαίνουν σε μια εκδήλωση
  1. Γιατί;
  2. Γιατί;
  3. Γιατί;
  4. Γιατί;
  5. Γιατί;



34

34

## 5 λόγοι για να βρούμε τη σημασία

---

- *An example of a problem is: The vehicle will not start.*
- *Why? – The battery is dead. (First why)*
- *Why? – The alternator is not functioning. (Second why)*
- *Why? – The alternator belt has broken. (Third why)*
- *Why? – The alternator belt was well beyond its useful service life and not replaced. (Fourth why)*
- *Why? – The vehicle was not maintained according to the recommended service schedule. (Fifth why, a root cause)*

35

35

## Χρονικά περιορισμένη

---

- Σταμάτα την ιστορία όταν ξεπεράσεις τον χρονικό προϋπολογισμό
  - Εγκατάλειψε ή διαίρεσε σε μικρότερες ιστορίες ή αναπρογραμμάτισε ό,τι έμεινε στη μέση
- Για να αποφύγεις την υποτίμηση της χρονικής διάρκειας του έργου
- Το Pivotal Tracker παρακολουθεί την ταχύτητα, το οποίο βοηθάει στην αποφυγή της κακής εκτίμησης



36

36

Ποιο από τα παρακάτω χαρακτηριστικά είναι το ΛΙΓΟΤΕΡΟ «ΕΞΥΓΙΝΟ» (SMART);

1. Ο χρήστης μπορεί να αναζητήσει μια ταινία με βάση τον τίτλο
2. Εφόσον έχω ένα δωρεάν εισιτήριο ταινίας, θέλω να το εξαργυρώσω για μια διαθέσιμη ταινία πριν τη λήξη του
3. Όταν προσθέτω μια ταινία, το 99% των σελίδων προσθήκης ταινίας (Add Movie) πρέπει να εμφανίζονται σε 3 δευτερόλεπτα
4. Ως πελάτης, θέλω να βλέπω τις 10 κορυφαίες ταινίες σε πωλήσεις, σε σειρά τιμής, έτσι ώστε να αγοράζω πρώτα τις φθηνότερες

37

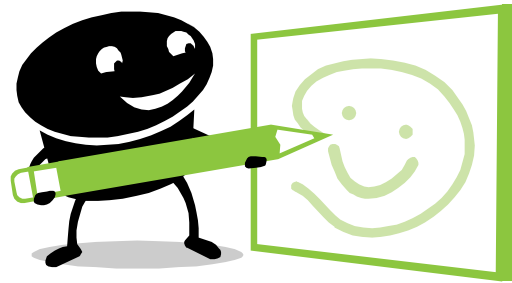
37



38

38

# Σκίτσα διεπαφής χρήστη χαμηλής πιστότητας και εικονοσενάρια (Τεχνολογία Ανάπτυξης Λογισμικού ως Υπηρεσίας §7.4)



David Patterson

© 2012 David Patterson & Armando Fox  
Licensed under [Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License](#)



39

39

## Κατασκευή επιτυχημένων διεπαφών χρήστη

- Οι εφαρμογές SaaS συχνά αλληλοεπιδρούν με τους χρήστες  
⇒ Οι ιστορίες χρηστών χρειάζονται διεπαφή χρήστη (User Interface, UI)
- Πώς κάνω τον πελάτη να συμμετάσχει στη σχεδίαση της διεπαφής ώστε να είναι ευχαριστημένος όταν ολοκληρωθεί;
  - Αποφυγή UI WISBNWIW\*;
  - Έκδοση UI σε κάρτες 3x5;
- Πώς δείχνω την αλληλεπίδραση χωρίς να κατασκευάσω πρωτότυπο;

\* What-I-Said-But-Not-What-I-Want  
(αυτό που είπα αλλά όχι αυτό που θέλω)

40

40

## Σχεδίαση διεπαφής χρήστη SaaS

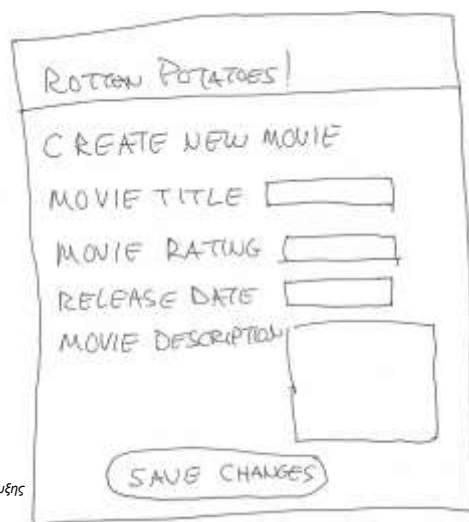
- **Σκίτσα UI:** σχέδια με μολύβι και χαρτί ή «Lo-Fi UI» (διεπαφή χρήστη χαμηλής πιστότητας)



41

41

## Παράδειγμα διεπαφής χαμηλής πιστότητας



(Από το βιβλίο Τεχνολογία Ανάπτυξης  
Λογισμικού ως Υπηρεσίας © των  
Armando Fox και David Patterson,  
χρησιμοποιείται με άδεια)

42

42

## Εικονοσενάρια

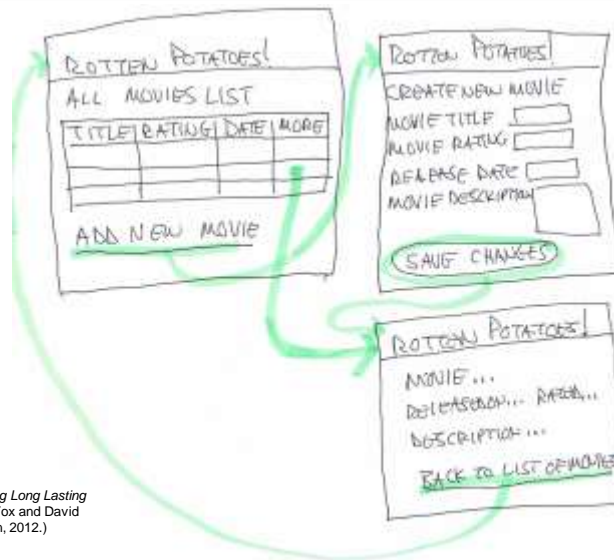
- Χρειάζεται να δείξετε πώς οι αλλαγές στο UI βασίζονται στις αλληλεπιδράσεις με τον χρήστη
- HCI => «εικονοσενάρια» (storyboards)
- Σαν τις σκηνές ταινιών, αλλά μη γραμμικές



43

43

## Παράδειγμα εικονοσεναρίου



(Εικόνα 4.4, *Engineering Long Lasting Software* by Armando Fox and David Patterson, Alpha edition, 2012.)

44

44

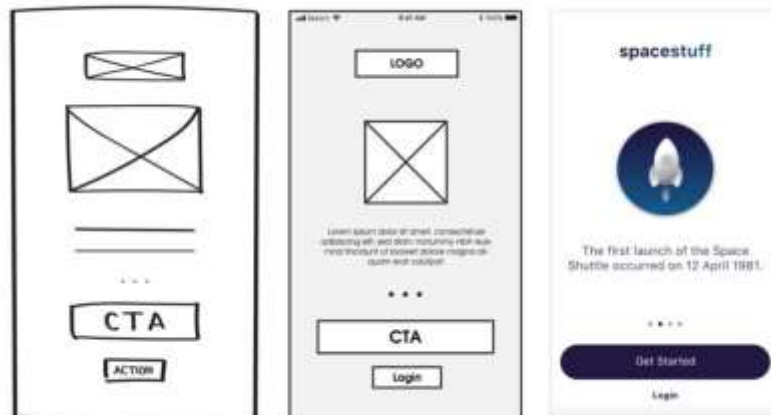
## Lo-Fi σε HTML

- Η δημιουργία σκίτσων και εικονοσεναρίων είναι κουραστική, αλλά ευκολότερη από την δημιουργία HTML! **Και...**
  - Λιγότερο τρομακτική για ενδιαφερόμενους χωρίς τεχνικές γνώσεις
  - Πιο πιθανό να προταθούν αλλαγές στη διεπαφή χρήστη αν δεν υπάρχει κώδικας από πίσω
  - Πιο πιθανό να εστιάσει στην *αλληλεπίδραση* αντί στα χρώματα, γραμματοσειρές, ...
- Τα CSS (Φύλλα Επάλληλων Στυλ) μπορούν να κάνουν τη διεπαφή κομψότερη αργότερα

45

45

## Low – Medium – High Fidelity



46

# Prototypes Tools

---

- Marvel (κινητα)
- Justinmind
- Proto.io
- Balsamiq

---

47

Ποιο από τα παρακάτω είναι ΛΑΘΟΣ για τις διεπαφές χρήστη χαμηλής πιστότητας;

- ☐ Όπως οι κάρτες 3x5, τα σκίτσα και τα εικονοσενάρια (σε σύγκριση με τον κώδικα) είναι πιο πιθανό να εμπλέκουν τους ενδιαφερόμενους
- ☐ Ο σκοπός της προσέγγισης διεπαφών χαμηλής πιστότητας είναι η αποσφαλμάτωση της διεπαφής πριν τον προγραμματισμό της
- ☐ Οι εφαρμογές SaaS συνήθως έχουν μια διεπαφή χρήστη που συσχετίζεται με τις ιστορίες χρηστών
- ☐ Ενώ τα εικονοσενάρια χαμηλής πιστότητας απαιτούν περισσότερο χρόνο από τη δημιουργία ενός πρωτοτύπου της διεπαφής χρήστη με CSS/HTML, η προσέγγιση χαμηλής πιστότητας είναι πιο πιθανό να οδηγήσει σε μια διεπαφή που αρέσει στους πελάτες

48

---

48





## #8. Συνεργασία με τον πελάτη: BDD & κατασκευή πρωτοτύπου χαμηλής πιστότητας

---

- «Η χαμηλή πιστότητα και τα εικονοσενάρια βοηθούν πολύ στη συνεργασία με τον πελάτη»
  - «Η συχνή ανατροφοδότηση από τον πελάτη είναι αναγκαία»
  - «Αυτό που πιστέψαμε ότι είναι ωραίο δεν ήταν αυτό που ενδιέφερε τον πελάτη»
  - «Δημιουργήσαμε πρωτότυπα υψηλής πιστότητας, και επενδύσαμε πολύ χρόνο μόνο για να συνειδητοποιήσουμε ότι δεν άρεσαν στον πελάτη»
  - «Ποτέ δεν καταλάβαμε πόσο μεγάλη πρόκληση ήταν η μετάβαση από την περιγραφή του πελάτη σε ένα τεχνικό σχέδιο»
- 

49

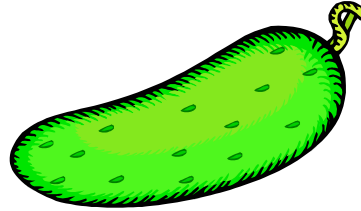
# ΤΕΛΟΣ

50

50

## Εισαγωγή στα Cucumber & Capybara

(Τεχνολογία Ανάπτυξης  
Λογισμικού ως  
Υπηρεσίας §7.5)



David Patterson

© 2013 Armando Fox & David Patterson, all rights reserved

51

51

## Ιστορίες χρηστών => Έλεγχοι αποδοχής;

- Δεν θα ήταν ωραία να αντιστοιχίζονται αυτόματα οι ιστορίες χρηστών από κάρτες 3x5 σε ελέγχους για να αποφασίζει ο χρήστης αν θα αποδεχθεί την εφαρμογή;
- Πώς θα αντιστοιχίζατε την περιγραφή σε κώδικα ελέγχου;
- Πώς θα εκτελούσατε τους ελέγχους χωρίς να εμπλέκεται κάποιο άτομο στη διαδικασία για την εκτέλεση των ενεργειών;

52

52

## Cucumber: Μεγάλη Ιδέα

- Έλεγχοι από ιστορίες χρηστών φιλικές προς τον πελάτη
  - Αποδοχή: διασφαλίζεται η ικανοποίηση του πελάτη
  - Ολοκλήρωση: διασφαλίζεται ότι οι διεπαφές μεταξύ των υπομονάδων έχουν συνεπείς παραδοχές, επικοινωνούν σωστά
- Το Cucumber παρεμβάλλεται ανάμεσα στον πελάτη και τον προγραμματιστή
  - Οι ιστορίες χρηστών δεν είναι κώδικας, άρα είναι ξεκάθαρες στον πελάτη και μπορούν να χρησιμοποιηθούν για να επιτευχθεί συμφωνία
  - Επίσης δεν έχουν πλήρως ελεύθερη μορφή, έτσι μπορούν να συνδεθούν με πραγματικούς ελέγχους

53

53

## Παράδειγμα ιστορίας χρήστη

### 1 Χαρακτηριστικό

- Χαρακτηριστικό: Ο χρήστης μπορεί να προσθέσει χειροκίνητα μια ταινία

### • Σενάριο: Προσθήκη ταινίας $\geq 1$ Σενάριο / Χαρακτηριστικό

- Εφόσον βρίσκομαι στην αρχική σελίδα της RottenPotatoes
- Όταν ακολουθώ τον σύνδεσμο προσθήκης νέας ταινίας (Add new movie)
- Τότε θα πρέπει να βρίσκομαι στη σελίδα δημιουργίας ταινίας (Create New Movie)
- Όταν συμπληρώνω τον τίτλο (Title) με τη φράση "Men In Black"
- Και επιλέγω καταλληλότητα "PG-13" από το μενού "Rating"
- Και πατάω στο κουμπί αποθήκευσης αλλαγών (Save Changes)
- Τότε θα πρέπει να βρίσκομαι στην αρχική σελίδα της RottenPotatoes
- Και θα πρέπει να δω την ταινία "Men In Black"

### 3 με 8 Βήματα / Σενάριο

54

54

## Ιστορία χρήστη, χαρακτηριστικό, και βήματα στο Cucumber

- **Ιστορία χρήστη:** συνήθως αντιστοιχεί σε ένα χαρακτηριστικό
- **Χαρακτηριστικό:**  $\geq 1$  **σενάρια** που δείχνουν διαφορετικούς τρόπους με τους οποίους χρησιμοποιείται ένα χαρακτηριστικό
  - Οι λέξεις-κλειδιά **Χαρακτηριστικό** και **Σενάριο** προσδιορίζουν τα αντίστοιχα μέρη
  - σενάρια **χαρούμενου & λυπηρού μονοπατιού**  
`features/*.feature`
- **Σενάριο:** συνήθως 3 - 8 **βήματα**
- **Ορισμοί βημάτων:** Κώδικας Ruby για έλεγχο **βημάτων**  
`features/step_definitions/*_steps.rb`



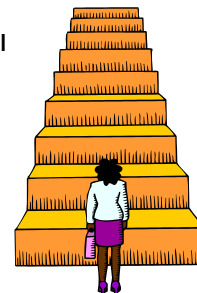
55

55

## 5 Λέξεις-κλειδιά βημάτων

1. Τα βήματα **Given** (**Εφόσον**) αναπαριστούν την κατάσταση πριν το συμβάν: προσυνθήκες
2. Τα βήματα **When** (**Όταν**) αναπαριστούν το συμβάν
  - π.χ., εξομοιώνουν το πάτημα ενός κουμπιού από τον χρήστη
3. Τα βήματα **Then** (**Τότε**) αναπαριστούν τις αναμενόμενες μετασυνθήκες· ελέγχουν αν είναι επαληθεύονται
4. / 5. Τα βήματα **And** (**Και**) και **But** (**Αλλά**) επεκτείνουν το προηγούμενο βήμα

*Αυτά όλα είναι ψευδώνυμα της ίδιας μεθόδου*



56

56

## Βήματα => Ορισμοί βημάτων με κανονικές παραστάσεις

- **Οι κανονικές παραστάσεις ταυτίζουν τις (αγγλικές) φράσεις σε βήματα σεναρίων με ορισμούς βημάτων!**

- `Given /^(:|I )am on (.+)$/`
- “I am on the Rotten Potatoes home page”
- Οι ορισμοί βημάτων (κώδικας Ruby) πιθανόν χρησιμοποιούν την ταυτισμένη συμβολοσειρά
  - “the Rotten Potatoes home page”



57

57

## Προσομοίωση χρήστη για τη δοκιμή σεναρίων;

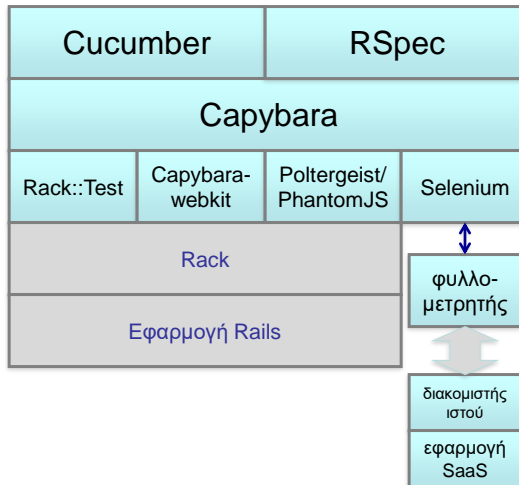
- Εργαλείο το οποίο προσποιείται ότι είναι ο χρήστης για να ακολουθήσει τα σενάρια της ιστορίας χρήστη
- Το Carylbara προσομοιώνει τον φυλλομετρητή
  - Μπορεί να αλληλεπιδράσει με την εφαρμογή για να λάβει σελίδες
  - Αναλύει την HTML
  - Υποβάλλει φόρμες σαν χρήστης



58

58

## Στοιίβες ελέγχου του Cucumber

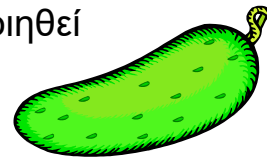


- Με το Selenium, μπορούν να υλοποιηθούν με σενάρια (script) οι εξωτερικές αλληλεπιδράσεις
- Η SauceLabs.com θα εκτελέσει τους ελέγχους Selenium και θα σας στείλει βίντεο με τα αποτελέσματα

59

## Από Κόκκινο σε Πράσινο

- **cucumber** *όνομα\_αρχείου* για την εκτέλεση ενός χαρακτηριστικού, **rake cucumber** για την εκτέλεση όλων
- **Πράσινο** για επιτυχή βήματα
- **Κίτρινο** όταν δεν έχει ακόμη υλοποιηθεί
- **Κόκκινο** για αποτυχία (τα ακόλουθα βήματα είναι **Μπλε**)
- Στόχος: Όλα τα βήματα να γίνουν **πράσινα** που σημαίνει ότι εκτελούνται με επιτυχία (Από εκεί προέρχεται και το όνομα του εργαλείου ως πράσινου λαχανικού)



60

60

Ποια πρόταση, αν υπάρχει, είναι **ΛΑΘΟΣ** σχετικά με το «Cuke + Capy»;

1. Τα χαρακτηριστικά θα πρέπει να περιλαμβάνουν σενάρια για το χαρούμενο και το λυπηρό μονοπάτι
2. Τα «Cuke+Capy» είναι κατάλληλα για τον έλεγχο ολοκλήρωσης/πλήρους στοίβας αλλά όχι για τον έλεγχο μονάδων/υπομονάδων
3. Μερικά σενάρια Cuke που εκτελούνται με Rack::Test ίσως να μην λειτουργούν με το Selenium
4. Όλα τα παραπάνω είναι αληθή· κανένα δεν είναι λάθος

61

61

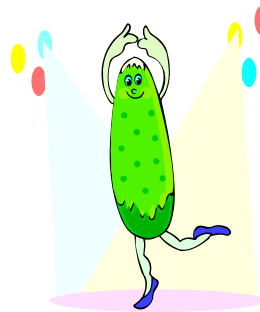


**ΤΕΛΟΣ**

62

62

# Εκτέλεση των Cucumber και Carybara (Τεχνολογία Ανάπτυξης Λογισμικού ως Υπηρεσίας §7.6)



David Patterson

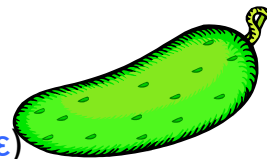
© 2013 Armando Fox & David Patterson, all rights reserved

66

66

## Ανάλυση Κόκκινου-Κίτρινου- Πράσινου

- Στο Cucumber τα βήματα χρωματίζονται
- **Πράσινο** για επιτυχή εκτέλεση
- **Κίτρινο** όταν δεν έχει υλοποιηθεί
- **Κόκκινο** για αποτυχία  
(τότε τα ακόλουθα βήματα είναι **Μπλε**)
- Στόχος: Όλα τα βήματα να γίνουν **πράσινα**  
που σημαίνει ότι εκτελούνται με επιτυχία  
(Από εκεί προέρχεται και το όνομα του  
εργαλείου ως πράσινου λαχανικού)



67

67



## Επίδειξη

---

- Προσθήκη χαρακτηριστικού για την κάλυψη υπάρχουσας λειτουργικότητας
  - Σημείωση: Σε αυτό το παράδειγμα γίνεται σε λάθος σειρά – θα πρέπει να γραφούν οι έλεγχοι πρώτα
  - Γίνεται μόνο για παιδαγωγικούς σκοπούς
- (Ή μπορείτε να δείτε την εκπομπή:  
<http://vimeo.com/34754747>)

68

68



69

69

# Βελτίωση της Rotten Potatoes ξανά (Τεχνολογία Ανάπτυξης Λογισμικού ως Υπηρεσίας §7.8)



David Patterson

© 2013 Armando Fox & David Patterson, all rights reserved

70

70

## Προσθήκη ενός πραγματικού νέου χαρακτηριστικού;

- Τι θα γινόταν αν προσθέταμε κάτι δυσκολότερο;
  - π.χ., περιλαμβάνει φόρμα για συμπλήρωση
  - π.χ., χρειάζεται μια διεπαφή χρήστη
  - π.χ., χρειάζεται να προστεθεί διαδρομή για τη σύνδεση προβολής σε ελεγκτή
  - π.χ., περιλαμβάνει και χαρούμενο μονοπάτι και λυπηρό μονοπάτι

71

71

## Ενοποίηση με τη The Movie Database (TMDb)

- Νέο χαρακτηριστικό: Εισαγωγή δεδομένων από την TMDb αντί της εισαγωγής πληροφοριών «με το χέρι»
- Πρέπει να προστεθεί η δυνατότητα αναζήτησης στην TMDb από την αρχική σελίδα της Rotten Potatoes
- Χρειάζεται διεπαφή χρήστη χαμηλής πιστότητας και εικονοσενάριο

72

72

### Εικονοσενάριο TMDb

- Εικόνα 7.6 από το βιβλίο *Τεχνολογία Ανάπτυξης Λογισμικού ως Υπηρεσίας*



73

73

## Ιστορία χρήστη για αναζήτηση στην TMDb (Εικόνα 7.7 στο βιβλίο *Τεχνολογία SaaS*)

Χαρακτηριστικό: Ο χρήστης μπορεί να προσθέσει ταινία κάνοντας αναζήτηση στη The Movie Database (TMDb)

Ως θαυμαστής της ταινίας

Ωστε να μπορώ να προσθέτω νέες ταινίες χωρίς να ταλαιπωρούμαι

Θέλω να προσθέτω ταινίες αναζητώντας τις λεπτομέρειές τους στην TMDb

Σενάριο: Απόπειρα προσθήκης μη υπάρχουσας ταινίας (λυπηρό μονοπάτι)

Εφόσον βρίσκομαι στην αρχική σελίδα της RottenPotatoes

Τότε θα πρέπει να βλέπω το κείμενο "Search TMDb for a movie"

Όταν συμπληρώνω το πεδίο "Search Terms" με το κείμενο "Movie That Does Not Exist"

Και πατώ το κουμπί "Search TMDb"

Τότε θα πρέπει να βρίσκομαι στην αρχική σελίδα RottenPotatoes

Και θα πρέπει να βλέπω το κείμενο "'Movie That Does Not Exist' was not found in TMDb."

74

74

## Haml για αναζήτηση στη σελίδα TMDb (Εικόνα 7.8 στο βιβλίο *Τεχνολογία SaaS*)

```
-# προσθήκη στο τέλος του
```

```
app/views/movies/index.html.haml:
```

```
%h1 Search TMDb for a movie
```

```
= form_tag :action => 'search_tmdb' do
```

```
  %label{:for => 'search_terms'} Search Terms
```

```
  = text_field_tag 'search_terms'
```

```
  = submit_tag 'Search TMDb'
```

<http://pastebin/18yYBVbC>

75

75

## Επέκταση της Haml στις τελευταίες 2 γραμμές

---

- Αυτή η Haml:
 

```
= text_field_tag 'search_terms'
= submit_tag 'Search TMDb'
```
- Μετατρέπεται σε αυτήν την HTML:
 

```
<label for='search_terms'>Search
  Terms</label>
<input id="search_terms" name="search_terms"
  type="text" />
```
- Η ιδιότητα `for` της ετικέτας `label` ταυτίζεται με την ιδιότητα `id` της ετικέτας `input`, από τη βοηθητική `text_field_tag` (φαίνεται παραπάνω)

76

76

## Δοκιμάσατε το Cucumber;

---

- Αν δοκιμάσετε το Cucumber, αποτυγχάνει
- Λείπει η διαδρομή
- Επίσης η `MoviesController#search_tmdb` είναι ενέργεια ελεγκτή που θα λάβει τη φόρμα, αλλά δεν υπάρχει ακόμα στο `movies_controller.rb`
- Θα πρέπει να χρησιμοποιήσουμε ανάπτυξη οδηγούμενη από ελέγχους (μελλοντική διάλεξη) για να υλοποιήσουμε τη μέθοδο `search_tmdb`
- Παρόλα αυτά, για να ολοκληρώσουμε το λυπηρό μονοπάτι, προσθέτουμε μια ψευδομέθοδο ελεγκτή που πάντα αποτυγχάνει

77

77

## Πυροδότηση ψευδοελεγκτή όταν δημοσιεύεται με POST η φόρμα (Εικόνα 7.9)

---

```
# προσθήκη στο routes.rb, μόλις πριν ή μετά τη
  'resources :movies' :

# διαδρομή που δημοσιεύει τη φόρμα 'Search
  TMDb'
post '/movies/search_tmdb'
```

<http://pastebin/FrFkF6pd>

78

78

## Ψευδομέθοδος ελεγκτή: Θα αποτύχει στην εύρεση ταινίας (Εικόνα 7.9)

---

```
# προσθήκη στο movies_controller.rb, οπουδήποτε
# μέσα στο 'class MoviesController <
  ApplicationController':

def search_tmdb
  # προρυθμισμένο για προσομοίωση αποτυχίας
  flash[:warning] = "'#{params[:search_terms]}'
  was not found in TMDb."
  redirect_to movies_path
end
```

<http://pastebin/smwvxv70i>

79

79



80

80

Ποια πρόταση είναι ΣΩΣΤΗ;

1. Συνήθως πρώτα ολοκληρώνεται η φάση του οδηγούμενου από συμπεριφορές σχεδιασμού με το Cucumber πριν ξεκινήσει η φάση της οδηγούμενης από ελέγχους ανάπτυξης με το Rspec
2. Συνήθως πρώτα γράφεται ο κώδικας του λυπηρού μονοπατιού
3. Ένα λυπηρό μονοπάτι μπορεί να επιτύχει χωρίς να έχει γραφεί ο κώδικας που απαιτείται για την επιτυχή εκτέλεση ενός χαρούμενου μονοπατιού
4. Κανένα από τα παραπάνω δεν είναι σωστό

81

81

# ΤΕΛΟΣ

82

82

Ρητά και υπονοούμενα,  
και προστατικά και  
δηλωτικά  
σενάρια  
(Τεχνολογία Ανάπτυξης  
Λογισμικού ως Υπηρεσίας \$7.9)



David Patterson

© 2013 Armando Fox & David Patterson, all rights reserved

89

89



## Τύποι σεναρίων

---

- Προκύπτουν όλες οι απαιτήσεις απευθείας από τις ιστορίες χρηστών;
- Τα σενάρια θα πρέπει να έχουν 3 μέχρι 8 βήματα· υπάρχει τρόπος να τα διατηρήσουμε κοντά στο 3 αντί στο 8;

90

90

## Ρητά και υπονοούμενα σενάρια

---

- Οι ρητές απαιτήσεις συνήθως αποτελούν μέρος των ελέγχων αποδοχής
  - Πιθανές ρητές ιστορίες χρηστών και σενάρια: λίστα ταινιών
- Οι έμμεσες απαιτήσεις είναι λογική συνέπεια των ρητών απαιτήσεων, συνήθως του ελέγχου ολοκλήρωσης
  - Οι ταινίες παρουσιάζονται σε χρονολογική ή αλφαβητική σειρά;

91

91

## Προστακτικά και δηλωτικά σενάρια

- Προστακτικά: Αρχικές ιστορίες χρηστών με πολλά βήματα, που καθορίζουν τη λογική ακολουθία προς το επιθυμητό αποτέλεσμα
  - Δεν είναι DRY αν είναι προστακτικές πολλές ιστορίες χρηστών
- Δηλωτικά: περιγράφουν κατάσταση, όχι ακολουθία
  - Λιγότερα βήματα
- Παράδειγμα χαρακτηριστικού: οι ταινίες πρέπει να εμφανίζονται σε αλφαβητική σειρά, όχι σε σειρά εισαγωγής
- Παράδειγμα σεναρίου: προβολή λίστας ταινιών μετά την προσθήκη 2 ταινιών

92

92

## Παράδειγμα προστακτικού σεναρίου

Εφόσον βρίσκομαι στην αρχική σελίδα της RottenPotatoes  
 Όταν ακολουθώ τον σύνδεσμο προσθήκης νέας ταινίας (Add new movie)  
 Τότε θα πρέπει να βρίσκομαι στη σελίδα δημιουργίας ταινίας (Create New Movie)  
 Όταν συμπληρώνω το πεδίο τίτλου (Title) με τη φράση "Zorro"  
 Και επιλέγω "PG" από το μενού καταλληλότητας (Rating)  
 Και πατώ στο κουμπί "Save Changes"  
 Τότε θα πρέπει να βρίσκομαι στην αρχική σελίδα της RottenPotatoes  
 Όταν ακολουθώ τον σύνδεσμο προσθήκης νέας ταινίας (Add new movie)

Τότε θα πρέπει να βρίσκομαι στη σελίδα δημιουργίας ταινίας (Create New Movie)  
 Όταν συμπληρώνω το πεδίο τίτλου (Title) με το κείμενο "Apocalypse Now"  
 Και επιλέγω "R" από το μενού καταλληλότητας (Rating)  
 Και πατώ στο κουμπί "Save Changes"  
 Τότε θα πρέπει να βρίσκομαι στη σελίδα RottenPotatoes  
 Και θα πρέπει να βλέπω τον τίτλο "Apocalypse Now" πριν από τον "Zorro"

Το μόνο βήμα που καθορίζει συμπεριφορά· τα υπόλοιπα είναι υλοποίηση. Αλλά η BDD καθορίζει συμπεριφορά, όχι υλοποίηση!

93

93

## Γλώσσα πεδίου

- Δηλωτική σαν γλώσσα πεδίου
- Χρησιμοποιεί όρους και την έννοια της εφαρμογής
- Άτυπη γλώσσα
- Τα δηλωτικά βήματα περιγράφουν την κατάσταση στην οποία πρέπει να βρίσκεται η εφαρμογή
  - Προστακτική: ακολουθία βημάτων που αλλάζουν την τρέχουσα κατάσταση στην επιθυμητή κατάσταση

94

94

## Παράδειγμα δηλωτικού σεναρίου

Χαρακτηριστικό: οι ταινίες όταν προστίθενται πρέπει να εμφανίζονται στην λίστα ταινιών

Σενάριο: προβολή λίστας ταινιών μετά την προσθήκη ταινίας (δηλωτική και DRY)

Εφόσον έχω προσθέσει την ταινία «Zorro» με καταλληλότητα «PG-13»

Και έχω προσθέσει την ταινία «Apocalypse Now» με καταλληλότητα «R»

Τότε θα πρέπει να δω το

«Apocalypse Now» πριν το «Zorro» στην αρχική σελίδα της RottenPotatoes

**3 βήματα αντί 15 βήματα:**  
**2 για την προετοιμασία του ελέγχου,**  
**1 για τη συμπεριφορά**

**Τα δηλωτικά σενάρια εστιάζουν την προσοχή στο χαρακτηριστικό που περιγράφεται και ελέγχεται αντί στα βήματα που χρειάζονται για την προετοιμασία του ελέγχου**

**Τι γίνεται με νέους ορισμούς βημάτων;**

95

95

## Ένα δηλωτικό σενάριο χρειάζεται νέους ορισμούς βημάτων

```

Given /I have added "(*)" Then /I should see "(*)"
  with rating "(*)"/ do      before "(*)" on (.) / do
  |title, rating|           |string1, string2, path|
Given I am on the Create    step "I am on #{path}"
New Movie page              regexp =
When I fill in "Title"      /#{string1}.*#{string2}/m
  with "#{title}"           # /m means match across
And I select "#{rating}"    newlines
  from "Rating"             page.body.should =~
And I press "Save           regexp
  Changes"                  end
end

```

- Καθώς εξελίσσεται η εφαρμογή, επαναχρησιμοποιήστε βήματα από τα πρώτα λίγα προστατικά σενάρια για να δημιουργήσετε πιο συνοπτικά και περιγραφικά σενάρια

96

96

Ποια απάντηση είναι ΣΩΣΤΗ για τα υπονοούμενα σενάρια σε σύγκριση με τα ρητά, και για τα δηλωτικά σενάρια σε σύγκριση με τα προστατικά;

1. Οι ρητές απαιτήσεις συνήθως ορίζονται με προστατικά σενάρια και οι υπονοούμενες απαιτήσεις συνήθως ορίζονται με δηλωτικά σενάρια
2. Τα ρητά σενάρια συνήθως αποτυπώνουν ελέγχους ολοκλήρωσης
3. Τα δηλωτικά σενάρια στοχεύουν στην αποτύπωση και της υλοποίησης και της συμπεριφοράς
4. Όλα είναι λάθος.

97

97

# ΤΕΛΟΣ

98

98



## Εκτίμηση κόστους στο Ευέλικτο μοντέλο

(Τεχνολογία Ανάπτυξης Λογισμικού ως Υπηρεσίας  
§7.4)

David Patterson

© 2013 David Patterson & David Patterson  
Licensed under [Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License](#)



99

99

## Εκτίμηση κόστους στο Ευέλικτο μοντέλο

---

- Σε πραγματικές συνθήκες, το κόστος πρέπει να εκτιμάται πριν ο πελάτης να συμφωνήσει για το έργο
- Εφόσον δεν υπάρχει προσεκτικός σχεδιασμός και χρονοπρογραμματισμός στο Ευέλικτο μοντέλο, πώς μπορούμε να εκτιμήσουμε το κόστος του έργου;

100

100

## Το μοντέλο της Pivotal Labs

---

- Η Pivotal Labs διδάσκει το Ευέλικτο μοντέλο στον πελάτη
- Χρησιμοποιώντας το Ευέλικτο μοντέλο, η Pivotal ποτέ δεν δεσμεύεται να παραδώσει τα χαρακτηριστικά X, Y και Z μέχρι την ημερομηνία D
- Αντίθετα, δεσμεύει πόρους για να εργαστεί με τον πλέον αποδοτικό μέχρι την ημερομηνία D
  - Ο πελάτης εργάζεται με την ομάδα για να ορίζει προτεραιότητες συνεχώς μέχρι την ημερομηνία D
- Εξακολουθεί να είναι απαραίτητη μια εκτίμηση του έργου

101

101

## Τρόπος εκτίμησης από την Pivotal Labs στο Ευέλικτο μοντέλο

---

1. 1 ώρα τηλεφωνική κλήση για επεξήγηση της μεθόδου
  - Κοινή προσπάθεια, δέσμευση χρόνου πελάτη, ...
2. Επίσκεψεις πελάτη για «καθορισμό εμβέλειας» 1,5 ώρας
  - Ο πελάτης φέρνει τον σχεδιαστή, τον προγραμματιστή, σχέδια
    - Οτιδήποτε που θα τον βοηθήσει να ξεκαθαρίσει τι ακριβώς θέλει να γίνει
  - Η Pivotal φέρνει 2 μηχανικούς που κάνουν ερωτήσεις
    - Προσπαθώντας να προσδιορίσει τι προσθέτει αβεβαιότητα στην εκτίμηση
3. Οι μηχανικοί χρειάζονται ½ ώρα για μια εκτίμηση εβδομάδων
  - Μικρή αντί μεγάλης αβεβαιότητας: 20-22 αντί 18-26 εβδομάδες
4. Προσφορά κόστους σε μορφή χρόνου και υλικών στον πελάτη

102

102



103

103

Ποια πρόταση είναι ΣΩΣΤΗ σχετικά με την εκτίμηση κόστους;  
(PL = Pivotal Labs)

1. Ως οπαδός της Ευέλικτης Ανάπτυξης, η PL δεν χρησιμοποιεί συμβόλαια
2. Ως οπαδός του προγραμματισμού σε ζεύγη, η PL εκτιμά το κόστος για 1 ζεύγος, το οποίο αντιστοιχίζει σε ολόκληρο το έργο
3. Το προσφερόμενο κόστος είναι για τον χρόνο και τα υλικά της PL που καλύπτουν τον αριθμό εβδομάδων στην εκτίμηση
4. Όπως δείχνουν σχετικές μελέτες, το 84%-90% των έργων ολοκληρώνονται εντός χρόνου και προϋπολογισμού, οι διευθυντές σχεδιασμού και τεκμηρίωσης υπόσχονται στους πελάτες ένα σύνολο χαρακτηριστικών για ένα προσυμφωνημένο κόστος μέχρι μια προσυμφωνημένη ημερομηνία

104

104



105

105





## Η προοπτική του σχεδιασμού και τεκμηρίωσης (Τεχνολογία Ανάπτυξης Λογισμικού ως Υπηρεσίας §7.10)

David Patterson

© 2013 David Patterson & David Patterson  
Licensed under [Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License](#)

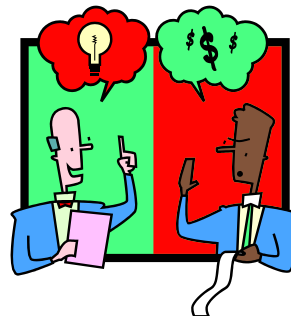


106

106

## Εισαγωγή

- Τι υπάρχει στο μοντέλο σχεδιασμού και τεκμηρίωσης αντί για
  - Ιστορίες χρηστών;
  - Σημεία;
  - Ταχύτητα;
- Πώς εκτιμά τα κόστη ένας διευθυντής; Πώς φτιάχνει το χρονοδιάγραμμα;



107

107

## Ισοδύναμα+ στον Σ&Τ

1. Εξαγωγή απαιτήσεων
2. Τεκμηρίωση απαιτήσεων
3. Εκτίμηση κόστους
4. Χρονοπρογραμματισμός και παρακολούθηση προόδου
5. Διαχείριση αλλαγών για απαιτήσεις, κόστος & χρονοδιάγραμμα
6. Διασφάλιση ταύτισης της υλοποίησης με τα χαρακτηριστικά απαιτήσεων
7. Ανάλυση & διαχείριση ρίσκου



108

108

## Εξαγωγή απαιτήσεων στον Σ&Τ

- Εξαγωγή *λειτουργικών* & *μη λειτουργικών* απαιτήσεων:

### 1. *Συνέντευξη* – δείτε πώς *πραγματικά* γίνεται σήμερα

- Οι ενδιαφερόμενοι απαντούν σε προκαθορισμένες ερωτήσεις
- Ή απλώς έχουν άτυπες συζητήσεις

### 2. Συνεργατική δημιουργία *σεναρίων*

- Αρχική κατάσταση, εμφάνιση ροής για χαρούμενα και λυπηρά μονοπάτια, τι είναι ταυτόχρονο, τελική κατάσταση

### 3. Δημιουργία *περιπτώσεων χρήσης*

- Λίστα βημάτων για να επιτευχθεί ο στόχος μεταξύ χρήστη και συστήματος· υπάρχει γλώσσα (UML) για την περιγραφή

109

109

## Τεκμηρίωση απαιτήσεων στον Σ&Τ

- Τεκμηρίωση απαιτήσεων μέσω  
*Προδιαγραφών Απαιτήσεων Λογισμικού*  
(*Software Requirements Specification, SRS*)  
– εκατοντάδες σελίδες· πρότυπο IEEE για SRS!
- Οι ενδιαφερόμενοι διαβάζουν τις SRS, ή κατασκευάζουν ένα βασικό πρωτότυπο, ή δημιουργούν περιπτώσεις ελέγχου για δοκιμή:
  - *Εγκυρότητα*: είναι αναγκαίες όλες οι απαιτήσεις;
  - *Συνέπεια*: υπάρχουν απαιτήσεις που έρχονται σε διένεξη;
  - *Πληρότητα*: περιλαμβάνονται όλες οι απαιτήσεις και οι περιορισμοί;
  - *Εφικτότητα*: μπορούν να υλοποιηθούν οι απαιτήσεις;



110

110

## Εκτίμηση κόστους στον Σ&Τ

- Ο διευθυντής διασπά τις SRS σε εργασίες
- Εκτιμά εβδομάδες ανά εργασία  
–  $1 \text{ εβδομάδα} \leq \text{Εργασίες} \leq 8 \text{ εβδομάδες}$
- Μετατρέπει τις ανθρωπο-εβδομάδες σε \$ μέσω μισθών και επιβαρύνσεων
- Εκτίμηση *πριν & μετά* το συμβόλαιο
  - Προσθήκη περιθωρίου ασφαλείας: 1,3 έως 1,5X
  - Γίνονται 3 εκτιμήσεις (βέλτιστη περίπτωση, αναμενόμενη, χειρότερη) και μετά βέλτιστη πρόβλεψη



111

111

## Εκτίμηση κόστους στον Σ&Τ

## 1. Πειραματική εκτίμηση

- Η ακρίβεια βασίζεται στην πείρα του διευθυντή

## 2. Ποσοτική εκτίμηση

- Εκτίμηση εργασιών σε γραμμές κώδικα (lines of code, LOC), διαίρεση με LOC/ανθρωπο-μήνα
- COCOMO (Constructive Cost Model):  

$$\text{Effort} = \text{OrgFactor} * \text{CodeSize}^{\text{Penalty}} * \text{ProdFactor}$$
  - Organization Factor = 2.94,  $1.10 \leq \text{SizePenalty} \leq 1.24$ ,  $0.90 \leq \text{Product Factor} \leq 1.40$

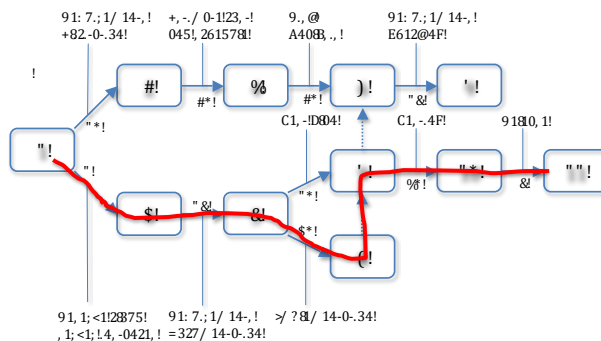
- 92% χρησιμοποιούν εμπειρική εκτίμηση αντί του τύπου

112

112

## Κατάρτιση χρονοδιαγράμματος στον Σ&Τ

- Χρήση διαγράμματος PERT για την εμφάνιση παραλληλισμού εργασιών και κρίσιμου μονοπατιού του χρονοδιαγράμματος



- Οι κόμβοι είναι ορόσημα, τα ονόματα συνδέσμων είναι εργασίες, οι αριθμοί συνδέσμων είναι προσπάθεια, τα βέλη είναι εξαρτήσεις

113

113

## Παρακολούθηση προόδου στον Σ&Τ

- Σύγκριση πρόβλεψης με πραγματικά στοιχεία
  - Χρόνος εργασιών
  - Έξοδα
- Το ενδιαμέσο ορόσημο βοηθά όλους τους ενδιαφερόμενους να δουν αν το έργο βρίσκεται εντός χρόνου και προϋπολογισμού



114

114

## Έλεγχος απαιτήσεων στον Σ&Τ

- Ο διευθυντής χρησιμοποιεί εργαλεία για την *ιχνηλασιμότητα των απαιτήσεων*
- Το εργαλείο έχει παραπομπές μεταξύ
  - Τμήματος των SRS με την απαίτηση
  - Τμήματος του κώδικα που υλοποιεί την απαίτηση
  - Ελέγχων που επικυρώνουν την απαίτηση



115

115

## Ανάλυση και διαχείριση κινδύνων στον Σ&Τ

- Για τη βελτίωση της ακρίβειας του προϋπολογισμού/χρονοδιαγράμματος
- Έγκαιρος προσδιορισμός κινδύνων για
  - Να γίνει επιπλέον δουλειά μείωσης κινδύνων
  - Αλλαγή σχεδίου για αποφυγή κινδύνων
- Τεχνικό: η RDB δεν κλιμακώνεται
- Οργανωτικό: J2EE;;;;
- Επιχειρηματικό: πολύ αργά να βγει στην αγορά
- Δημιουργία πίνακα κινδύνων:  
% πιθανότητα x επίδραση
  - Αντιμετωπίστε το κορυφαίο 20%, ελπίζοντας για 80% των κινδύνων



116

116

## Απαιτήσεις και εκτίμηση χρονοδιαγράμματος/ κόστους στον Σ&Τ και στο Ευέλικτο μοντέλο

Εργασίες	Στο μοντέλο σχεδιασμού και τεκμηρίωσης	Στο Ευέλικτο μοντέλο
Τεκμηρίωση Απαιτήσεων	Προδιαγραφή απαιτήσεων λογισμικού όπως το πρότυπο IEEE 830-1998	Ιστορίες χρηστών, Cucumber, σημεία, Ταχύτητα
Εξαγωγή απαιτήσεων	Συνεντεύξεις, σενάρια, περιπτώσεις χρήσης	
Διαχείριση αλλαγών για απαιτήσεις, χρονοδιάγραμμα, και προϋπολογισμό	Έλεγχος εκδόσεων για τεκμηρίωση και κώδικα	
Διασφάλιση χαρακτηριστικών απαιτήσεων	Ιχνηλασιμότητα για σύνδεση των χαρακτηριστικών με τους ελέγχους, τις αναθεωρήσεις και τον κώδικα	Αξιολόγηση για την επιλογή του εύρους της προσπάθειας όσον αφορά το συμβόλαιο χρόνου και υλικών
Χρονοπρογραμματισμός και παρακολούθηση	Νωρίς στο έργο, ημερομηνία παράδοσης βασισμένη στην εκτίμηση κόστους με συμβόλαιο, χρήση διαγραμμάτων PERT. Ορόσημα για την παρακολούθηση της προόδου.	
Εκτίμηση κόστους	Νωρίς στο έργο, κόστος βασισμένο στην πείρα του διαχειριστή με χρήση συμβολαίου, ή εκτιμήσεις του μεγέθους εργασιών συνδυασμένες με μέτρα παραγωγικότητας	
Διαχείριση κινδύνων	Νωρίς στο έργο, προσδιορισμός κινδύνων στον προϋπολογισμό και το χρονοδιάγραμμα, και ανάληψη δράσεων για την υπέρβαση ή την αποφυγή τους	

Εικόνα 7.14: Η σχέση μεταξύ των εργασιών που σχετίζονται με τις απαιτήσεις στη μεθοδολογία σχεδιασμού και τεκμηρίωσης και στην Ευέλικτη μεθοδολογία.

117

117



118

118

Ποια πρόταση σχετικά με τις απαιτήσεις και την εκτίμηση κόστους στον Σ&Τ είναι ΛΑΘΟΣ;

1. Τα πλησιέστερα αντίστοιχα στις εργασίες χρονοπρογραμματισμού και παρακολούθησης στο μοντέλο Σ&Τ είναι τα σημεία και η ταχύτητα του Ευέλικτου μοντέλου
2. Το πλησιέστερο αντίστοιχο στο έγγραφο Προδιαγραφών Απαιτήσεων Λογισμικού (SRS) του Σ&Τ είναι οι ιστορίες χρηστών του Ευέλικτου μοντέλου
3. Δεν υπάρχει ισοδύναμο στο Ευέλικτο μοντέλο για τη διασφάλιση των απαιτήσεων, όπως η ιχνηλατησιμότητα
4. Στην πραγματικότητα, τα 1, 2, και 3 είναι όλα σωστά· κανένα δεν είναι λάθος

119

119



120

120

## Εν κατακλείδι: §§9.4-9.5, 7.1-7.4, 7.7

- Μέθοδοι SOFA: Σύντομη (Short), κάνουμε Ένα πράγμα (do One thing), Λίγα ορίσματα (Few arguments), συνεπής Αφαίρεση (consistent Abstraction)
  - Τα μετρικά υποδεικνύουν προβληματικά σημεία του προγράμματος
- BDD: Ιστορίες χρηστών για την εξαγωγή απαιτήσεων
  - SMART: Συγκεκριμένη (Specific), Μετρήσιμη (Measureable), Επιτεύξιμη (Achievable), Σημαντική (Relevant), Χρονικά περιορισμένη (Timeboxed)
- Σημεία/Ταχύτητα για τον υπολογισμό της προόδου (Tracker)
- Διεπαφή χρήστη χαμηλής πιστότητας/εικονοσενάρια: σκίτσο για τη σχεδίαση της διεπαφής χρήστη
- Κύκλοι ζωής: σχεδιασμός & τεκμηρίωση (προσεκτικός σχεδιασμός & τεκμηρίωση) αντί του Ευέλικτου μοντέλου (αποδοχή αλλαγής)

121

121



Πλάνες & παγίδες,  
πλεονεκτήματα &  
μειονεκτήματα της  
BDD, τέλος  
Κεφαλαίου 7  
(Τεχνολογία Ανάπτυξης  
Λογισμικού ως Υπηρεσίας  
§7.10-§7.12)



David Patterson

© 2013 Armando Fox & David Patterson, all rights reserved

122

122

## Παγίδες

- Παράδοση μιας ιστορίας ως «ολοκληρωμένης» όταν έχει ελεγχθεί μόνο το χαρούμενο μονοπάτι
  - Πρέπει να ελεγχθούν τόσο το χαρούμενο μονοπάτι όσο και το λυπηρό μονοπάτι
- Η σωστή συμπεριφορά της εφαρμογής όταν ο χρήστης κάνει ακούσια το λάθος πράγμα είναι το ίδιο σημαντικό με τη σωστή συμπεριφορά όταν κάνει το σωστό πράγμα
  - Το σφάλλιν ανθρωπινον

123

123

## Παγίδες

---

- Απρόσεκτη χρήση αρνητικών προσδοκιών
  - Προσοχή στην κατάχρηση του «Τότε δεν θα πρέπει να βλέπω...» (“Then I should not see...”)
  - Δεν μπορείτε να γνωρίζετε αν η έξοδος είναι αυτό που θέλετε, μόνο ότι δεν είναι αυτό που θέλετε
  - Υπερβολικά πολλές έξοδοι είναι εσφαλμένες
- Συμπερίληψη θετικών για τον έλεγχο των αποτελεσμάτων «Τότε θα πρέπει να βλέπω...» (“Then I should see ...”)

124

124

## Παγίδες

---

- Απρόσεκτη χρήση θετικών προσδοκιών
  - Τότε θα πρέπει να βλέπω “Emma” (Then I should see “Emma”)  
τι γίνεται αν η συμβολοσειρά εμφανίζεται πολλές φορές στη σελίδα;
  - Μπορεί να περάσει με επιτυχία ακόμη και αν το χαρακτηριστικό δεν λειτουργεί
- Χρήση της βοηθητικής συνάρτησης `within` του Capybara
  - Περιορίζει την εμβέλεια των συναρτήσεων ταύτισης (matchers) σε έναν επιλογέα CSS
  - Then I should see “Emma” within “div#shopping\_cart”
  - Δείτε την τεκμηρίωση του Capybara

125

125

Ποια πρόταση είναι ΛΑΘΟΣ σχετικά με τη διεπαφή χρήστη χαμηλής πιστότητας και τη BDD;

1. Ο σκοπός της προσέγγισης διεπαφής χρήστη χαμηλής πιστότητας είναι η αποσφαλμάτωση της διεπαφής χρήστη πριν τον προγραμματισμό της
2. Ένα μειονέκτημα της BDD είναι η απαίτηση της συνεχούς επαφής με τους πελάτες, που μπορεί να μην είναι εφικτή
3. Ένα μειονέκτημα της BDD είναι ότι μπορεί να οδηγήσει σε μια ανεπαρκή αρχιτεκτονική λογισμικού, αφού εστιάζει στη συμπεριφορά
4. Κανένα δεν είναι λάθος· και τα τρία είναι σωστά

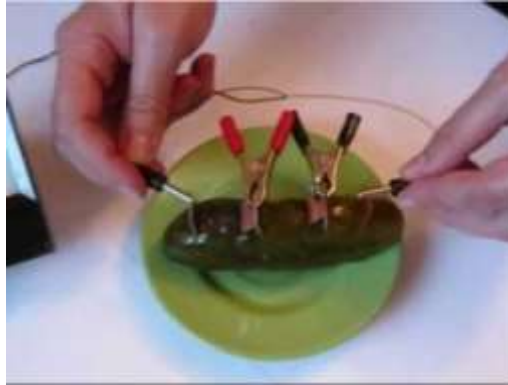
126

126



127

127



---

128

## Χρήση των Cucumber/Capybara

- Πραγματοποιήστε αναζήτηση στη Google με τις λέξεις `capybara cheat sheet` ή `rspec rails cheatsheet capybara`
- Το μαγικό αντικείμενο: `page`
- Εύρεση στοιχείων μέσω CSS ή XPath (βήματα του τύπου "Then I should see")
  - `find(:css, ...), find(:xpath)`
  - `page.should have_css('...')`,  
`page.should have_xpath('...')`
  - Σωστό & συνιστώμενο για τη διαγραφή/αντικατάσταση ορισμών βημάτων "Then I should see" από το `web_steps.rb`
- Επιλογή ημερομηνιών και άλλων σύνθετων αντικειμένων

129

---

129

- 
- Διαχείριση δεδομένων σε πίνακες
  - Συνόψεις σεναρίων
  - Μεταβίβαση κατάστασης από βήμα σε βήμα με χρήση μεταβλητών στιγμιοτύπων
  - Ταξίδι στον χρόνο με το Timecop
  - Σχετικό: χρήση ετικετών (tags) για συνθήκες σεναρίων

130

130



131

131