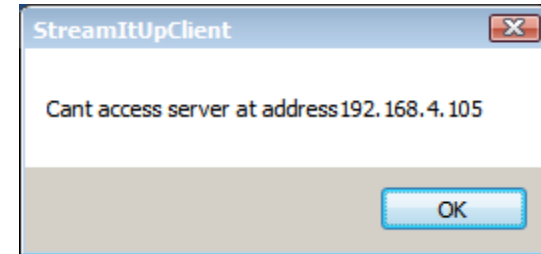


הלל: מה ששנא עליך על
תעשה לחברך



הנדסת תוכנה

13. חווית משתמש – UX

שחרור תוכנה + כלים ||

"I have always wished for my computer to be as easy to use as my telephone; my wish has come true because I can no longer figure out how to use my telephone"

[Bjarne Stroustrup](#) (C++ Creator)



http://groups.google.com/group/microsoft.public.powerpoint/browse_thread/thread/27c8f0b03f69fd4d?hl=en&ie=UTF-8&q=powerpoint+undo+deleted+slides#6608dc06dad9ca8a



microsoft.public.powerpoint

can i get a deleted slide back?

★ 3 messages - [Collapse all](#)

Crowe [View profile](#)

I **deleted** a slide from my **powerpoint** slide and hit "save". I need that slide back! Any way to get it???

sigh

Thanks!

[Reply](#) [Reply to author](#) [Forward](#)

Rate this post: ★★★★★

Sandy Johnson [View profile](#)

Dear Crowe,

Sorry can't get it back -- unless you saved it earlier under a different name or perhaps emailed the file to someone previous to saving it...

היום?



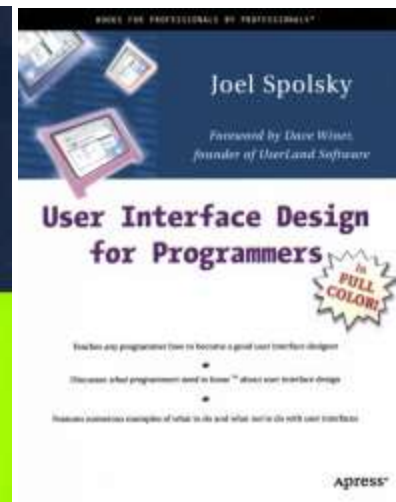
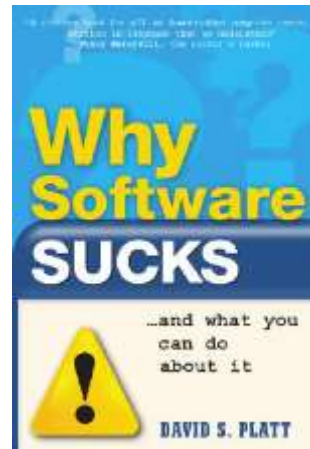
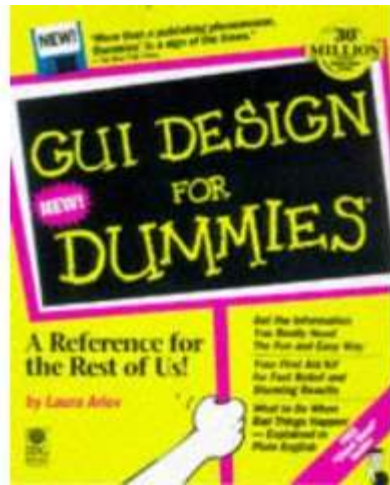
- חווית משתמש
 - שמישות \ עיצוב ממשק משתמש
 - GUI מסורתי, web והתקנים ניידים
- שחרור תוכנה
- כלים מתקדמים
- שעה 3 \ תרגיל – משימת סבב אחרון
 - Code Review with pull requests
 - סביבת CI / CD (בונוס)
- הכנה להרצאה האחרונה (סקר)

Brooks (MMM): Question:
How does a large software
project get to be one year
late?
se14b-yagel

Answer: One day at a time!

מקורות

- Pressman Chap. 12
- ברק דנין, מדריכי שמישות (ול-סמארטפון)
- Amber, Agile Usability
- Spolsky, User Interface Design For Programmers
<http://www.joelonsoftware.com/primerFriendly/uibook/fog0000000249.html>
- Usability In Practice, MSDN series



Usability in the Front

- [Windows 8 UX Manager](#)
- [The Science and Art of User Experience at Google](#) (gtalk example)
- [Google Hangout Button](#) (btw [readability](#)...)



מאפייני מערכת לדוגמא

- פונקציונליות
- גודל
- ביצועים
- מחיר
- אמינות
- בטיחות
- סטנדרטים
- חווית\ממשק משתמש\שמישות
- בתיכון מערכת צריך לשקלל מאפיינים שונים
- היום דגש מסוים

סוגי ממשקים

- ממשק בין רכיבים
- ממשק למערכות חיצוניות
- ממשק משתמש

שלשת חוקי הזהב [Mandel 97]

- Place the user in control
- Reduce the user's memory load
- Make the interface consistent

Place the User in Control

- Define **interaction** modes in a way that does not force a user into unnecessary or undesired actions.
- Provide for flexible interaction.
- Allow user interaction to be interruptible and undoable.
- Streamline interaction as skill levels advance and allow the interaction to be customized.
- Hide technical internals from the casual user.
- Design for direct interaction with objects that appear on the screen.

Reduce the User's Memory Load

- Reduce demand on short-term memory.
- Establish meaningful defaults.
- Define shortcuts that are intuitive.
- The visual layout of the interface should be based on a real world metaphor.
- Disclose information in a progressive fashion.

Make the Interface Consistent

- Allow the user to put the current task into a meaningful context.
- Maintain consistency across a family of applications.
- If past interactive models have created user expectations, do not make changes unless there is a compelling reason to do so.

ממשק משתמש – לא רק ציור

- עיצוב אינטראקטיביות משתמש
- ארגון המידע
- מחקר משתמשים
- עיצוב ויזואלי

חווית משתמש (נילסן\נורמן)

"חווית משתמש כוללת את כל האספקטים של האינטראקציה בין משתמש הקצה לבין החברה, השירותים שלה והמוצרים שלה. הדרישה הבסיסית לחווית משתמש מעולה היא לענות על הצרכים המדויקים של המשתמש, ללא סיבוכים מיותרים. על המוצר או השירות להיות פשוט ואלגנטי, **כך שיהיה כיף להחזיק בו, וכיף להשתמש בו**. חווית משתמש אמיתית נותנת ללקוחות הרבה יותר מאשר בדיוק את מה שהם אומרים שהם רוצים, או ממלאת אחרי רשימת יכולות ודרישות. כדי להגיע לחווית משתמש באיכות גבוהה על-פני כל החברה, חייב להיות **רצף אחד על-פני כל השירותים** שהיא מספקת בתחומים שונים, כולל הנדסה, שיווק, עיצוב גראפי ותעשייתי, ועיצוב ממשק".

Usability - שמישות

- מידת האפקטיביות שבה משתמש יכול להשיג את מטרותיו עם התוכנה
- חלק מ- Human Computer Interaction, חיתוך עם מדעים קוגניטיביים \ ארגונומיה-פיזיולוגיה \ הנדסת אנוש
- המשתמש במרכז!
- ISO Definition
- Morning: designed for use (video)

DILBERT by Scott Adams





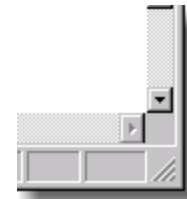
מאפייני שמישות (Nielsen)

- **לימודיות** - המידה שבה הממשק ניתן ללמידה על ידי המשתמש
- **יעילות** - המידה שבה השימוש בממשק הוא יעיל
- **זכירות** - מידת הקלות שבה המשתמש זוכר את פעולות הממשק.
- **שגיאות** - כמות הטעויות הפוטנציאלית הקיימת בממשק
- **שביעות רצון** - מידת שביעות הרצון של המשתמש מהממשק

Affordance - מושג קרוב: נגישות |

- מצב שבו המאפיינים החושיים של אובייקט מרמזים על אופן השימוש בו

– כפתור בולט, מציע שאמורים בד"כ ללחוץ עליו
– פינה עבה של חלון מרמזת על אפשרות להגדלה



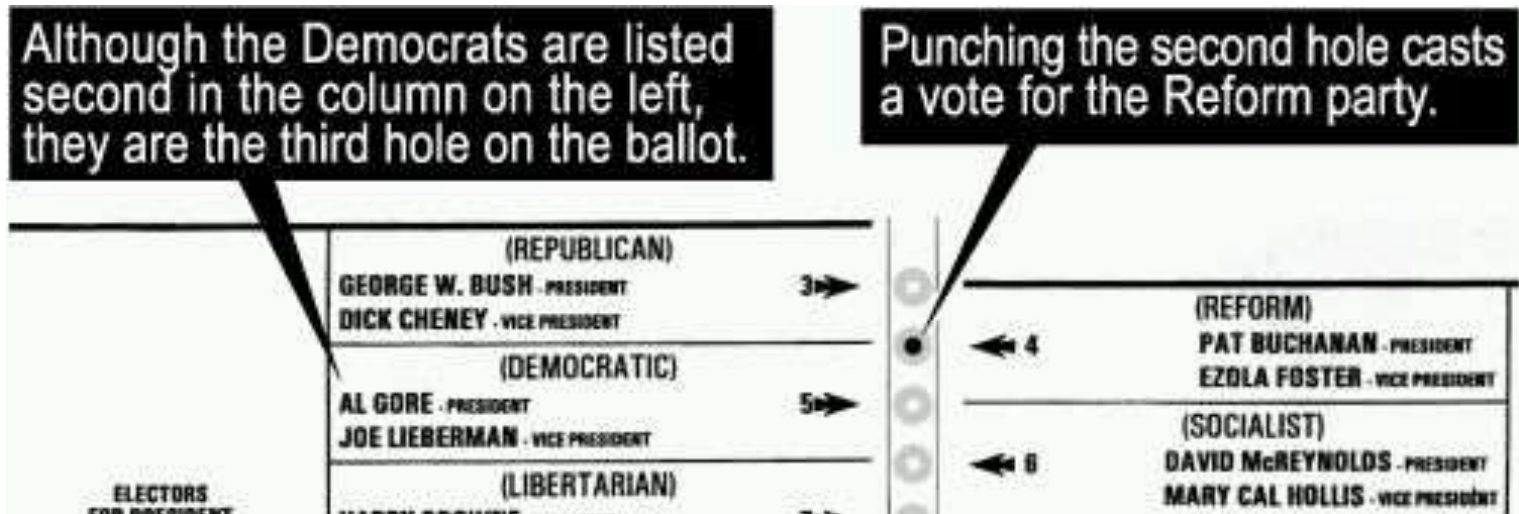
- תכונה רצויה לממשק משתמש, עוד בהמשך...
- [Affordance Definition](#), [Udacity Course](#)

נגישות II - Accessibility

- נגישות למרות מוגבלויות שונות
 - ראייה
 - מוטוריקה
- נגישות למניעת מחלות וכדו'
- תקנים:
- Web Content Accessibility Guidelines (WCAG)
 - <http://www.w3.org/TR/WCAG20/>
- <http://www.access-board.gov/508.htm>
- Tools: Herra, Booby

שמישות וממשק משתמש

- קשר הדוק בין השניים
- לממשק גרוע יכולה להיות השפעה רבה



Apollo Usability

- We have a problem
- UI issue?
- Russ Olsen - To the Moon!
@ a Clojure conf. 2013

כיצד משיגים שמישות?

- מבדקי שמישות (Usability Testing)
 - Eye Tracking
 - ראיונות\סקרים עם מומחים\ קבוצות מיקוד
 - אב טיפוס
 - נייר
 - קוד (VB, Rails)
 - פיתוח איטרטיבי
 - במיוחד ברשת
 - תהליכי עיצוב



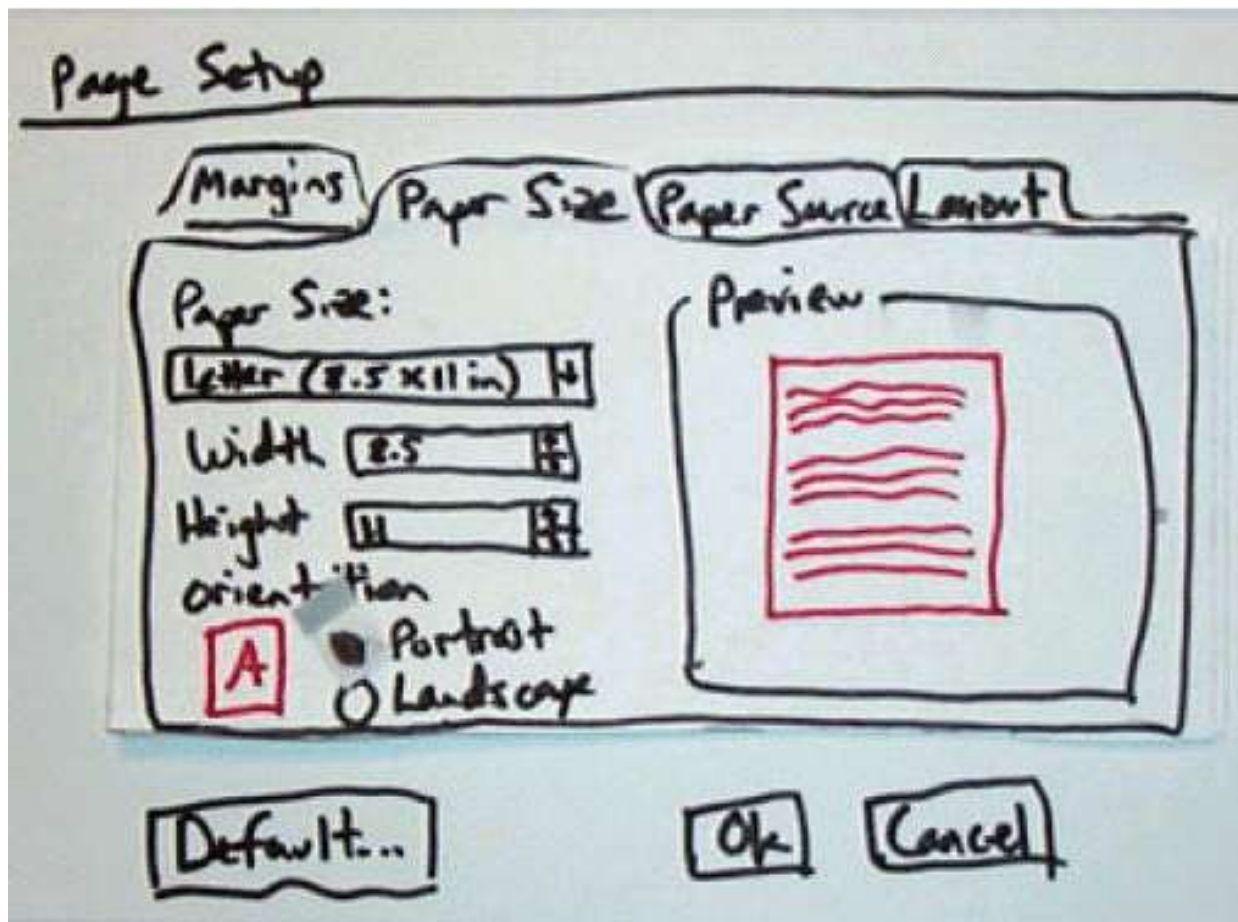
DILBERT reprinted by permission of United Feature Syndicate.

אב טיפוס מנייר

- משחק תפקידים, ללא מחשב



אב טיפוס מנייר





יתרונות לנייר

- בדיקת הרעיונות לפני המימוש
- ביצוע שינויים מהיר
- אפשרות לגלות מה באמת צריך
- נטרול משתנים טכנולוגיים ומימושיים
 - עוד צעד לקראת הלקוח
 - שינויים נראים קלים יותר
 - התמקדות בדברים העקרוניים

בתוכנה



- סביבות פיתוח מהירות, למשל:

– [Ruby on Rails](#)

– Nancy, [ASP.NET MVC](#)

- סביבות אבות טיפוס:

– לאתרים: <https://gomockingbird.com/>

– [SketchFlow](#), MS Expression Blend, [Video](#)

– [BalsamiQ Mockups](#), [וידאו](#)

– <http://www.pretotyping.org/androgen>

– <http://astrails.com/> איך עושים זאת? למשל:

– VS2012 Story Boarding

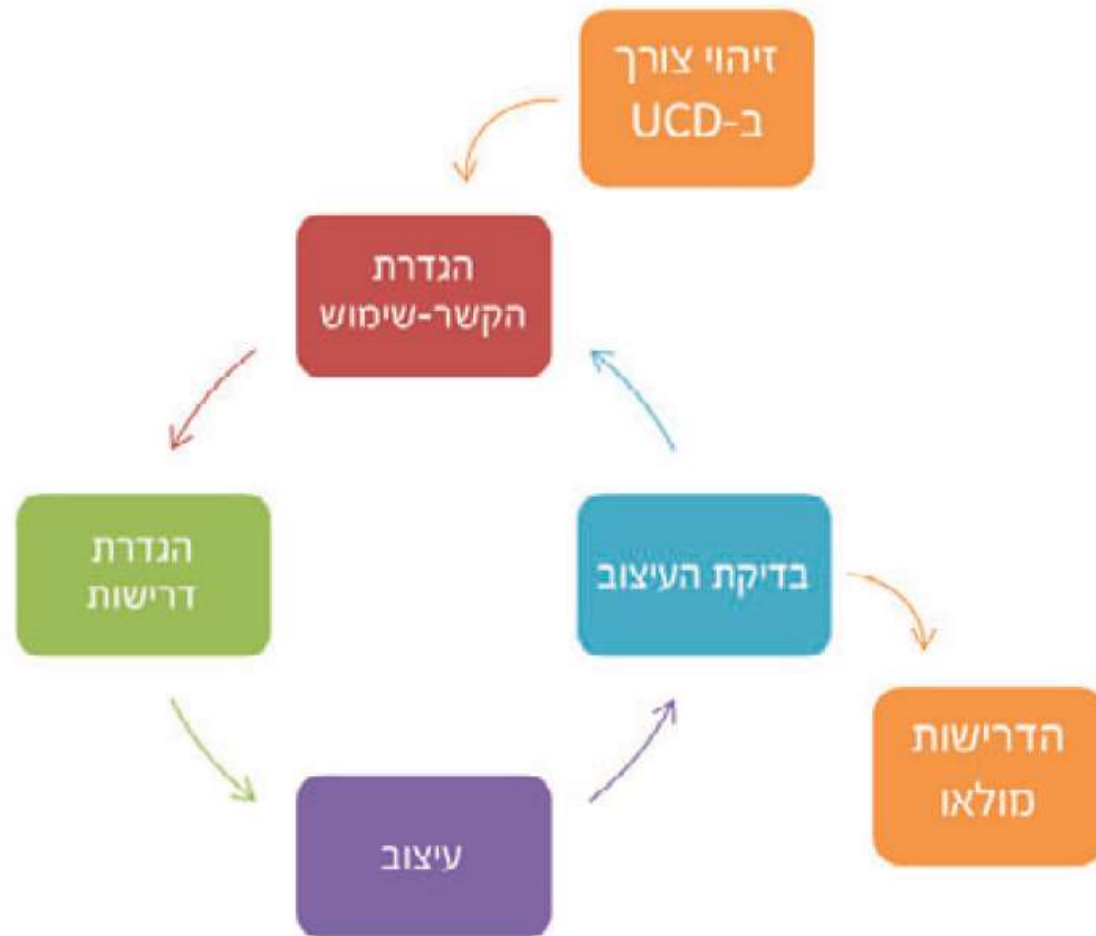
- בדיקות אוטומטיות, למשל [Selenium](#)

– Julian Harty, [Finding Usability Bugs with Automated Tests](#)

איזה סוג בדיקה כללית שלמדנו הכי משמעותי לחוויית משתמש

1. בדיקות קבלה (Acceptance)
2. בדיקות שמישות (מעבדת שמישות)
3. סקר דרישות
4. בדיקות יחידה

תהליך עיצוב מוכוון משתמש (UCD)



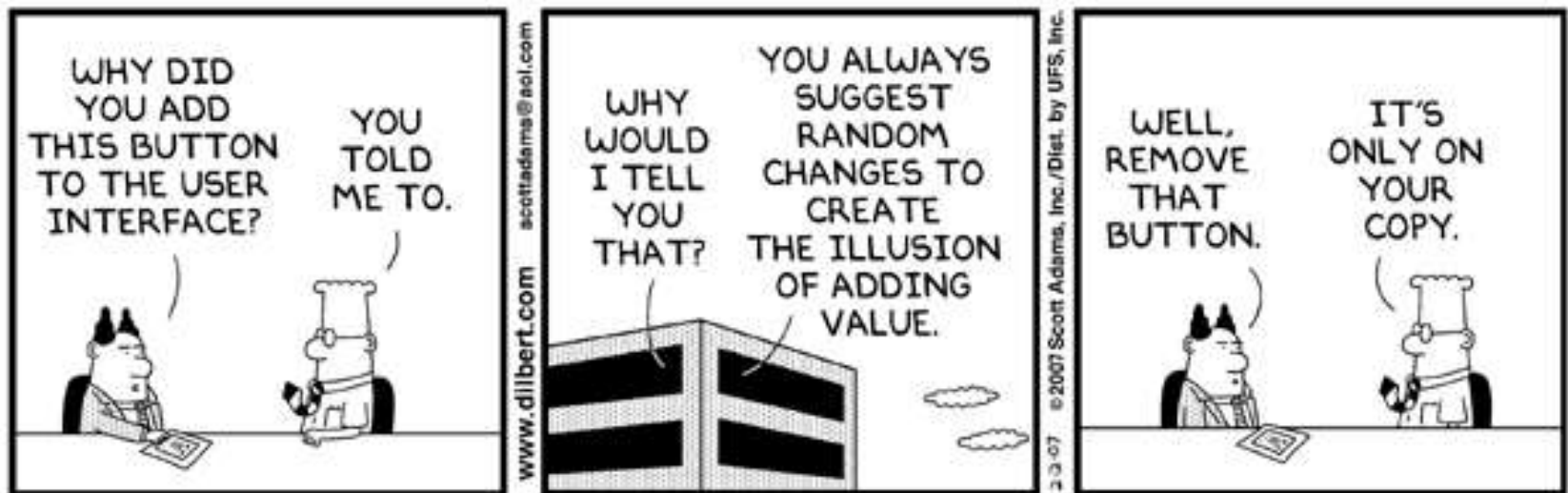
מתי בודקים?

- “The rule of thumb...is that the cost-benefit ratio for **usability** is **\$1:\$10-\$100**. Once a system is in **development**, correcting a problem costs 10 times as much as fixing the same problem in **design**. If the system has been **released**, it costs 100 times as much relative to fixing in design.” (Gilb, 1988, [here](#))

User Centered Design •

עוד לא בדקתם? •

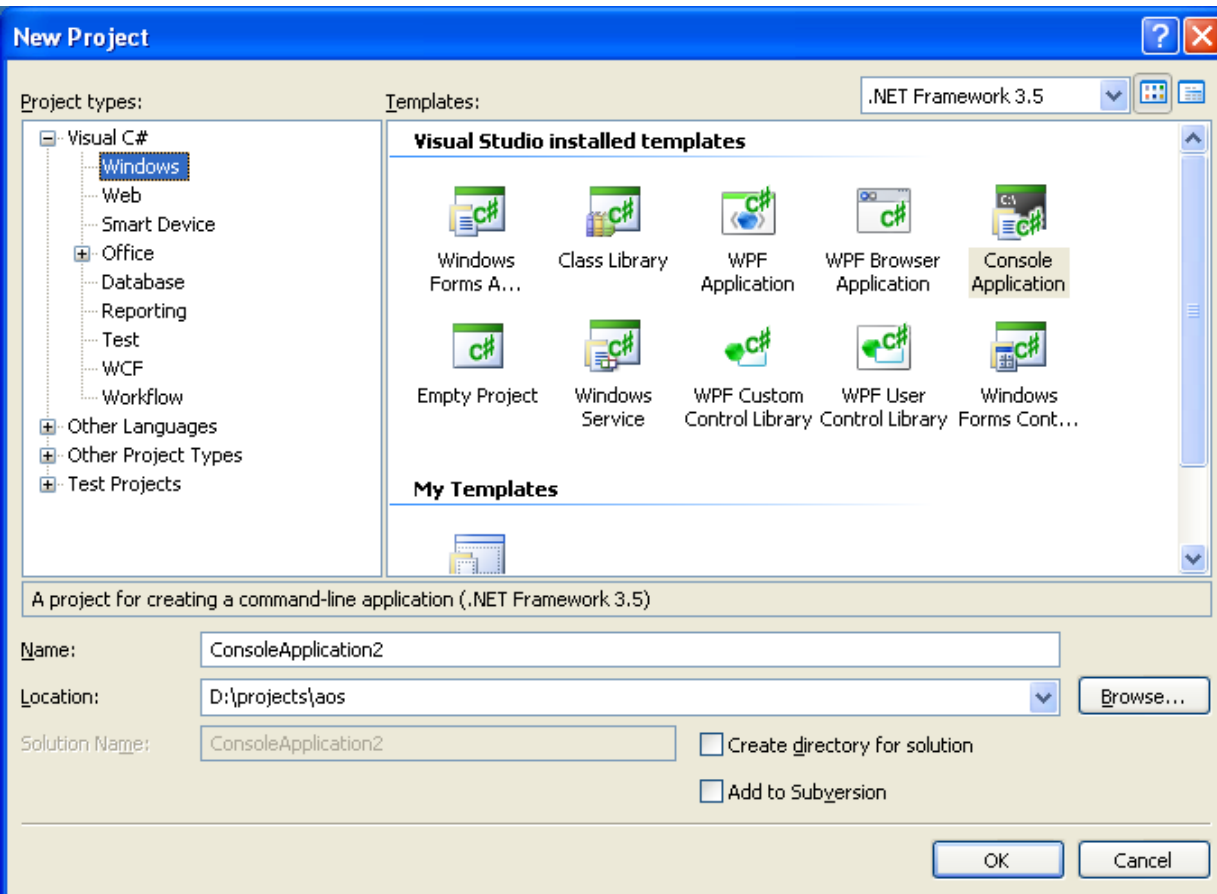
כמה עקרונות בסיסיים לתוכניות חלונאיות



© Scott Adams, Inc./Dist. by UFS, Inc.

מתי כדאי להשתמש ב:

- כפתור?
- תיבת סימון?
- כפתור רדיו?
- שדה טקסט?
- רשימה/נגללת?
- עץ?
- תפריט?
- תיבת שיחה?
- אחרים....?



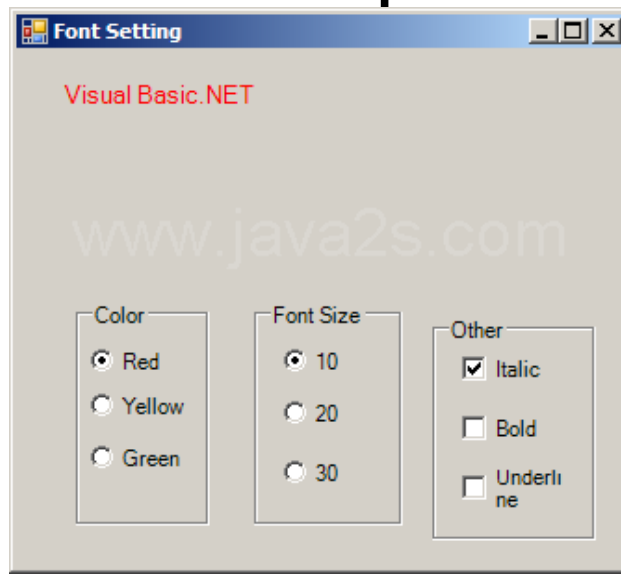
כפתורים, תפריטים

- כפתור – עבור פעולות עצמאיות, רלוונטיות למסך הנוכחי
 - מתי שמים ... (שלש נקודות)?
- סרגל כלים – פעולות נפוצות
- תפריט – פעולות פחות נפוצות שיכולות להיות ישימות למספר מסכים (או כולם)



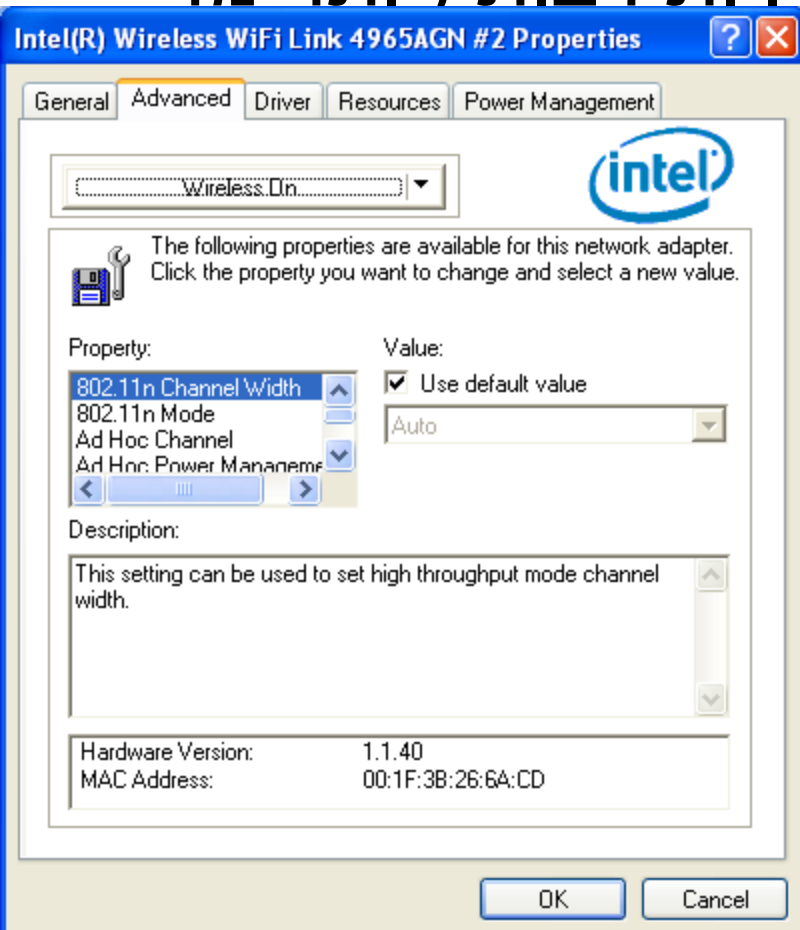
תיבות סימון

- Check-box – מפסקי הפעלה וכיבוי, כל אחד בלתי תלוי באחרים (לרוב תואם לערכים בוליאניים)
- Radio button – אפשרויות קשורות שרק אחת צריכה להיבחר (לרוב תואם לקבועים\enum)



רשימות

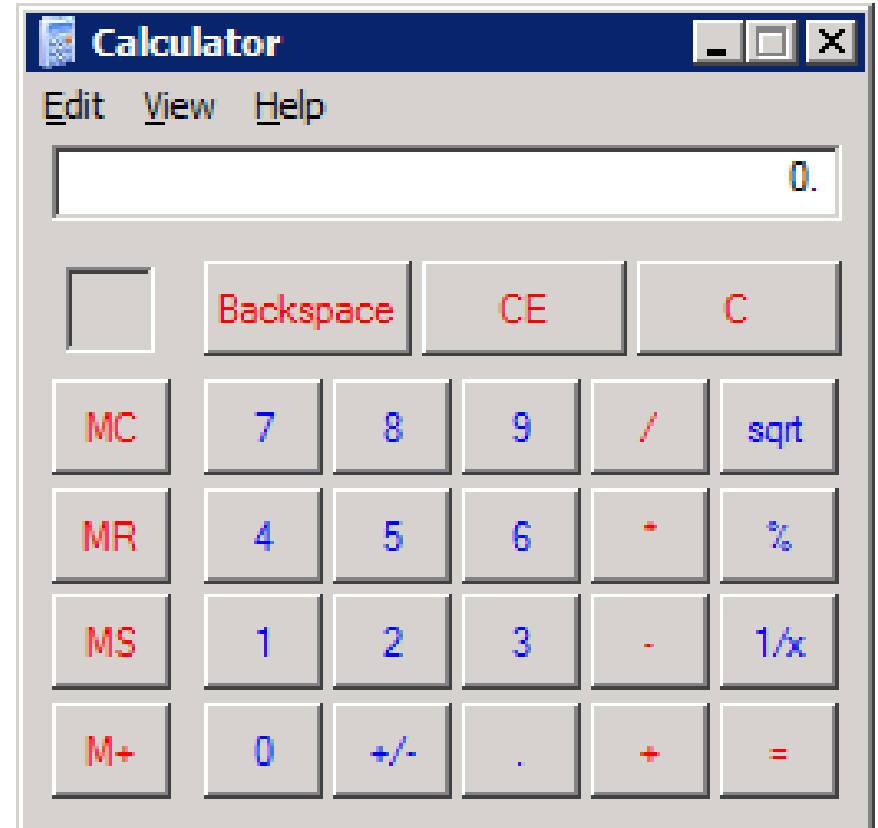
- טקסט (בד"כ בצרוף תגית) – קלט חופשי
- רשימה – בחירה מתוך אפשרויות רבות (יותר מדי בשביל רדיו) ורוצים שיראו את כולן
- Combo-box – כנ"ל, אך רוצים לחסוך במקום על המסך



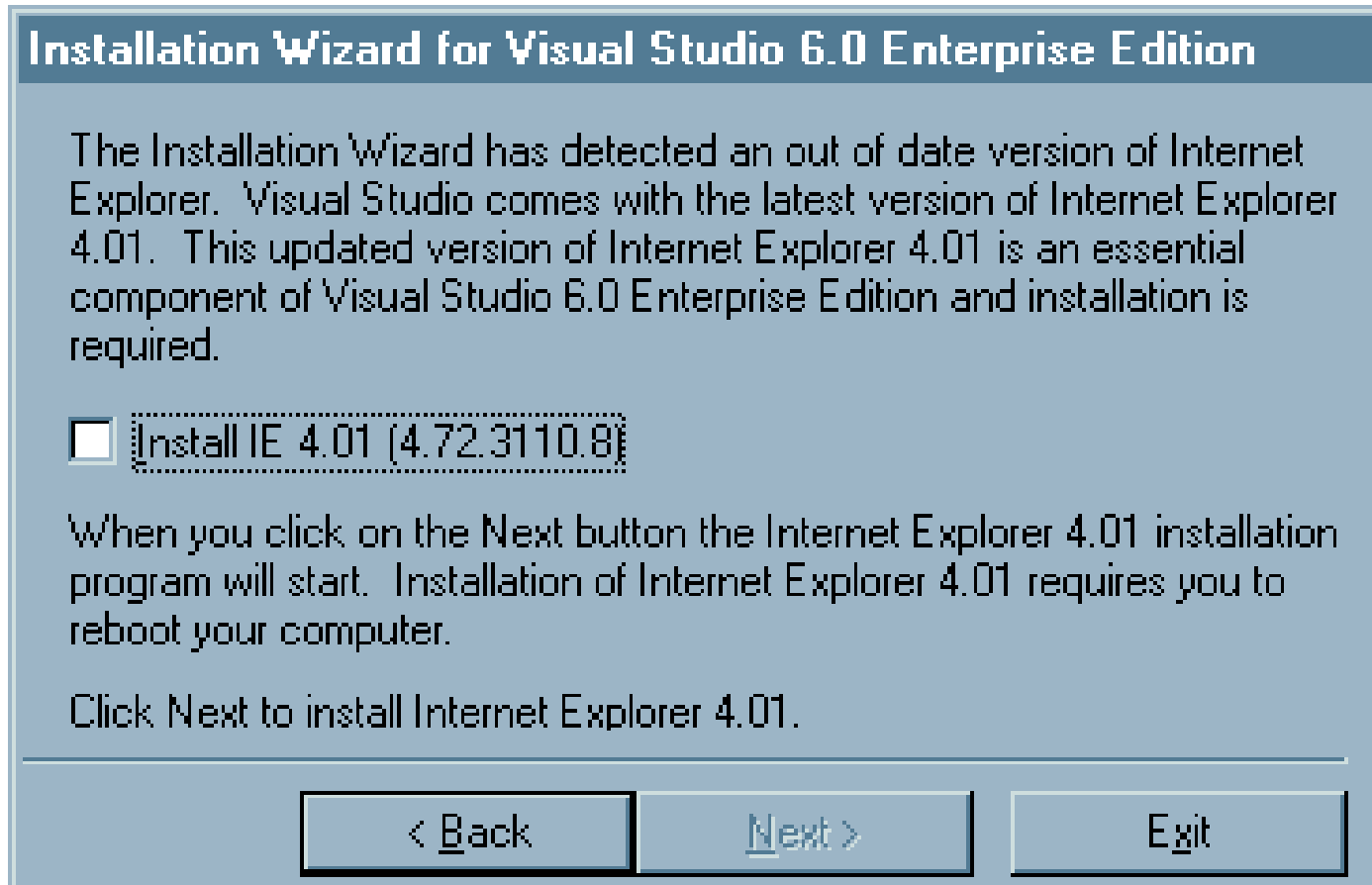
מסכים מרובים

- Tabbed Pane – המשתמש יכול לעבור בכל רגע בין המסכים השונים
- אשף (wizard) – הנחיית המשתמש לאורך תהליך
- תיבת שיחה – הצגת מסכים זמניים או תיבת מאפיינים

על ספסל הנאשמים



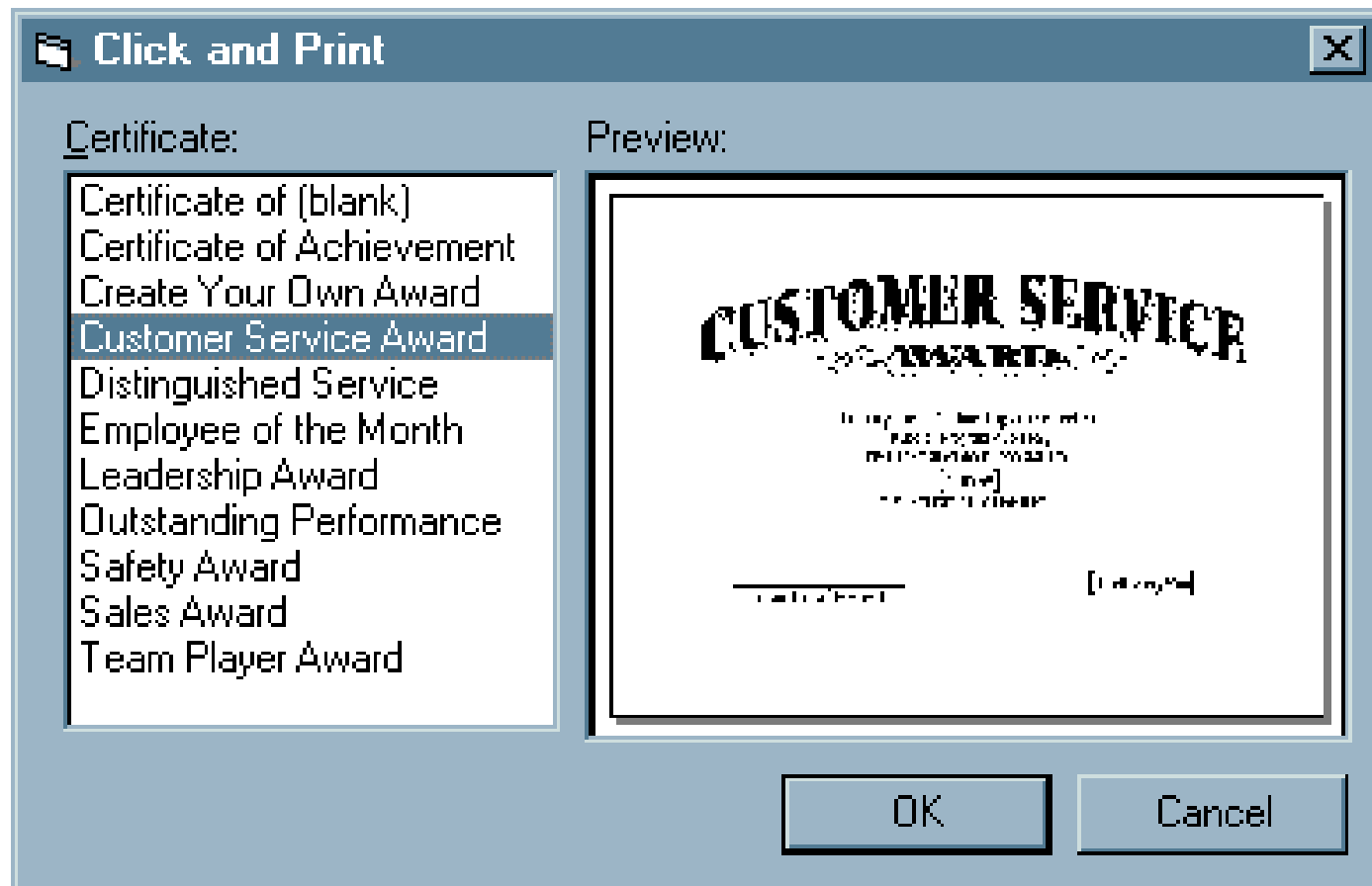
Interface Hall of Shame - דוגמאות



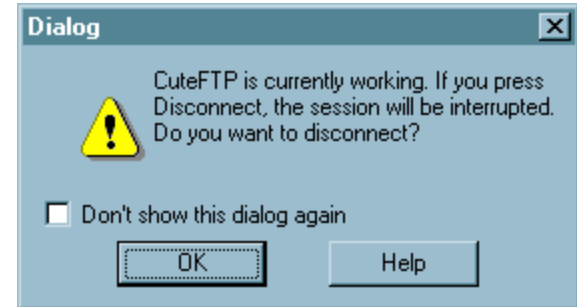
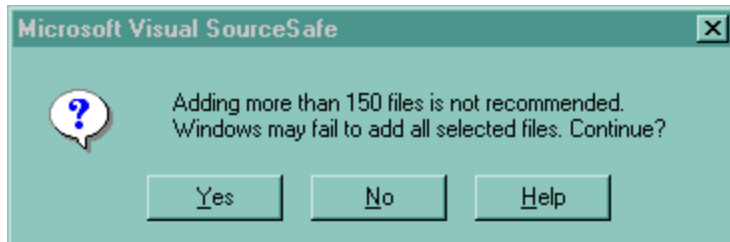
Interface Hall of Shame - דיון



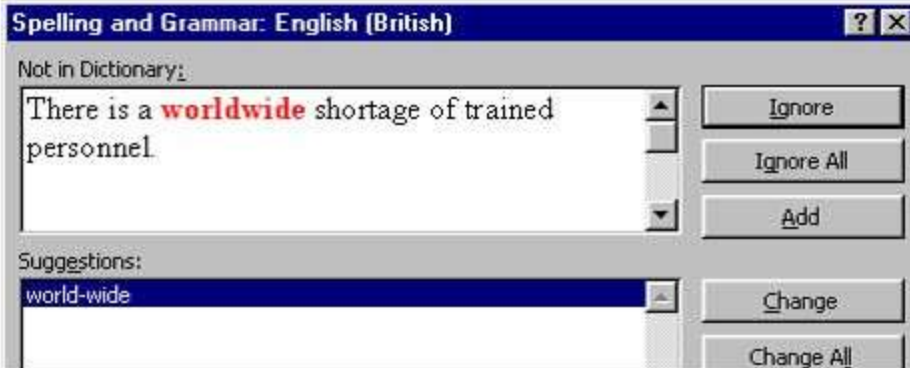
קצת יותר טוב



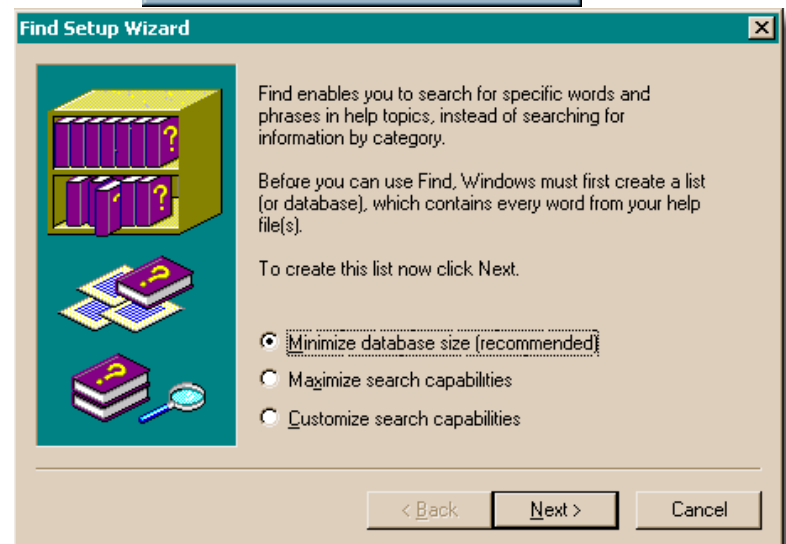
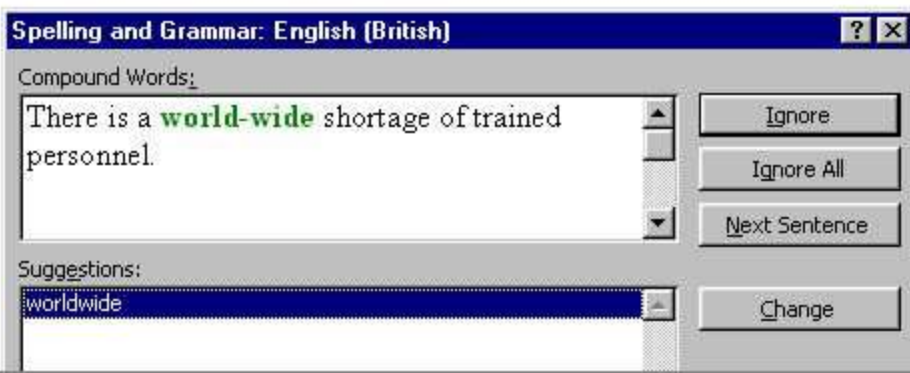
עוד דוגמאות והודעות שגיאה



There is a **worldwide** shortage of trained personnel.



There is a world-wide shortage of trained personnel.



D. Jackson , MIT

- TLV Lecture
<http://people.csail.mit.edu/dnj/talks/tel-aviv14/tel-aviv-sackler-anim-14.pdf>
- P. 35 OSX Alarm

שמישות ברשת (+300M\$)

Login

You must be a registered user to proceed. Please login or register below.

Already registered? Login

Username:

Password:

Login

☐ Remember my username on this machine.

[Forgot your login information? Click here.](#)

New to the Site?

Sign up in a few easy steps!

Register Now

Login

You must be a registered user to proceed. Please login or register below.

Already registered? Login

Username:

Password:

Login

☐ Remember my username on this machine.

[Forgot your login information? Click here.](#)

New to the Site?

Continue

You do not need to create an account to make purchases on our site. Simply click **Continue** to proceed to checkout.

To make your future purchases even faster, you can create an account during checkout.

se14b-yag

42

דברים שלא כל המשתמשים יודעים

- הלוגו מקשר לעמוד הבית



- אבטחה מול פרטיות



NetBank® Banking how you live.
Member FDIC

Apply Now

Get Free Bill Pay & Pay No Fees
NetValue Interest Checking

Only \$50 deposit to open, free unlimited transactions. [Apply Now](#)

Banking ▶
Investments ▶
Loans
About NetBank
Demos
Contact Us
Home
Security & Privacy
Site Map

Overview
Home Equity
Mortgages
Business Leasing
Loan Glossary
Loan Disclosures

Services	Our Rates
Money Market Account	4.75% APY* open
1 Year CDs (other terms available)	5.26% APY* open
Home Equity Loans (intro rate fixed for 3 months)	6.99% APR open

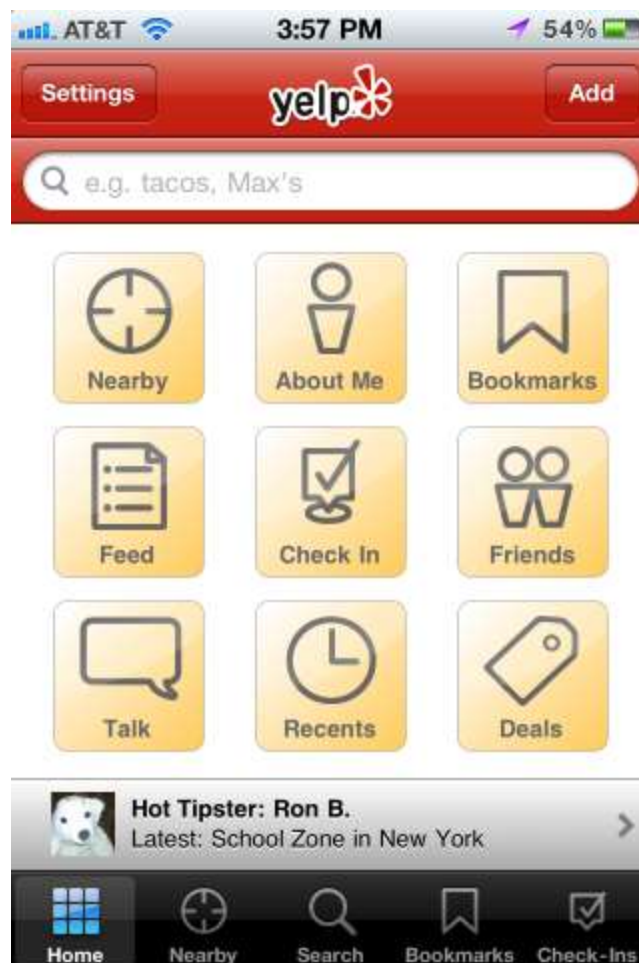
- שימוש בתפריטים נגללים
- כפתורי חיצים לרשימות
- ניווט באמצעות URL-ים
- הכרות עם כפתורי הדפדפן
- פתיחת חלון שני
- Web 2.0

דוגמת שמישות ברשת

- <http://graphs.gapminder.org/world/>
- => Hans Rosling's 200 Countries, 200 Years, 4 Minutes - The Joy of Stats

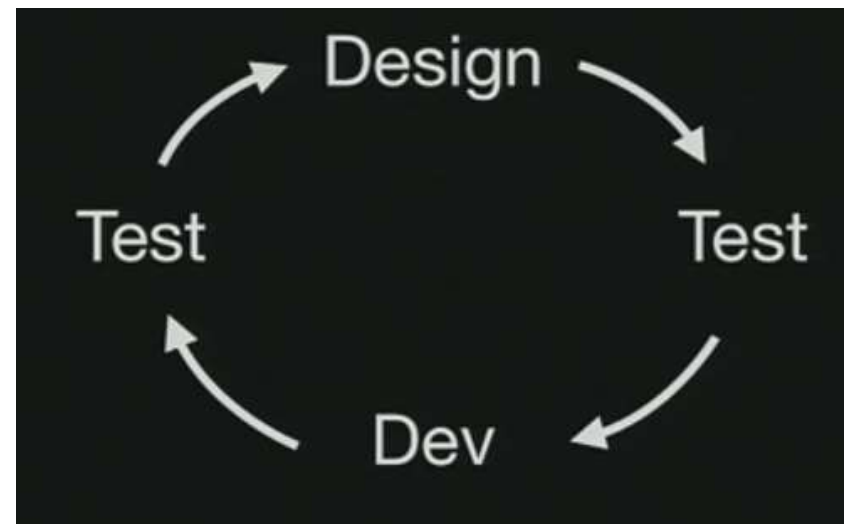


שמישות במחשוב נייד



עיצוב חווית משתמש

- Aral Balkan, “[A happy grain of sand](#)”, NDC 2012
 - Flush
 - Elevator
 - Washing Machine
 - DVD Burner @ Apple
 - Ticket Machine [44:10](#)
 - Phone network
- Bach, Exploratory testing of “[easy button](#)”



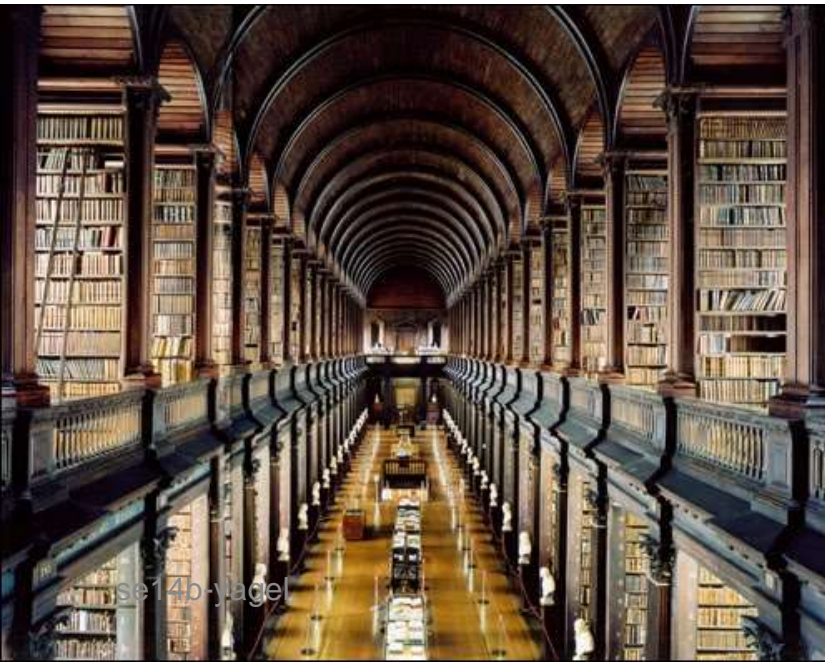
תרגיל עיצוב

- שרטטו ממשק משתמש לחיפוש בספרייה, הכולל את המאפיינים הבאים:

– אפשרות לחיפוש מחרוזת כמחבר, כותר או נושא

– אפשרות לחפש ספר או כתב-עת

– אפשרות לצפות במספר
תוצאות מסודרות ע"פ זמינות
או תאריך הוצאה
(אך לא שתי האפשרויות)



חווית משתמש ואנחנו

- ~~סקר שמישות עם נציג הלקוח במהלך הסבב~~
- ~~שיפורים בסבב הבא (אפשרות להתייחס בסקר עם יש קשר ישיר לקטע קוד)~~
- משוב (מקוון) מהמשתמשים, למשל:
 - A/B Testing
 - [UserVoice](#)
 - Google moderator, idea informer, ideaTorrent, etc...
 - במוצר שלכם?



סיכום - UX

- חווית משתמש
- מי מאפיין? מתי? כיצד בודקים?
- דרישות משתמשים (The User is Drunk)
- עיצוב חווית משתמש כמקצוע
 - ארגונומיה, עיצוב גראפי, קוגניציה, ניסוח טקסט, Gamification
 - מתי באחריות מהנדס התוכנה?
 - לקחים ל: API Usability, או
- Cambridge: Usability of Programming Languages
- MIT: User Interface Design & Implementation קורס: בחירה,



תהליכים וכלים ||

• ראינו: תהליכים, שיטות וכלים (למשל: בדיקות יחידה)

• תהליכים

– שחרור

– תחזוקה

• כלים משלימים:

– כלי פיתוח משלימים

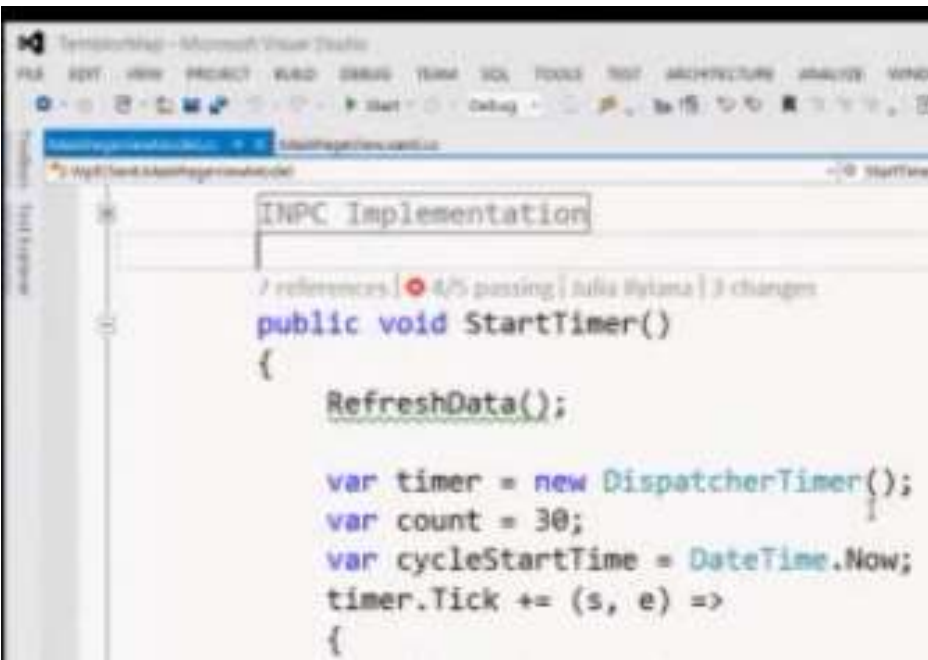
– סקרי קוד

– תיעוד

– בדיקות קבלה

– בדיקות כיסוי

– שילוב מתמשך והפצה מתמשכת



Links - Tools

- Beck&Feitelson, Development and Deployment at Facebook
<https://www.facebook.com/publications/514128035341603>
- G. Weinberger, Egoless Programming
 - אלעד סופר: [כלים אג'ילים – יש דבר כזה?](#)
 - משה קפלן: [סקרי קוד](#), [כלי הפיתוח שאתם פשוט חייבים](#)
- [Continuous Delivery: Anatomy of the Deployment Pipeline](#), Book chapter, 2010
- CI webcast
- Roy Osherov: Code Review [detailed post](#)

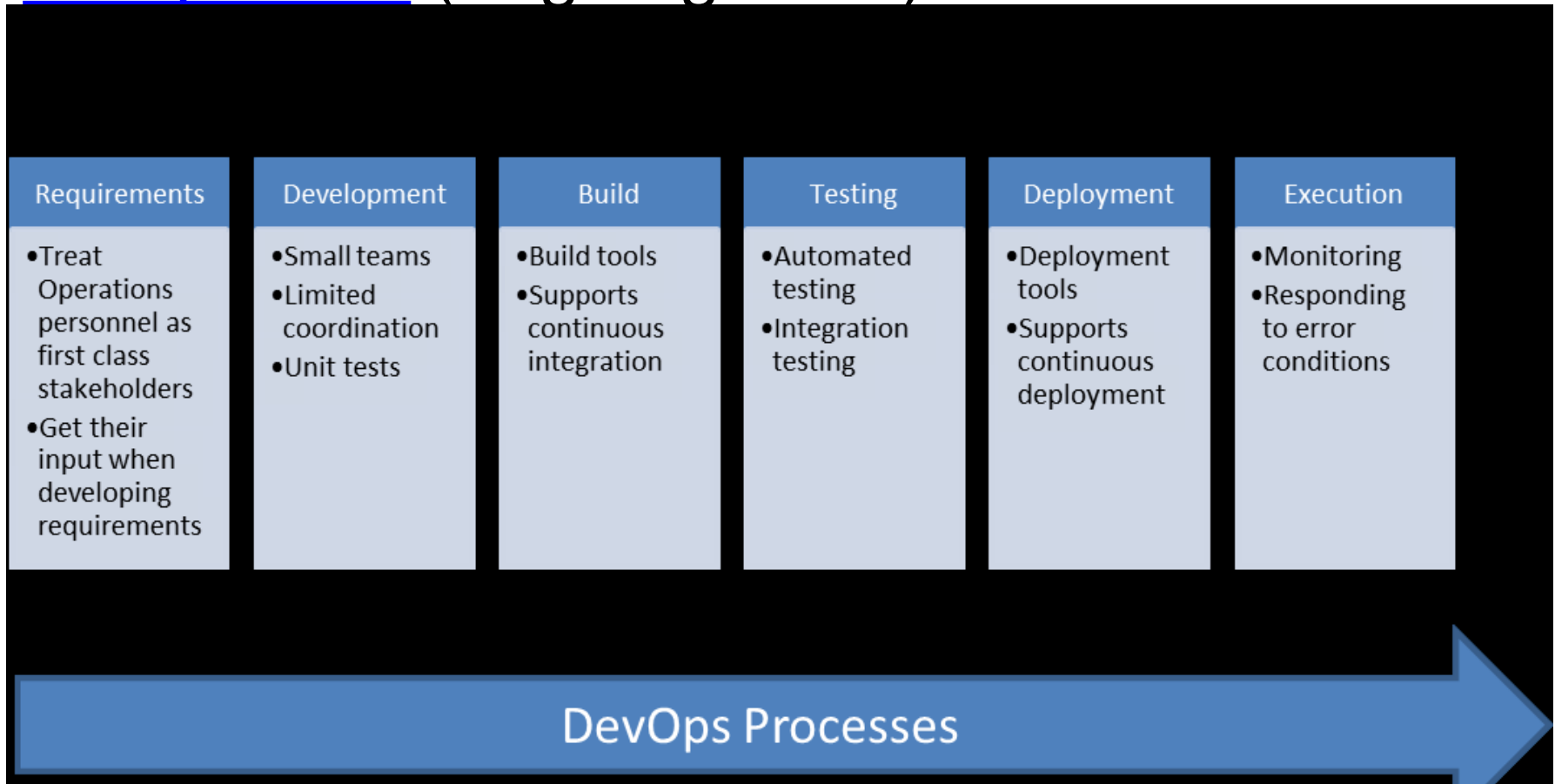
שחרור Release

- E.g., http://en.wikipedia.org/wiki/Deployment_Plan
 - Define and agree release and deployment plans with customers/stakeholders.
 - Ensure that each release package consists of a set of related assets and service components that are compatible with each other.
 - Ensure that integrity of a release package and its constituent components is maintained throughout the transition activities and recorded accurately in the configuration management system.
 - Ensure that all release and deployment packages can be tracked, installed, tested, verified, and/or uninstalled or backed out, if appropriate.
 - Ensure that change is managed during the release and deployment activities.
 - Record and manage deviations, risks, issues related to the new or changed service, and take necessary corrective action.
 - Ensure that there is knowledge transfer to enable the customers and users to optimise their use of the service to support their business activities.
 - Ensure that skills and knowledge are transferred to operations and support staff to enable them to effectively and efficiently deliver, support and maintain the service, according to required warranties and service levels

Agile: DevOps

- [Wiki](#): ..is a software development method that stresses communication, collaboration and integration between software developers and information technology (IT) operations professional
- “..is a new term describing what has also been called ‘**agile** system administration’ or ‘agile operations’ joined together with the values of agile collaboration between development and operations staff. Effectively, you can define DevOps as system administrators participating in an agile development process alongside developers and using a many of the same agile techniques for their systems work.”
<http://theagileadmin.com/what-is-devops/>
- => NoOps

Len Bass, DevOps: A Software Architect's Perspective (ongoing book)



0. כלי פיתוח נוספים

- Typing! (Yegge: non-touch-typists have to make sacrifices in order to sustain their productivity)

1. כלי פיתוח נוספים

- IDE/Compiler
 - Build tools, e.g. make, ant, maven
- Shared Development Env.
 - E.g. vagrant
 - Source control it!
- Static Analysis
 - Compiler warnings
 - E.g., [FindBugz](#) for Java

2. מהו סקר קוד (Code Review)?



- בחינה עקבית של הקוד

- במטרה למצוא באגים ולתקנם $2 < 1 + 1$

- שיפור הקוד ומיומנות המפתח

- האם אתם עושים זאת?

- האם אפשר לא לעשות

זאת?

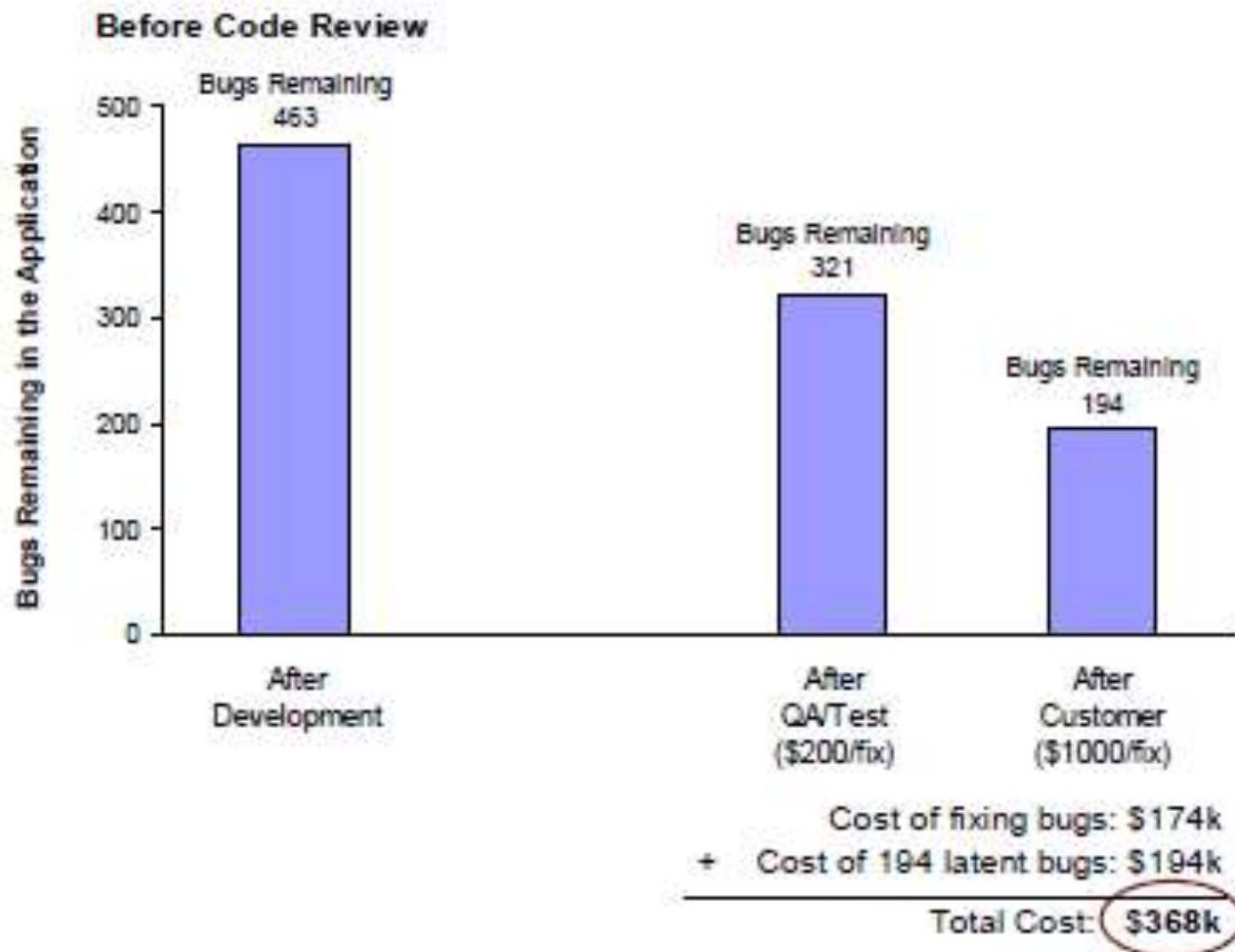
- “Developers at Khan Academy are expected to have all code reviewed”

- Atwood: “peer code reviews are the single biggest thing you can do to improve your code”

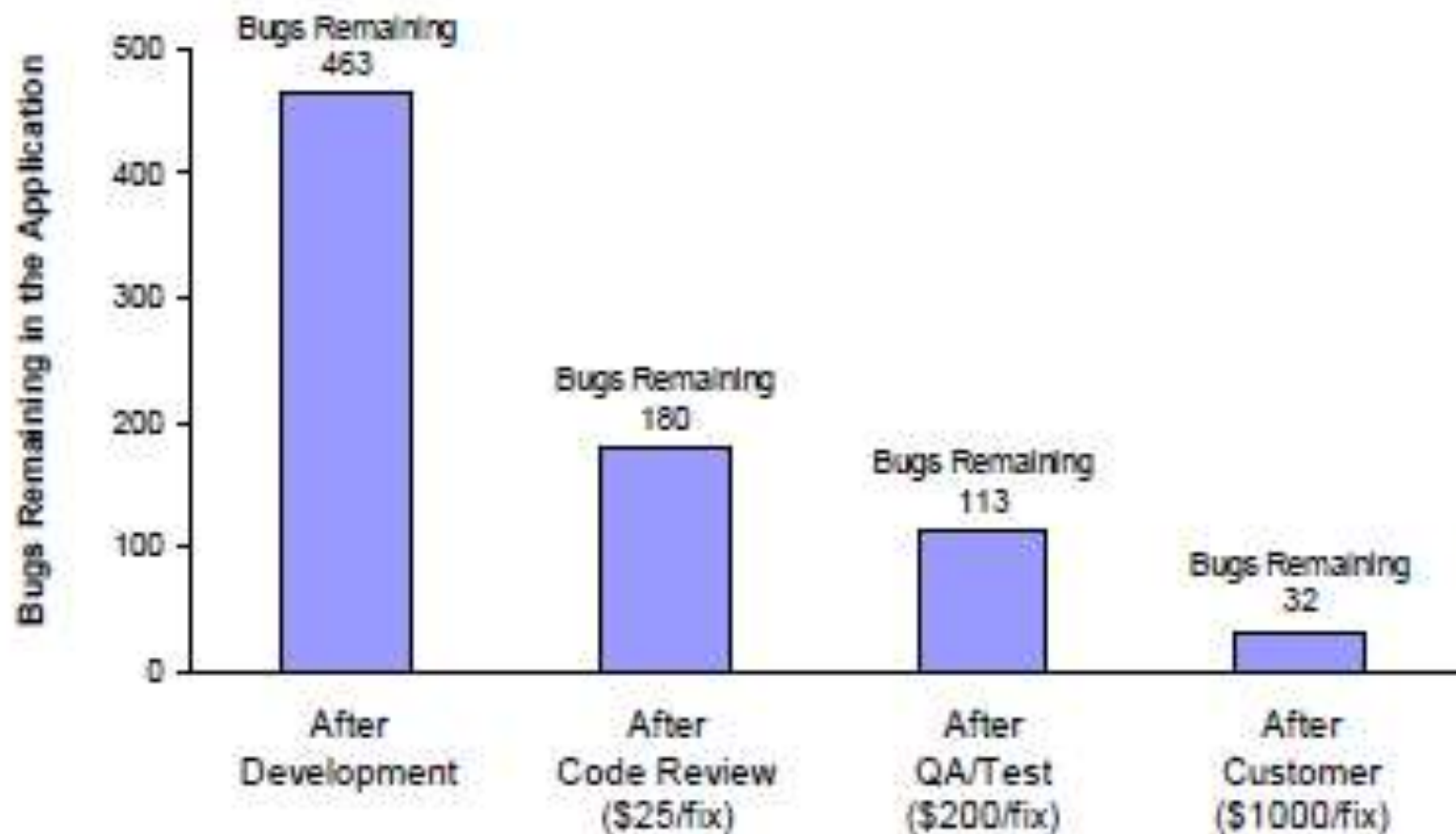
האם כדאי?

- Rigorous inspections can remove 60-90% of errors before the first test is run. (Fagan 1975)
- The first review and hour matter most. (Cohen 2006)

Saving \$150k: A real-world case study



After Code Review



Cost of fixing bugs: \$120k
+ Cost of 32 latent bugs: \$ 32k

Total Cost: **\$152k**

חסרונות?

- זמן מפתחים

– (אבל מצד שני יותר יקר לגלות בהמשך)

- קוד פתוח

– Linus's law: "given enough eyeballs,
all bugs are shallow"

– מי עוד רואה את הקוד?

שיטות לסקר

- מאחורי הכתף
- מייל, IM
- Pair-programming
 - בשילוב עם TDD ו-Refactoring
- כלים ייעודיים
 - [Kiln](#)
 - [Rietveld](#) (דוגמא)
 - [Mondrian: Code Review on the Web](#)
 - <http://www.review-board.org/>
 - <https://code.google.com/p/gerrit/>
 - JCE Course: <http://pair2program.appspot.com/>

GitHub Code Review

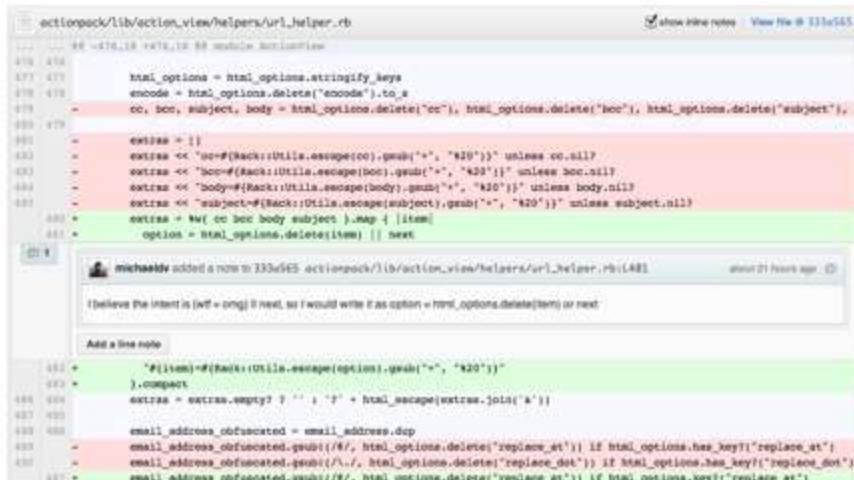
GitHub Pull Request = Code + Issue + •
Code Comments

Pull Requests •

Comments •

Process! •

Open Source Example •



```
476 476
477 477
478 478
479 479
480 480
481 481
482 482
483 483
484 484
485 485
486 486
487 487
488 488
489 489
490 490
491 491
492 492
493 493
494 494
495 495
496 496
497 497
498 498
499 499
500 500
501 501
502 502
503 503
504 504
505 505
506 506
507 507
508 508
509 509
510 510
511 511
512 512
513 513
514 514
515 515
516 516
517 517
518 518
519 519
520 520
521 521
522 522
523 523
524 524
525 525
526 526
527 527
528 528
529 529
530 530
531 531
532 532
533 533
534 534
535 535
536 536
537 537
538 538
539 539
540 540
541 541
542 542
543 543
544 544
545 545
546 546
547 547
548 548
549 549
550 550
551 551
552 552
553 553
554 554
555 555
556 556
557 557
558 558
559 559
560 560
561 561
562 562
563 563
564 564
565 565
566 566
567 567
568 568
569 569
570 570
571 571
572 572
573 573
574 574
575 575
576 576
577 577
578 578
579 579
580 580
581 581
582 582
583 583
584 584
585 585
586 586
587 587
588 588
589 589
590 590
591 591
592 592
593 593
594 594
595 595
596 596
597 597
598 598
599 599
600 600
601 601
602 602
603 603
604 604
605 605
606 606
607 607
608 608
609 609
610 610
611 611
612 612
613 613
614 614
615 615
616 616
617 617
618 618
619 619
620 620
621 621
622 622
623 623
624 624
625 625
626 626
627 627
628 628
629 629
630 630
631 631
632 632
633 633
634 634
635 635
636 636
637 637
638 638
639 639
640 640
641 641
642 642
643 643
644 644
645 645
646 646
647 647
648 648
649 649
650 650
651 651
652 652
653 653
654 654
655 655
656 656
657 657
658 658
659 659
660 660
661 661
662 662
663 663
664 664
665 665
666 666
667 667
668 668
669 669
670 670
671 671
672 672
673 673
674 674
675 675
676 676
677 677
678 678
679 679
680 680
681 681
682 682
683 683
684 684
685 685
686 686
687 687
688 688
689 689
690 690
691 691
692 692
693 693
694 694
695 695
696 696
697 697
698 698
699 699
700 700
701 701
702 702
703 703
704 704
705 705
706 706
707 707
708 708
709 709
710 710
711 711
712 712
713 713
714 714
715 715
716 716
717 717
718 718
719 719
720 720
721 721
722 722
723 723
724 724
725 725
726 726
727 727
728 728
729 729
730 730
731 731
732 732
733 733
734 734
735 735
736 736
737 737
738 738
739 739
740 740
741 741
742 742
743 743
744 744
745 745
746 746
747 747
748 748
749 749
750 750
751 751
752 752
753 753
754 754
755 755
756 756
757 757
758 758
759 759
760 760
761 761
762 762
763 763
764 764
765 765
766 766
767 767
768 768
769 769
770 770
771 771
772 772
773 773
774 774
775 775
776 776
777 777
778 778
779 779
780 780
781 781
782 782
783 783
784 784
785 785
786 786
787 787
788 788
789 789
790 790
791 791
792 792
793 793
794 794
795 795
796 796
797 797
798 798
799 799
800 800
801 801
802 802
803 803
804 804
805 805
806 806
807 807
808 808
809 809
810 810
811 811
812 812
813 813
814 814
815 815
816 816
817 817
818 818
819 819
820 820
821 821
822 822
823 823
824 824
825 825
826 826
827 827
828 828
829 829
830 830
831 831
832 832
833 833
834 834
835 835
836 836
837 837
838 838
839 839
840 840
841 841
842 842
843 843
844 844
845 845
846 846
847 847
848 848
849 849
850 850
851 851
852 852
853 853
854 854
855 855
856 856
857 857
858 858
859 859
860 860
861 861
862 862
863 863
864 864
865 865
866 866
867 867
868 868
869 869
870 870
871 871
872 872
873 873
874 874
875 875
876 876
877 877
878 878
879 879
880 880
881 881
882 882
883 883
884 884
885 885
886 886
887 887
888 888
889 889
890 890
891 891
892 892
893 893
894 894
895 895
896 896
897 897
898 898
899 899
900 900
901 901
902 902
903 903
904 904
905 905
906 906
907 907
908 908
909 909
910 910
911 911
912 912
913 913
914 914
915 915
916 916
917 917
918 918
919 919
920 920
921 921
922 922
923 923
924 924
925 925
926 926
927 927
928 928
929 929
930 930
931 931
932 932
933 933
934 934
935 935
936 936
937 937
938 938
939 939
940 940
941 941
942 942
943 943
944 944
945 945
946 946
947 947
948 948
949 949
950 950
951 951
952 952
953 953
954 954
955 955
956 956
957 957
958 958
959 959
960 960
961 961
962 962
963 963
964 964
965 965
966 966
967 967
968 968
969 969
970 970
971 971
972 972
973 973
974 974
975 975
976 976
977 977
978 978
979 979
980 980
981 981
982 982
983 983
984 984
985 985
986 986
987 987
988 988
989 989
990 990
991 991
992 992
993 993
994 994
995 995
996 996
997 997
998 998
999 999
1000 1000
```

הנשק הסודי?

- של Extreme Programming – [Pairing vs. Code Review](#) דיון
- של חברות שמצליחות להחזיק מוצרים לאורך זמן
- בפרויקט שלכם: תיעוד שליחת diff והערות שהתקבלו



Richie Rump
@Joriss



RT @ashalynd If the programmers like each other, they play a game called "pair programming". And if not, then the game is called "peer review"

9:13 PM - 4 May 2013

3. תיעוד ומסמכים

```
int five = 7; //11
```

- למה\האם לכתוב תיעוד? מי? מתי? סוגים?
- הערות: שנוי במחלוקת, למשל
<http://apdevblog.com/comments-in-code/>
Stackoverflow [thread](#), [Do not comment code](#) 😊
- כלים ליצירת תיעוד מהערות
[Javadoc](#)/Ndoc, Doxygen, Sandcastle, [docco](#), [ועוד](#)
– למשל [String in Java](#)
- מתי כלים אלו נצרכים?
- איזה סוג תיעוד נותנות בדיקות יחידה? (CSVReader [דוגמא](#))
- דיון: [Agile Documentation](#), Amber,

4. כלים ותהליכים תומכי בדיקות

- Unit Testing Frameworks

 - ✓ xUnit

 - Parameterized tests, White-box, fuzzing (security, [Heartbleed bug](#)), GUI

- Acceptance Tests Frameworks

- Code Coverage

- Continuous Integration

- A/B Testing ([paper](#))

```
בכמה מקרים תתרחש שגיאה?:  
int foo(int x) { // x is an input  
    int y = x + 3;  
    if (y == 13) abort(); // error  
    return 0;  
}
```

בדיקות קבלה (הזכרנו)

- שמות שונים: בדיקות משתמש, בדיקות קצה, פונקציונליות, מפרטים מורצים, BDD, **A**TDD
- סביבות מתפתחות לאחרונה, למשל
Fit, [Cucumber](#), SpecFlow, [BDDify](#), [TextTest](#) –
Web Automation: Selenium, Watir –
– <http://swarm.jquery.org/> סטטוס מעודכן לדוג'

כיסוי קוד ע"י בדיקות

- כמה בדיקות צריך לכתוב?
- ב-TDD לא כותבים אף שורה בלי בדיקה
 ⇐ 100% כיסוי
 ⇐ אם יש 90% כיסוי, היכן מסתבר שרוב הבאגים?
- כלים לדוגמא:
 - .Net : Ncover (משולב ב- TestDriven.Net)
 - Eclipse : [EclEmma](#)
 - ~~נבדוק עוד דוגמא [CsvReader](#)~~



5. שילוב מתמשך – CI/CD

- מתי לבנות הכל ולהריץ את הבדיקות?
- באופן אידיאלי: לפני ואחרי כל שינוי
- לפחות כל פעם שמכנסים משהו חדש לבקרת הגרסאות?
 - למה זה חשוב
- מי מריץ?
- שרתים: למשל Jenkins, Cruise Control, TeamCity ([on ec2](#))
- ענן: Heroku, Cloudbee, MS Azure, jelastic
- שרות: [travis-ci](#), [drone.io](#), [codebetter](#), [codeship.io](#)
- כלי הידור לדוגמא: Make, MsBuild, Ant, Kayak
- <http://martinfowler.com/articles/continuousIntegration.html>
- <http://martinfowler.com/bliki/ContinuousDelivery.html>

Continuous integration - Recommended practices

http://en.wikipedia.org/wiki/Continuous_integration#Recommended_practices

- Maintain a code repository
- Automate the build
- Make the build self-testing
- Everyone commits every day
- Every commit (to mainline) should be built
- Keep the build fast
- Test in a clone of the production environment
- Make it easy to get the latest deliverables
- Everyone can see the results of the latest build
- Automate deployment

יתרונות וחסרונות

- איתור באגים קל ומהיר
- לא נופלים על בעיות שילובים בסוף
- עותק זמין ועובד של המוצר כל הזמן
- משוב תמידי על איכות הקוד
- מעודד כתיבת קוד מודולרי ופחות תלותי
- מצריך הקמת תשתיות בהתחלה
- כדי להיות אפקטיבי מצריך פיתוח סט בדיקות

בפרויקט - בונים

• Jenkins

– הקמת שרת (צוות הקורס\סיסטם)

– הוספת פרויקט והגדרות (צוות)

- בקרת גרסאות

- הידור

- בדיקות

- הודעות

– (Tray Application)

- תוספים רבים (כולל משחק)

- שרותי: Travis-ci, CircleCI

- הצגה בשבוע הבא



הפצה מתמשכת - CD

- פרויקט תוכנה כפס ייצור
- לאילו פרויקטים מתאים? האם עדיין יש צורך באיטרציות? (דוגמא [github](https://github.com))
- תשתית CI כשלב מקדים
- חמשת השלבים למימוש Continuous Deployment
- CI/CD/cloud Demo - "Bootstrapping an agile project with continuous deployment using cloudbees"

לסיכום - כלים



- צידה לדרך
 - סקרי קוד, תיעוד
 - בדיקות מעבר ליחידה, כיסוי
 - שילוב מתמשך
- כלים רבים נוספים, למשל
 - אכיפת סגנון FxCop
- מהו המינון הנכון? ...



בפעם הבאה

- סיכום הקורס

- [סקב](#) קורס, פרויקט, שיטות, תהליכים וכלים

- מבנה הבחינה

- מצגות סיום

- הצגה צוותית: מוצר + רטרוספקטיבה

- כל אחד מספר על חלקו (משימה אישית 5) ויכול להישאל על כך

- בונוס CI