BERNHARD VON GUNTEN

GENERATIONS

AGENDA

- About me
- Software Development Evolved
- The Java Problem
- Training of Employees & Colleagues
- Q & A

BERNHARD VON GUNTEN

- Age 43, Coding since 13
- Working with Java since 1998
- Java Software Developer / Technical Lead (Swisslog)
- Slight Trainer Role

SOFTWARE DEVELOPMENT EVOLVED OVER THE DECADES – 1

- One man armies became dev teams and software departments
- A couple of programs became a whole IT landscape
- Developers had to accept a subordinate role to the new world

SOFTWARE DEVELOPMENT EVOLVED OVER THE DECADES – 2

- The programming Languages evolved too
- Cobol became C++
- C++ was too complex and became Java
- Java became also complex (but for different reason)
- Developers again, had to adapt to new languages and their tools.

SOFTWARE DEVELOPMENT EVOLVED OVER THE DECADES – 3

- Astonishingly, we're facing a challenge now. We have …

GENERATIONS OF SOFTWARE & PROGRAMMING LANGUAGES

(but still the same people)

THE JAVA PROBLEM

- Not only software development changed, but also Java and our understanding of what is the best way to craft software

THE JAVA PROBLEM

- We learned Java ...
- First, we did Applets
- We did Web UIs with Servlets & JSP
- We did UI Applications with AWT
- We did Batch Jobs with plain Java
- We did Webservices with Axis
- We did EJBs on all fours

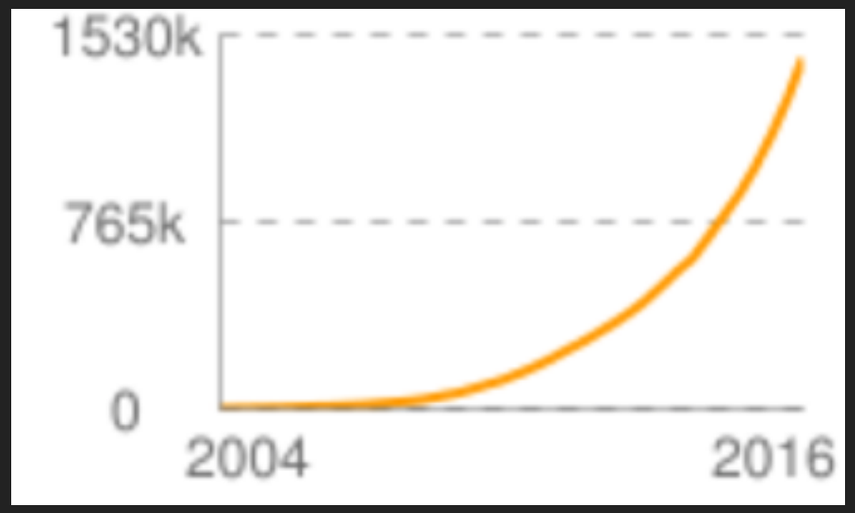
THE JAVA PROBLEM

And then we did it all over again, but with a better framework (yay !)

- We learned new Java Language features
- We threw applets away
- We did Web UIs with STRUTS or Cocoon or any other framework
- We did UI Applications with Swing and FX
- We did Batch Jobs with Spring
- We did Webservices with JEE
- We did EJBs with CDI

THE JAVA PROBLEM

- We've seen an endless list of frameworks, libraries & tools come *AND* go
- The amount of hosted artifacts @ mvnrepository.com is 1500k+ !



- We must be the worst engineers in the history of mankind!!
  (But at least, we do share ...)

THE JAVA PROBLEM

- Back in 1860 Christian Reithmann invented the four-stroke engine.

150 Years later, the design is still used …

THE JAVA PROBLEM

- Back in 2000 we wanted traceability in Java applications

15 Years later, we still don’t know how to do this …

THE JAVA PROBLEM

- The Dark Art of Logging in Java:
- Version 1: System.out.println()
- Version 2: MyLogSingleton().getInstance().writeLog()
- Version 3: Log4j
  - log4j.properties (but where to place it? Best in the JRE!)
  - info, debug, or warn ? (Let’s better write some guidelines!)

THE JAVA PROBLEM

- Version 3.1
  - We migrate to log4j.xml (Still no clue, where to place the damn file)
  - But we have appenders to write informations to space!
- Version 4: Logging is part of the JDK
  - It is so bad, it hurts. But let’s use it anyways
- Version 5: SLF4J on application servers
  - Now we can’t find the crucial information anymore. Let’s log manually into a second file!

THE JAVA PROBLEM

- Yes, we saw A LOT of frameworks and »best practices« come and go
- People and companies invested a lot of time and money to adapt
- It is not a problem as long as developers are well trained
- The situation is not *that* bad! Really, Java is still your best choice! But …

THE JAVA PROBLEM

- For a lot of people, the tools are overwhelming too
  - The IDE is a monster
  - Maven is a mystery
  - Application Servers are frightening
  - And Sonar is the devil

THE JAVA PROBLEM AFFECTS MOST EMPLOYEES

- It affects the older developers the most, and they are not comfortable with moving to, or working with Java
- It is very hard to train these developers to be as productive in the Java world than where they come from
- They know everything about the business and software in place, so they are irreplaceable. End of story

TRAINING OF EMPLOYEES

- For the next couple of slides, let’s assume that all developers in scope are eager to join and master the Java world

HOW CAN MANAGERS HELP TO WORK THE JAVA PROBLEM

- Know the strengths and weaknesses of your developers from a technical point of view
- Create assessments using detailed tests
- Based on the tests, train your developers

MANAGER: JAVA COURSES

- Do not rely on the "Java for Developers" courses
- Get an internal or external professional that will do trainings in house
- Create a course agenda based on knowledge and needs based on earlier assessments
- Do regular course reruns

MANAGER: CONFERENCES

- The "Return of Investment" is huge
  - If the participant matches the level of talks
    - Accompanied by experienced colleagues
    - Set goals to the participants (e.g. prototyping, demos, info sessions afterwards)
- Broadens the mind of a developer, also for trends and the future

MANAGER: METRICS FOR CLEAN(ER) CODE

- The amount of sonar issues does not measure the quality of code, but …
  - It is a number and an understandable target
  - It does help developers to write cleaner code, and speeds up later code reviews
- The same goes for »code coverage«

HOW CAN YOUNG JAVA CRACKS HELP THEIR COLLEAGUES

- First and most important: PAY THEM RESPECT !
- Those people (ok, their predecessors) flew and landed on the moon with nothing but a small calculator and less than 5000 lines of code! (Imagine doing this with JEE)
- Those people know more about hardware and cpu close programming than you will ever do
- Their software already runs the business
- You will learn from them too, trust me on this ;)
- If you’re a good coder, they will respect you too !

DEVELOPERS: FOR THE LOVE OF GOD, SHARE YOUR KNOWLEDGE !!!

- BUT, don’t you dare to be Adam Bien!! (Nobody writes working applications within 10 minutes, ever)
- If you don’t do pair programming, still invite your colleagues to join in when you work or change shared code
- Imagine that your colleagues don’t know that books like »Clean coder« do exist
- Listen to your colleagues when they explain why they did stuff a certain way in the past, talk pros and cons of todays approaches in your work

CODE REVIEWS

- Code reviews are *THE* essential tool to improve the know-how and code of any developer
- Look at the code together, discuss different solutions. Make sure all agree or at least understand proposed changes
- Code review comments are no metrics, and therefore not meant to be compared or refactored in every case

CODING EXERCISES / DOJOS – 1

- Mix theory and practice
- It’s first about coding, second about the language
- Select a basic (!) topic and speak about it
- Use the consolidated knowledge in one or more exercises
- Demonstrate and talk about the written code

CODING EXERCISES / DOJOS – 2

- Variation: Eliminate IDEs, use eg. cyberdojo.com
- Variation: Mix computer languages during exercises to talk about pros and cons

END OF TALK

- Q & A
- Thanks ;)

Bernhard von Gunten

bvg@nostromo.ch

www.nostromo.ch

@BigBadBaerni