Supporting girls who code

http://bit.ly/saponlinetrack-fundraiser

[SOT202]
Refactoring for Dummies

Laurens van Rijn

May 30th, 2020

13:20 - 14:00 UTC

What you always wanted to know about refactoring but were afraid to ask...

Refactoring for Dummies

Today's topics

Agenda

- Defining refactoring…
- Why and when should you refactor?
- Trouble in paradise?
- Do I tell my manager? 
- How will I know?
- Transforming your code
- Safety first
- Want to see and know more?
- Q&A

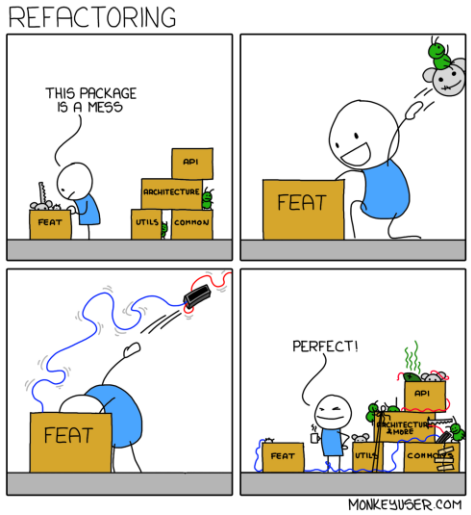
Refactoring for Dummies

Defining refactoring…

"Making changes to the internal structure of software to make it easier to understand and cheaper to modify without changing its observable behaviour."
- Martin Fowler

Please note

- Both a noun and a verb
- Two hats
- Refactoring NE performance tuning



Refactoring for Dummies

Why and when should you refactor?

Why?

- Improving the design of the software, making software easier to understand, easier to find bugs, develop faster, professional pride

When

- Opportunistic: preparatory refactoring (prefactoring), comprehension
- Planned: long-term, code review

Why not?

- The code is not being used nor expected to be used
- No modifications needed
- Dilemma: rewrite vs refactor

Refactoring for Dummies

Trouble in paradise?

Complications

- “Slowing down new features”
- Code ownership
- Branches
- Testing
- Legacy code
- Databases

Refactoring for Dummies

Do I tell my manager? 😊

It depends…

- There’s no need to
- Convince your manager
- Just don't

Refactoring for Dummies

How will I know?

Hints

- Code smells
- Code stench

Unclean code ➔

Refactoring for Dummies

Code Smells… (refactored)

Refactoring for Dummies

Transforming your code

Possible solutions for code smells

- Duplicate code
- Comments
- Shotgun surgery
- Mysterious name
- Large class

Available tools?

- SAPGUI: CCAPPS, SE24 Refactoring assistant
- Eclipse ADT, see [SOT307] Refactoring in ADT by Johann Fößleitner

Refactoring for Dummies

Safety first

How can you refactor safely?

- Save the code you start with
- Keep refactorings small, one at a time
- Build a parking lot
- Make frequent checkpoints
- Retest (but also add new test cases)
- Review all changes
- Adjust your approach depending on the risk level of the refactoring
- Consider pair programming

Refactoring for Dummies

Want to see more?

SAP Online Track:

- [SOT307] Refactoring in ADT by Johann Fößleitner, May 30, 17:45 - 18:05 UTC (19:45 – 20:05 CET)

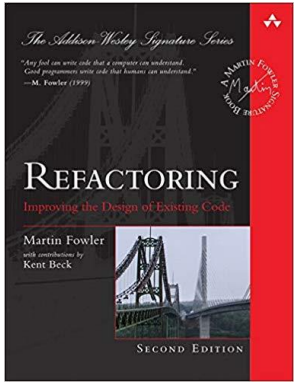
  I will demonstrate how you can refactor your ABAP code with the ABAP Development tools (Eclipse)…

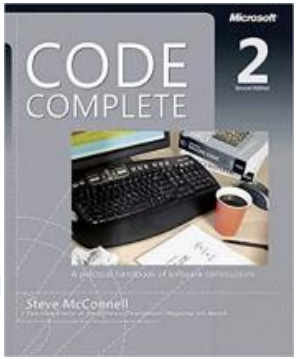
- [SOT112] Poltergeist! Transforming Legacy ABAP Code by Paul Hardy, May 31, 00:00 - 01:00 UTC (02:00 – 03:00 CET)

  For the last year and a half I have been transforming a gigantic monolithic procedural DYNPRO program into an OO masterpiece. Everybody wants to do that sort of thing but it is deemed far too dangerous…

Refactoring for Dummies

Want to know more?





Websites:

- https://blogs.sap.com/2019/09/02/refactoring-an-abap-include-monolith/
- https://blogs.sap.com/2006/03/04/when-should-we-refactor/
- https://www.martinfowler.com/bliki/DesignStaminaHypothesis.html
- https://en.wikipedia.org/wiki/Code_refactoring

Books:

- Refactoring: Improving the Design of Existing Code 2nd edition
- Code Complete (Developer Best Practices) 2nd edition

Refactoring for Dummies

Q&A

Thank you

Contact information

Laurens van Rijn
Technical ES Consultant

Ordina
Ringwade 1
3439 LM Nieuwegein
The Netherlands

✉ laurens.van.rijn@ordina.nl
@laurensvanrijn