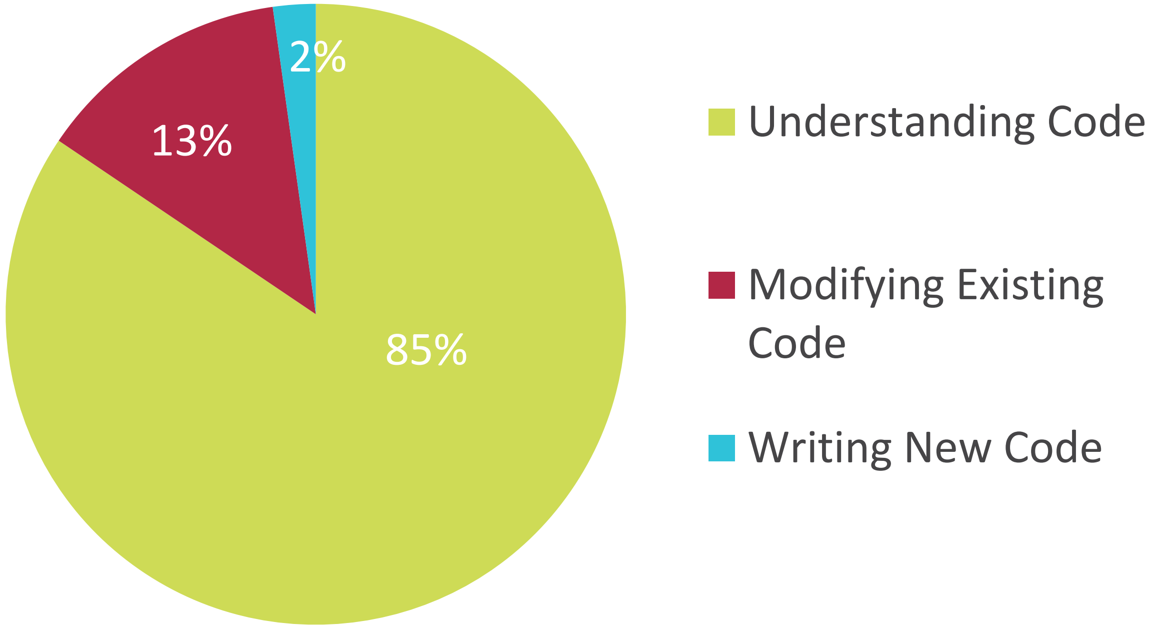
Taking care of clean code

Clean Code training.

TRAINING

CENTER

Developer’s effort



Why Clean Code matters

1. READ FAST
2. CHEAP CHANGE
3. LESS EXPENSIVE
4. LESS COMPLEXITY
5. LESS BUGS
6. PROFESSIONAL

Keeping the code quality

1 TEAM COMMITMENT

2 REGULAR ACTIVITY

3 STANDARDS

4 PRACTICES

5 TOOLS

Code convention

Clean Code training.

Naming variables

Clean Code training.

Naming variables

USE

- Follow code convention
- Use comprehensive names
- 1 and 0 harcoded
- Java convention:
    - `i` and `j` allowed for loops
    - `ALL_CAPS` – constant
    - `ClassFromCapital`
    - `methodOrVariable`

AVOID

- Magic numbers
- Hungarian notation
- Abbreviations
- x, y, z or any one-letter apart i/j
- file2, sum1
- Unclear “flag”, “result”, “status”

Naming classes

Clean Code training.

Naming classes

USE

- Noun
- Use comprehensive names

AVOID

- God class/ Magic Hammer
- Data storage
- Abbreviations
- Unclear “Instance”, “Entity”
- Numbers like “Test2”
- “MyClass”

Naming functions

Clean Code training.

Naming functions

USE

- Verb
- Use long names

AVOID

- Several actions in one method
- Unclear “check…”, “verify”, “try…”
- void get(), String print() etc.
- Lot of parameters

Comments

Clean Code training.

Comments

USE

- Legal (license)
- “Javadoc” for libraries
- Intention for solution made

AVOID

- /* */
- // TODO
- Obvious statement
- // for,if,try,while close
- Commented code
- “All methods should have comment”

Excess code

Clean Code training.

Excess code

USE

- Direct return
- Ternary

AVOID

- System.out.println()
- Comparing Boolean with boolean
- Variable just for return

Exceptions

Clean Code training.

Exceptions

USE

- Use them

AVOID

- Empty catch
- Throwing default exceptions like “Exception”

<epam>

Tests

Clean Code training.

TRAINING
CENTER
<epam>

Tests

USE

- Must have assert
- One test – one assert
- Before/After
- Comprehensive exception name

AVOID

- Test data in test name
- loops/conditional statements/lambdas
- Strict tests order
- Tests dependency

Design principles

Clean Code training.

Design principles

USE

- KISS
- DRY

AVOID

- Blind copy-pasting
- Complexity
- Write in main, then split into classes
- “I’ll refactor it later”

Tools

Clean Code training.

TRAINING
CENTER

Tools

USE

- Code Convention
- Code Review
- Static Code Analyzers

Persona

Clean Code training.

Persona

USE

- Be Engineer not Coder
- Read professional books/blogs
- Talk with mature developers
- Be Honest
- Be Disciplined

Check list & summary

Clean Code training.

Summary

- Engineer reads more code than writes
- Naming is extremely important for variables, classes, functions
- Use comments only were it’s necessary
- Avoid complexity in code and tests
- KISS and DRY
- Throw custom Exceptions
- Tools like static code analyzers and code coverage trackers can help may help
- Be Engineer not Coder
- Read books

Extras

Version Control with Git. DevTestOps training.

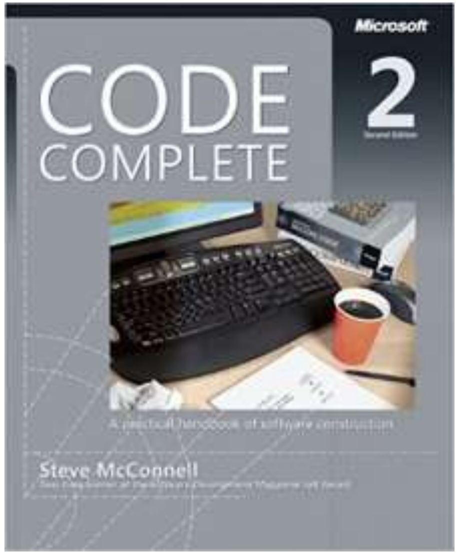
Extras

EXTRAS

- SOLID
- Law of Demeter
- Multi-threading
- Refactoring
- IoC
- Design Patterns
- Anti-patterns

READ MORE

- **Code Complete** by Steve McConnell
- **Clean Code** Robert Martin



Robert C. Martin Series
PRENTICE HALL
Clean Code
A Handbook of Agile Software Craftsmanship
Foreword by James O. Coplien
Robert C. Martin