Lesson 9

clean code

© 2020 ICT Intelligence

Coding is for humans

Programming is the art of telling another human what one wants the computer to do.

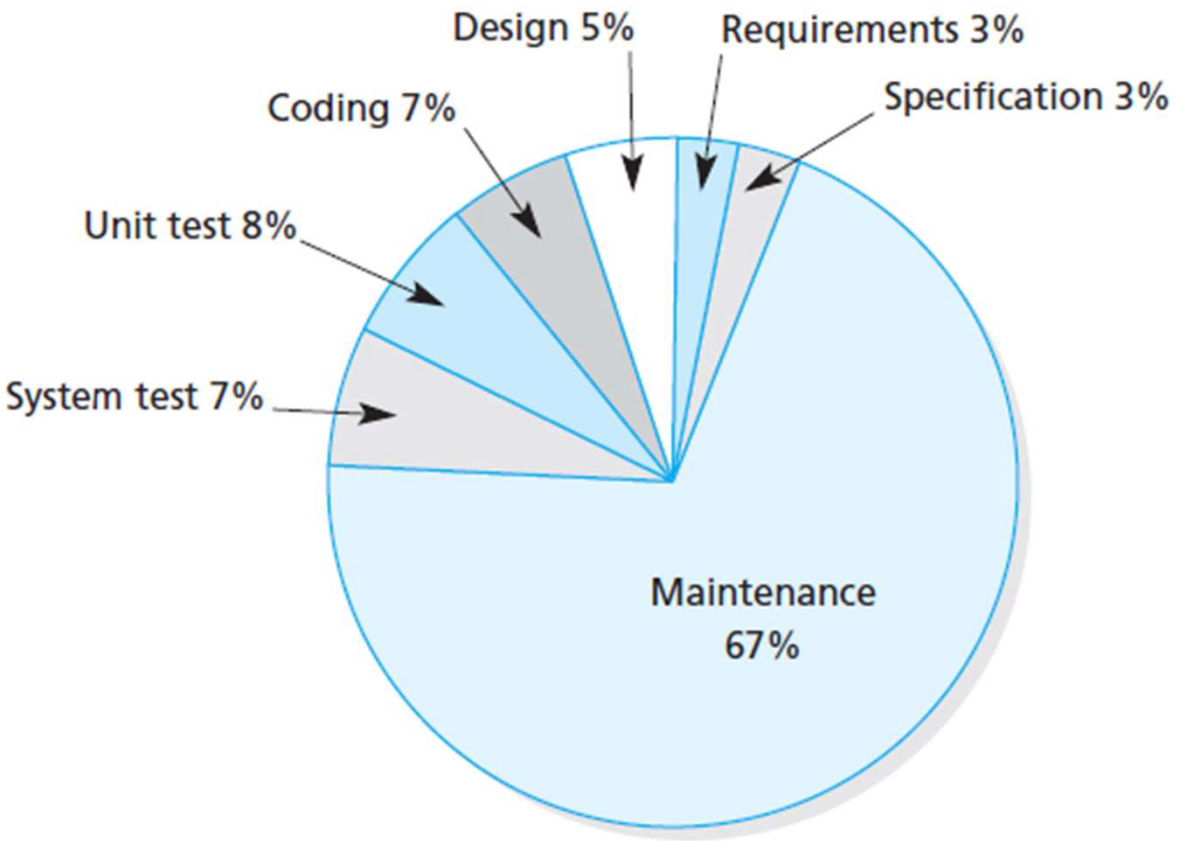
Donald Knuth

Any fool can write code that a computer can understand. Good programmers write code that humans can understand.

Martin Fowler

© 2020 ICT Intelligence

Cost of software development



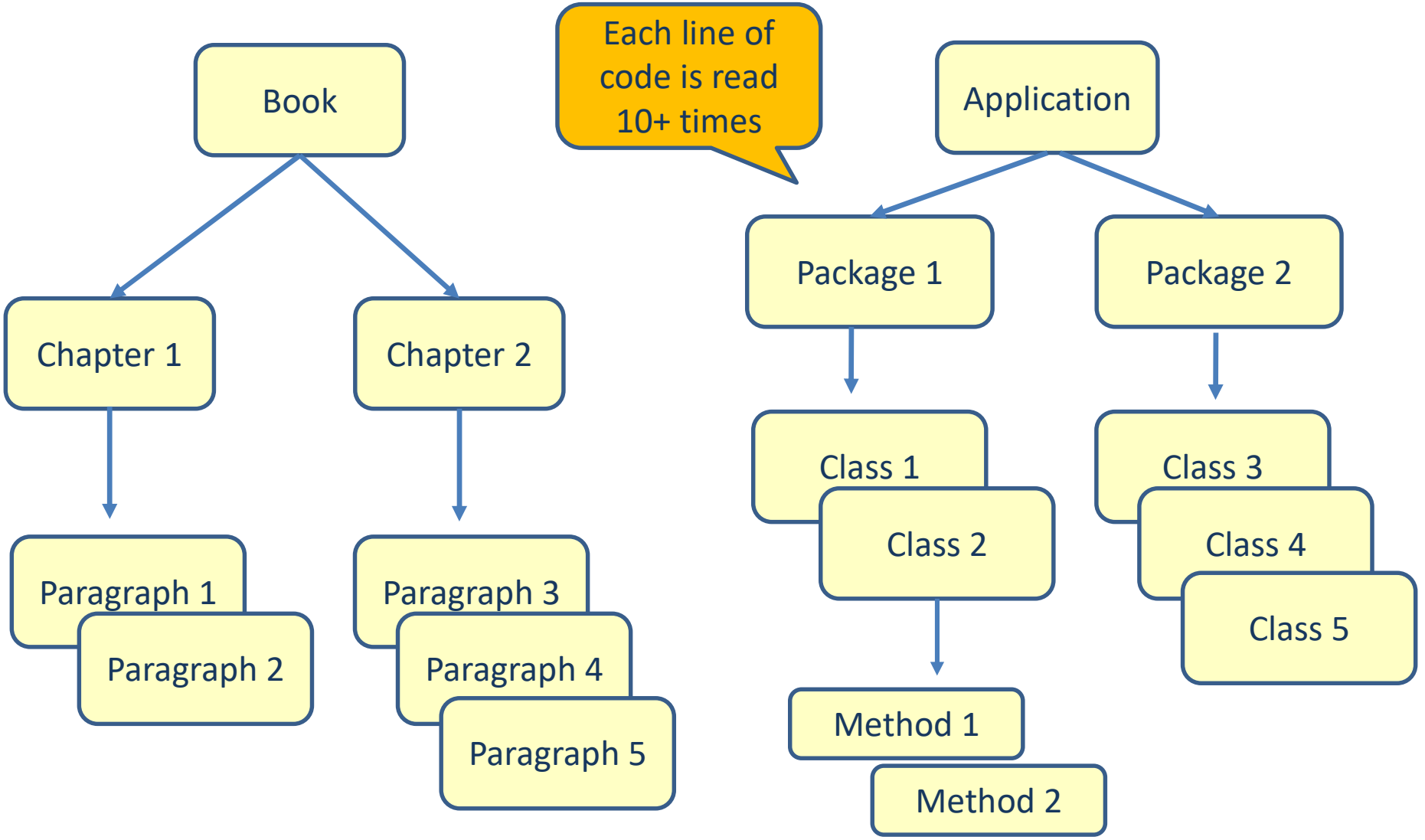
© 2020 ICT Intelligence

Software craftsmanship

- A good craftsperson takes the time and effort to make something of quality which is worthwhile to have and which will still function way after the crafted item has been made.

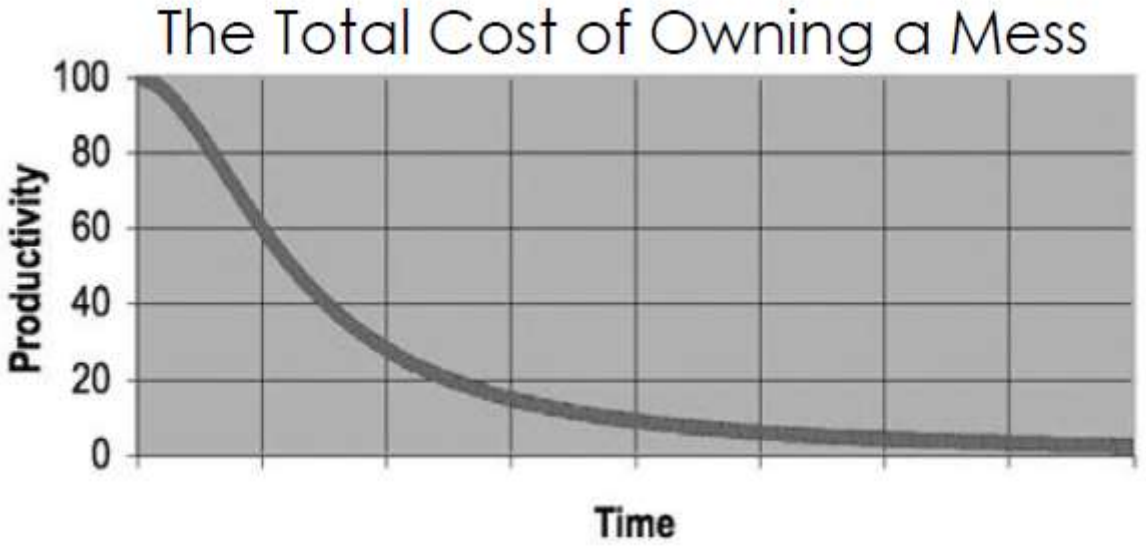
© 2018 ICT Intelligence

You are an author!



© 2020 ICT Intelligence

Why clean code



"Any one can write code that a computer can understand, the trick is to write code that humans can understand."

© 2020 ICT Intelligence

Boy scout rule

- Leave the campground cleaner than you found it

© 2020 ICT Intelligence

Ugly code

```
public class Inventory {
    List<Product> sl = new ArrayList<>();

    boolean isInStock(String n) {
        Product s = new Product(n);
        int k = 0;
        int h = sl.size() - 1;

        while (k <= h) {
            int m = k + (h - k) / 2;
            int c = sl.get(m).compareTo(s);

            if (c < 0) {
                k = m + 1;
            } else if (c > 0) {
                h = m - 1;
            } else {
                return true;
            }
        }
        return false;
    }
}
```

NOT OK

© 2020 ICT Intelligence

Clean code

```
public class Inventory {
    List<Product> sortedList = new ArrayList<>();

    boolean isInStock(String name) {
        Product supply = new Product(name);
        int low = 0;
        int high = sortedList.size() - 1;

        while (low <= high) {
            int middle = low + (high - low) / 2;
            int comparison = sortedList.get(middle).compareTo(supply);

            if (comparison < 0) {
                low = middle + 1;
            } else if (comparison > 0) {
                high = middle - 1;
            } else {
                return true;
            }
        }

        return false;
    }
}
```

OK

© 2020 ICT Intelligence

MEANINGFUL NAMES

© 2020 ICT Intelligence

Use Intention Revealing Names

- Choosing good names takes time but saves more than it takes
- If a name requires a comment, then the name does not reveal its intent.

© 2020 ICT Intelligence

Use Intention Revealing Names

NOT OK

```
public List<int[]> getThem() {
  List<int[]> list1 = new ArrayList<int[]>();
  for (int[] x : theList) {
    if (x[0] == 4)
      list1.add(x);
  }
return list1;
}
```

What is the significance of the zeroth item in the list?

What kind of things are in theList?

What is the significance of the value 4?

What does this method do?

© 2020 ICT Intelligence

Use Intention Revealing Names

```
public List<int[]> getThem() {
  List<int[]> list1 = new ArrayList<int[]>();
  for (int[] x : theList) {
    if (x[0] == 4)
      list1.add(x);
  }
return list1;
}
```

NOT OK

```
public List<Cell> getFlaggedCells() {
  List<Cell> flaggedCells = new ArrayList<Cell>();
  for (Cell cell : gameBoard) {
    if (cell.isFlagged())
      flaggedCells.add(cell);
  }
return flaggedCells;
}
```

OK

© 2020 ICT Intelligence

Avoid abrevations

```
int d; // elapsed time in days
int ds;
int dsm;
int faid;
```

NOT OK

```
int elapsedTimeInDays;
int daysSinceCreation;
int daysSinceModification;
int fileAgeInDays;
```

OK

© 2020 ICT Intelligence

Avoid disinformation

- Avoid leaving false clues that obscure the meaning of code.
  - Do not refer to a grouping of accounts as an ***accountList*** unless it's actually a List.
    - The word **list** means something specific to programmers.
    - Plain ***accounts*** would be better.
- Beware of using names which vary in small ways.
  - XYZControllerForEfficientHandlingOfStrings
  - XYZControllerForEfficientStorageOfStrings

© 2020 ICT Intelligence

Avoid disinformation

```
Customer[] customerList;
Table theTable;
```

NOT OK

```
Customer[] customers;
Table customers;
```

OK

© 2020 ICT Intelligence

Avoid disinformation

Avoid variable names like lowercase L or uppercase O

```
int a = l;
if (O == 1)
  a = O1;
else
  l = O1;
```

NOT OK

© 2020 ICT Intelligence

Make meaningful distinctions

```
public static void copyChars(char a1[], char a2[]) {
  for (int i = 0; i < a1.length; i++) {
    a2[i] = a1[i];
  }
}
```

NOT OK

```
public static void copyChars(char source[], char destination[]) {
  for (int i = 0; i < source.length; i++) {
    destination[i] = source[i];
  }
}
```

OK

© 2020 ICT Intelligence

Use Pronounceable Names

- If you can't pronounce it, you can't discuss it without sounding like an idiot.
- This matters because programming is a social activity.

```
class DtaRcrd102 {
  private Date genymdhms;
  private Date modymdhms;
  private string s1 = "102";
  /*.....*/
}
```

NOT OK

```
class Customer {
  private Date generationTimestamp;
  private Date modificationTimestamp;
  private string recordId = "102";
  /*.....*/
}
```

OK

Intelligent conversation is now possible: *"Hey, Mikey, take a look at this record! The generation timestamp is set to tomorrow's date! How can that be?"*

© 2020 ICT Intelligence

Use searchable names

- Single-letter names and numeric constants have a particular problem in that they are not easy to locate across a body of text.
  - `MAX_CLASSES_PER_STUDENT` vs. `7`
- `e` is a poor choice for any variable

© 2020 ICT Intelligence

Use searchable names

```
for (int j = 0; j < 34; j++) {
  s += (t[j] * 4) / 5;
}
```

NOT OK

```
int realDaysPerIdealDay = 4;
const int WORK_DAYS_PER_WEEK = 5;
int sum = 0;
for (int j = 0; j < NUMBER_OF_TASKS; j++) {
  int realTaskDays = taskEstimate[j] *   realDaysPerIdealDay;
  int realTaskWeeks = (realdays / WORK_DAYS_PER_WEEK);
  sum += realTaskWeeks;
}
```

OK

© 2020 ICT Intelligence

Avoid Encodings

```
public class Part {
  private String m_dsc; // The textual description
  void setName(String name) {
    m_dsc = name;
  }
}
```

NOT OK

```
PhoneNumber phoneString;
// name not changed when type changed!
```

```
public class Part {
  private String description;
  void setDescription(String description) {
    this.description = description;
  }
}
```

```
PhoneNumber phone;
```

OK

© 2020 ICT Intelligence

Avoid mental mapping

- Readers shouldn’t have to mentally translate your names into other names they already know.

© 2020 ICT Intelligence

Class names

- Classes and objects should have noun or noun phrase names
  - Customer, WikiPage, Account, and AddressParser.
- Use names from the problem domain
  - Customer, Bookmark, Document, Measurement
- A class should have a single responsibility
  - The class name should convey its purpose
- Use compound names if necessary
  - Example: TextStyle, LevelEditor
  - Use no more than two or three words
- Avoid words like Manager, Processor, Data, Object or Info in the name of a class.
- A class name should not be a verb.

© 2020 ICT Intelligence