HOW TO

TRAIN A MONSTER!

TRAIN THE "MONSTER"

OUR GOAL:
WE MUST TEACH THE MONSTER HOW TO JUGGLE

TRAIN THE "MONSTER"

OUR GOAL:
WE MUST TEACH THE MONSTER HOW TO JUGGLE

BUT...:
- IT IS DANGEROUS!

TRAIN THE "**MONSTER**"

OUR GOAL:
WE MUST TEACH THE MONSTER HOW TO JUGGLE

BUT...:
- IT IS DANGEROUS!
- WE DON'T KNOW THE TRICKS IT ALREADY CAN DO

TRAIN THE "MONSTER"

OUR GOAL:
WE MUST TEACH THE MONSTER HOW TO JUGGLE

BUT...:
- IT IS DANGEROUS!
- WE DON'T KNOW THE TRICKS IT ALREADY CAN DO
- WE CAN'T UNDERSTAND IT

TRAIN THE "MONSTER"

OUR GOAL:
WE MUST TEACH THE MONSTER HOW TO JUGGLE

BUT...:
- IT IS DANGEROUS!
- WE DON'T KNOW THE TRICKS IT ALREADY CAN DO
- WE CAN'T UNDERSTAND IT
- WE DON'T HAVE MUCH TIME

HOW TO TRAIN A "MONSTER"

- STEP 1: ANALYSE THE "**MONSTER**"

HOW TO TRAIN A "MONSTER"

- STEP 1: ANALYSE THE "**MONSTER**"
- STEP 2: ENABLE COMMUNICATION

HOW TO TRAIN A "MONSTER"

- STEP 1: ANALYSE THE "**MONSTER**"
- STEP 2: ENABLE COMMUNICATION
- STEP 3: ISOLATE DANGEROUS BODY PARTS

HOW TO TRAIN A "MONSTER"

- STEP 1: ANALYSE THE "**MONSTER**"
- STEP 2: ENABLE COMMUNICATION
- STEP 3: ISOLATE DANGEROUS BODY PARTS
- STEP 4: LEARN ITS BEHAVIOUR

HOW TO TRAIN A "MONSTER"

- STEP 1: ANALYSE THE "**MONSTER**"
- STEP 2: ENABLE COMMUNICATION
- STEP 3: ISOLATE DANGEROUS BODY PARTS
- STEP 4: LEARN ITS BEHAVIOUR
- FINALLY: TRAIN IT!

THE **REAL** MONSTER

```
public String printStatement() { //Long Method in Diff Class
    List<Quintet<String, Date, String, Integer, Integer>> allTs = new ArrayList<>();
    MongoCollection<Document> col = db.getCollection(Globals.DocCfg[0]);
    MongoCursor<Document> c = col.find(eq(Globals.DocCfg[1], userId)).cursor();
    try {
        while (c.hasNext()) {
            var doc : Document = c.next();
            allTs.add(new Quintet<>(
                    doc.getString(Globals.DocCfg[3]),
                    doc.getDate(Globals.DocCfg[2]),
                    doc.getString(Globals.DocCfg[5]),
                    doc.getInteger(Globals.DocCfg[4]),
                    doc.getString(Globals.DocCfg[5]).equals(Globals.SOLD) ? doc.getInteger(Globals.DocCfg[4])*-1 : d
        }
    }
    catch (Exception exc){
        mongoLogger.log(Level.SEVERE, exc.getMessage());
        //we should not get here, but John said just in case we do....
        //compensate();
    }
    finally {
        c.close();
    }

    Map<String, Quintet<String, Date, String, Integer, Integer>> gt = new LinkedHashMap<>();
    allTs.forEach(tr -> {
        String key = tr.getValue0();
        if (gt.containsKey(key)) {
            var t : Quintet<String, Date, String, Integer, Integer> = gt.get(key);
            gt.replace(key, new Quintet<>(t.getValue0(), tr.getValue1(), tr.getValue2(), tr.getValue3(), t.getValue4
        } else {
            gt.put(key, new Quintet<>(tr.getValue0(), tr.getValue1(), tr.getValue2(), tr.getValue3(), tr.getValue4()
        }
    });

    StringWriter sw = new StringWriter();
    sw.append(Globals.StmCfg[0]);
    sw.append("\n");
    for (String key:gt.keySet()) {
        var t : Quintet<String, Date, String, Integer, Integer> = gt.get(key);
        String sn = t.getValue0();
        int nOfShs = t.getValue4();
        DecimalFormat dc = new DecimalFormat(Globals.StmCfg[1]);
        SimpleDateFormat df = new SimpleDateFormat(Globals.StmCfg[2]);
        var cl : HttpClient = HttpClient.newHttpClient(); //Hardcoded Var in method
        var r : HttpRequest = HttpRequest
                .newBuilder(URI.create(Globals.StmCfg[3] + URLEncoder.encode(sn, StandardCharsets.UTF_8 ).replace( ta
                .header(Globals.StmCfg[4], Globals.StmCfg[5] )
                .header(Globals.StmCfg[6], Globals.StmCfg[7])
                .header(Globals.StmCfg[8], value: "7c9e6679-7425-40de-944b-e07fc1f90ae7")
                .build();//Diff Param, Global as Param, Hardcoded Var in Method Call

        int sc = 0;
        double prc = 0;
        try {
            var res : HttpResponse<String> = cl.send(r, HttpResponse.BodyHandlers.ofString());
            sc = res.statusCode();
            if (sc != 401) {
                String body = res.body();
```

STEP 1: ANALYSE THE "MONSTER"

ANALYSE THE "MONSTER"

WE CANT CHANGE SOMETHING WE DONT UNDERSTAND

- DONT CHANGE THE CODE – ANALYSE ONLY!
- REMOVING CLUTTER BY LAYOUTING
- ADD COMMENTS FOR LEGACY CODE SMELLS

```
public PortfolioService(int userId) {
    this.userId = userId;
    db = MONGO_CLIENT.getDatabase(Globals.DBConf[2]); //Hardcoded Var in Constructor
}
```

STEP 2: ENABLE COMMUNICATION

CODE COMMUNICATES WITH NAMING

WITHOUT PROPER NAMING, IT TALKS NONSENSE

ENABLE COMMUNICATION

REMOVE THE NONSENSE BY PROPER NAMING

RENAME PRIVATE VARIABLES/METHODS (LOW RISK REFACTORING)

LOOK AT THE CONTEXT OF THE VARIABLE/METHOD

- WHO USES IT?
- WHAT IS THE PURPOSE?

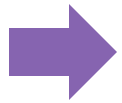
LONG OBVIOUS NAMES > SHORT GENERIC NAMES

STEP 3: ISOLATE DANGEROUS BODY PARTS

ISOLATE DANGEROUS BODY PARTS

ISOLATE THE DANGER BY INTRODUCING SEAMS FOR GLOBAL DEPENDENCIES (PRODUCTION DB/API...)

```
public PortfolioService(int userId) {
    this.userId = userId;
    db = MONGO_CLIENT.getDatabase(Globals.DBConf[2]);
}
```



```
public PortfolioService(int userId, MongoDatabase db) {
    this.userId = userId;
    this.db = db;
}

public PortfolioService(int userId) {
    this(userId, MONGO_CLIENT.getDatabase(STOCK_PORTFOLIO_DB_NAME));
}
```

STEP 4: LEARN ITS BEHAVIOUR

LEARN ITS BEHAVIOUR

USE CHARACTERIZATION TESTS AS A TOOL FOR LEARNING THE CURRENT BEHAVIOUR

- WRITE APPROVAL TESTS
- CHECK THE COVERAGE
- RUN A MUTATION TEST

```
@Test
public void printStatement() {
    PortfolioService testee = new PortfolioService( userId: 1,  db: null) {
        @Override
        protected double getSharePrice(String sharename) { return 33; }
    };

    String result = testee.printStatement();

    Approvals.verify(result);
}
```

FINALLY: TRAIN IT!

TRAINING THE "MONSTER"

NOW THAT WE HAVE COVERED THE CODE WITH TESTS WE CAN:

- REFACTOR SAFELY
- ADD FEATURES USING TDD

CONGRATULATIONS!

THE MONSTER HAS FINALLY LEARNED HOW TO JUGGLE

QUESTIONS?

DON'T BE AFRAID OF "MONSTERS"

THANK YOU!

CONTACT

samuel.degelo@gmail.com
https://github.com/degeloper

LINKS

https://alcor.academy/code-renovation
https://wiki.c2.com/?SoftwareSeam
https://michaelfeathers.silvrback.com/characterization-testing
https://approvaltests.com/
https://pitest.org/

IMAGES GENERATED BY

https://www.craiyon.com/