



SOFTWARE INGENIERITZA

2

PROIEKTUA: ERREFAKTORIZAZIOA

Egileak: Mikel Martin eta David Pintado

Data: 2022/10/11

WRITE SHORT UNITS OF CODE	3
Mikel Martin bad smell	3
HASIERAKO KODEA	3
ERREFAKTORIZATUTAKO KODEA	3
EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA	3
David Pintado bad smell	4
HASIERAKO KODEA	4
ERREFAKTORIZATUTAKO KODEA	5
EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA	5
WRITE SIMPLE UNITS OF CODE	6
Mikel Martin bad smell	6
HASIERAKO KODEA	6
ERREFAKTORIZATUTAKO KODEA	6
EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA	7
David Pintado bad smell	8
HASIERAKO KODEA	8
ERREFAKTORIZATUTAKO KODEA	8
EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA	9
DUPLICATE CODE	9
Mikel Martin bad smell	9
HASIERAKO KODEA	9
ERREFAKTORIZATUTAKO KODEA	10
EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA	10
David Pintado bad smell	10
HASIERAKO KODEA	10
ERREFAKTORIZATUTAKO KODEA	11
EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA	11
KEEP UNIT INTERFACES SMALL	12
Mikel Martin bad smell	12
HASIERAKO KODEA	12
ERREFAKTORIZATUTAKO KODEA	12
EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA	12
David Pintado bad smell	13
HASIERAKO KODEA	13
ERREFAKTORIZATUTAKO KODEA	13
EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA	13

WRITE SHORT UNITS OF CODE

1. Mikel Martin bad smell

HASIERAKO KODEA

Hemen aurkitu daiteke “write short units of code” smell-a errefaktorizatzeko aurkitu dudan kodea:

```
1219 public boolean mezuaBidali(User igorlea, String hartzailea, String titulo, String test, Elkarriketa elkarriketa) {
1220     User igorle = db.find(User.class, igorlea.getUsername());
1221     User hartzaile = db.find(User.class, hartzailea);
1222     Elkarriketa elk=null;
1223     if(hartzaile==null) {
1224         return false;
1225     }else {
1226         db.getTransaction().begin();
1227         Message m = new Message(igorle, hartzaile, test);
1228         db.persist(m);
1229         if(elkarriketa!=null) {
1230             elk = db.find(Elkarriketa.class, elkarriketa.getElkarriketaNumber());
1231         }else {
1232             elk= new Elkarriketa(titulo, igorle, hartzaile);
1233             db.persist(elk);
1234             m.setElkarriketa(elk);
1235             igorle.addElkarriketak(elk);
1236             hartzaile.addElkarriketak(elk);
1237         }
1238         elk.addMezua(m);
1239         igorle.addBidalitakoMezuak(m);
1240         hartzaile.addJasotakoMezuak(m);
1241         db.getTransaction().commit();
1242         return true;
1243     }
1244 }
```

ERREFAKTORIZATUTAKO KODEA

Hemen aurkitu daiteke kodea errefaktorizatuta:

```
1219 public boolean mezuaBidali(User igorlea, String hartzailea, String titulo, String test, Elkarriketa elkarriketa) {
1220     User igorle = db.find(User.class, igorlea.getUsername());
1221     User hartzaile = db.find(User.class, hartzailea);
1222     Elkarriketa elk=null;
1223     if(hartzaile==null) {
1224         return false;
1225     }else {
1226         db.getTransaction().begin();
1227         Message m = new Message(igorle, hartzaile, test);
1228         db.persist(m);
1229         elkarriketaEguneratu(titulo, elkarriketa, igorle, hartzaile, m);
1230         db.getTransaction().commit();
1231         return true;
1232     }
1233 }

1235 private void elkarriketaEguneratu(String titulo, Elkarriketa elkarriketa, User igorle, User hartzaile,Message m) {
1236     Elkarriketa elk;
1237     if(elkarriketa!=null) {
1238         elk = db.find(Elkarriketa.class, elkarriketa.getElkarriketaNumber());
1239     }else {
1240         elk= new Elkarriketa(titulo, igorle, hartzaile);
1241         db.persist(elk);
1242         m.setElkarriketa(elk);
1243         igorle.addElkarriketak(elk);
1244         hartzaile.addElkarriketak(elk);
1245     }
1246     elk.addMezua(m);
1247     igorle.addBidalitakoMezuak(m);
1248     hartzaile.addJasotakoMezuak(m);
1249 }
```

EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA

Egindako errefaktORIZAZIOAREN deskribapenari dagokionez, ikus daitekeen bezala mezuaBidali metodoaren lerro kopurua jaitsi egin dut, metodoaren kodea murriztuz. ElkarrizketaEguneratu metodo berri bat sortu dut lehen mezua bidali metodoan zegoen kodetik aterata. Metodo horrek parametro bezala pasatako elkarrizketa ez null bazen berri bat sortzen du eta dagokien User-ei gehitzen die, bestela soilik bilatzen du.

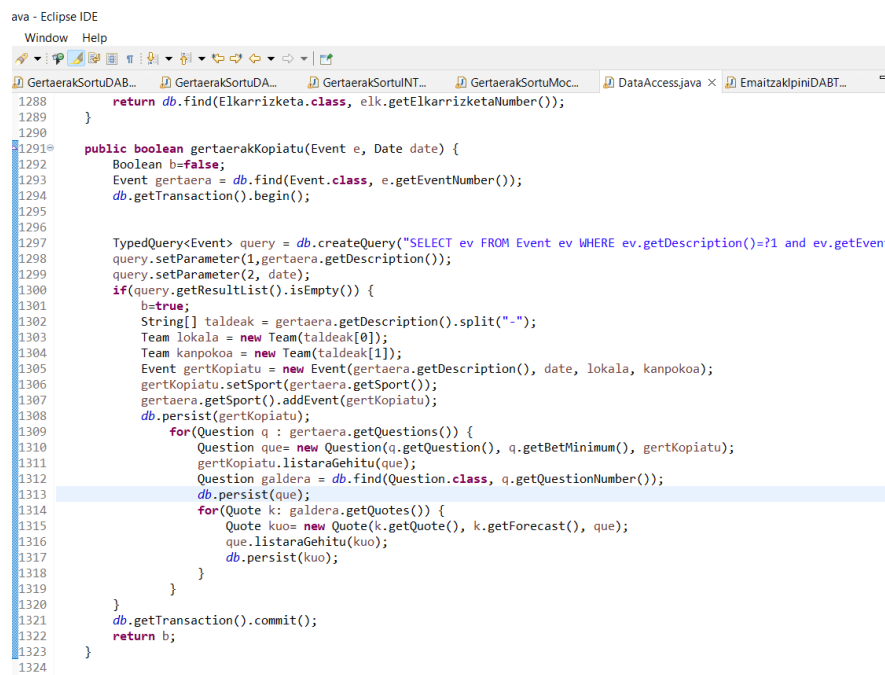
Lehengo mezuaBidali kodeak 25 lerro zituen oraingoan aldiz, mezuaBidali metodoak 14 lerro ditu eta elkarrizketaEguneratu metodo berriak beste 14.

Ondorioz kodearen mantenurako errazagoa da, metodoak lerro kopuru baxuago duenean ulertzeko hobeagoa baita.

2. David Pintado bad smell

HASIERAKO KODEA

Hemen aurkitu daiteke “write short units of code” smell-a errefaktORIZATZEKO aurkitu dudana kodea:



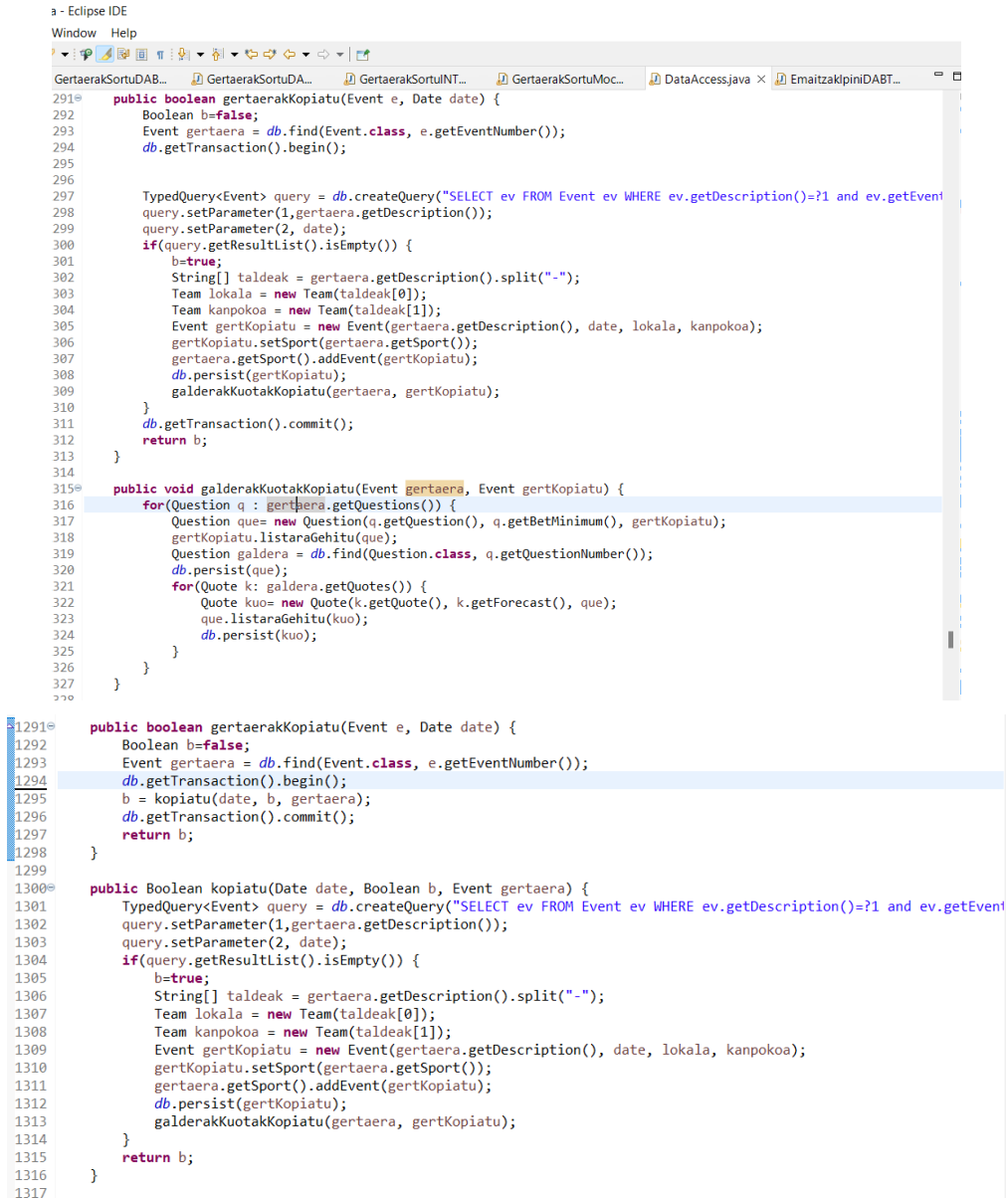
```

ava - Eclipse IDE
Window Help
GertaerakSortuDAB... GertaerakSortuDA... GertaerakSortuINT... GertaerakSortuMoc... DataAccess.java × EmailzakipiniDABT...
1288     return db.find(Elkarrizketa.class, elk.getElkarrizketaNumber());
1289 }
1290
1291 public boolean gertaerakKopiatu(Event e, Date date) {
1292     Boolean b=false;
1293     Event gertaera = db.find(Event.class, e.getEventNumber());
1294     db.getTransaction().begin();
1295
1296     TypedQuery<Event> query = db.createQuery("SELECT ev FROM Event ev WHERE ev.getDescription()=?1 and ev.getEventNumber()=?2");
1297     query.setParameter(1,gertaera.getDescription());
1298     query.setParameter(2, date);
1299     if(query.getResultList().isEmpty()) {
1300         b=true;
1301         String[] taldeak = gertaera.getDescription().split("-");
1302         Team lokala = new Team(taldeak[0]);
1303         Team kanpokoa = new Team(taldeak[1]);
1304         Event gertKopiatu = new Event(gertaera.getDescription(), date, lokala, kanpokoa);
1305         gertKopiatu.setSport(gertaera.getSport());
1306         gertaera.getSport().addEvent(gertKopiatu);
1307         db.persist(gertKopiatu);
1308         for(Question q : gertaera.getQuestions()) {
1309             Question que= new Question(q.getQuestion(), q.getBetMinimum(), gertKopiatu);
1310             gertKopiatu.listaraGehitu(que);
1311             Question galdera = db.find(Question.class, q.getQuestionNumber());
1312             db.persist(que);
1313             for(Quote k: galdera.getQuotes()) {
1314                 Quote kuo= new Quote(k.getQuote(), k.getForecast(), que);
1315                 que.listaraGehitu(kuo);
1316                 db.persist(kuo);
1317             }
1318         }
1319     }
1320     db.getTransaction().commit();
1321     return b;
1322 }
1323
1324

```

ERREFAKTORIZATUTAKO KODEA

Honela geratu da kodea behin errefaktORIZATUTA:



```

291 public boolean gertaerakKopiatu(Event e, Date date) {
292     Boolean b=false;
293     Event gertaera = db.find(Event.class, e.getEventNumber());
294     db.getTransaction().begin();
295
296     TypedQuery<Event> query = db.createQuery("SELECT ev FROM Event ev WHERE ev.getDescription()=?1 and ev.getEventNumber()=?2");
297     query.setParameter(1,gertaera.getDescription());
298     query.setParameter(2, date);
299     if(query.getResultList().isEmpty()) {
300         b=true;
301         String[] taldeak = gertaera.getDescription().split("-");
302         Team lokala = new Team(taldeak[0]);
303         Team kanpokoa = new Team(taldeak[1]);
304         Event gertKopiatu = new Event(gertaera.getDescription(), date, lokala, kanpokoa);
305         gertKopiatu.setSport(gertaera.getSport());
306         gertaera.getSport().addEvent(gertKopiatu);
307         db.persist(gertKopiatu);
308         galderakKuotakKopiatu(gertaera, gertKopiatu);
309     }
310     db.getTransaction().commit();
311     return b;
312 }
313
314 public void galderakKuotakKopiatu(Event gertaera, Event gertKopiatu) {
315     for(Question q : gertaera.getQuestions()) {
316         Question que= new Question(q.getQuestion(), q.getBetMinimum(), gertKopiatu);
317         gertKopiatu.listaraGehitu(que);
318         Question galdera = db.find(Question.class, q.getQuestionNumber());
319         db.persist(que);
320         for(Quote k: galdera.getQuotes()) {
321             Quote kuo= new Quote(k.getQuote(), k.getForecast(), que);
322             que.listaraGehitu(kuo);
323             db.persist(kuo);
324         }
325     }
326 }
327 }
328
1291 public boolean gertaerakKopiatu(Event e, Date date) {
1292     Boolean b=false;
1293     Event gertaera = db.find(Event.class, e.getEventNumber());
1294     db.getTransaction().begin();
1295     b = kopiatu(date, b, gertaera);
1296     db.getTransaction().commit();
1297     return b;
1298 }
1299
1300 public Boolean kopiatu(Date date, Boolean b, Event gertaera) {
1301     TypedQuery<Event> query = db.createQuery("SELECT ev FROM Event ev WHERE ev.getDescription()=?1 and ev.getEventNumber()=?2");
1302     query.setParameter(1,gertaera.getDescription());
1303     query.setParameter(2, date);
1304     if(query.getResultList().isEmpty()) {
1305         b=true;
1306         String[] taldeak = gertaera.getDescription().split("-");
1307         Team lokala = new Team(taldeak[0]);
1308         Team kanpokoa = new Team(taldeak[1]);
1309         Event gertKopiatu = new Event(gertaera.getDescription(), date, lokala, kanpokoa);
1310         gertKopiatu.setSport(gertaera.getSport());
1311         gertaera.getSport().addEvent(gertKopiatu);
1312         db.persist(gertKopiatu);
1313         galderakKuotakKopiatu(gertaera, gertKopiatu);
1314     }
1315     return b;
1316 }
1317 }

```

EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA

"write short units of code" smell-a errefaktORIZATZEKO, lerro kopurua jeitsi behar da. "Building Maintainable Software" liburuan esaten dutenez, 15 lerroko koderak mugatu behar gara. Beraz, barruko kodea hartu, eta beste metodo batean jarri dut; eta hurrengo argazkian, berdina egin izan behar dut, oraindik ere, 15 lerro kopuru baino gehiago zuelako metodoak. Beraz, orain hiru metodo lortu ditugu, beraien artean deitzen direlarik, aurreko modu onean funtzionatzeko.

WRITE SIMPLE UNITS OF CODE

3. Mikel Martin bad smell

HASIERAKO KODEA

Hemen aurkitu daiteke “write simple units of code” smell-a errefaktORIZATZEKO aurkitu dudan kodea:

```
public boolean gertaeraEzabatu(Event ev) {
    Event event = db.find(Event.class, ev);
    boolean resultB = true;
    List<Question> listQ = event.getQuestions();

    for(Question q : listQ) {
        if(q.getResult() == null) {
            resultB = false;
        }
    }
    if(resultB == false) {
        return false;
    }
    TypedQuery<Quote> Qquery = db.createQuery("SELECT q FROM Quote q WHERE q.getQuestion().getEvent().getEventNumber()=?1", Quote.class);
    Qquery.setParameter(1, event.getEventNumber());
    List<Quote> listQUO = Qquery.getResultList();
    for(int j=0; j<listQUO.size(); j++) {
        Quote quo = db.find(Quote.class, listQUO.get(j));
        for(int i=0; i<quo.getApustuak().size(); i++) {
            ApustuAnitza apustuAnitza = quo.getApustuak().get(i).getApustuAnitza();
            ApustuAnitza apl = db.find(ApustuAnitza.class, apustuAnitza.getApustuAnitzaNumber());
            db.getTransaction().begin();
            apl.removeApustua(quo.getApustuak().get(i));
            db.getTransaction().commit();
            if(apl.getApustuak().isEmpty() && !apl.getEgoera().equals("galduta")) {
                this.apustuaEzabatu(apl.getUser(), apl);
            }
            else if(!apl.getApustuak().isEmpty() && apl.irabazitaMarkatu()){
                this.ApustuaIrabazi(apl);
            }
            db.getTransaction().begin();
            Sport spo = quo.getQuestion().getEvent().getSport();
            spo.setApustuKantitatea(spo.getApustuKantitatea()-1);
            KirolEstatistikak ke=apl.getUser().kirolEstatistikakLortu(spo);
            ke.setKont(ke.getKont()-1);
            db.getTransaction().commit();
        }
    }
    db.getTransaction().begin();
    db.remove(event);
    db.getTransaction().commit();
    return true;
}
```

ERREFAKTORIZATUTAKO KODEA

Hemen aurkitu daiteke kodea errefaktORIZATUTA:

```
public boolean gertaeraEzabatu(Event ev) {
    Event event = db.find(Event.class, ev);
    boolean resultB = true;
    List<Question> listQ = event.getQuestions();

    resultB = EmaitzaGuztiakJarrita(resultB, listQ);
    if(resultB == false) {
        return false;
    }
    TypedQuery<Quote> Qquery = db.createQuery("SELECT q FROM Quote q WHERE q.getQuestion().getEvent().getEventNumber()=?1", Quote.class);
    Qquery.setParameter(1, event.getEventNumber());
    List<Quote> listQUO = Qquery.getResultList();
    for(int j=0; j<listQUO.size(); j++) {
        Quote quo = db.find(Quote.class, listQUO.get(j));
        for(int i=0; i<quo.getApustuak().size(); i++) {
            ApustuAnitza apustuAnitza = quo.getApustuak().get(i).getApustuAnitza();
            ApustuAnitza apl = db.find(ApustuAnitza.class, apustuAnitza.getApustuAnitzaNumber());
            db.getTransaction().begin();
            apl.removeApustua(quo.getApustuak().get(i));
            db.getTransaction().commit();
            apustuaBukatutzaEman(apl);
            db.getTransaction().begin();
            Sport spo = quo.getQuestion().getEvent().getSport();
            spo.setApustuKantitatea(spo.getApustuKantitatea()-1);
            KirolEstatistikak ke=apl.getUser().kirolEstatistikakLortu(spo);
            ke.setKont(ke.getKont()-1);
            db.getTransaction().commit();
        }
    }
    db.getTransaction().begin();
    db.remove(event);
    db.getTransaction().commit();
    return true;
}
```

```

private boolean emaitzaGuztiakJarrita(boolean resultB, List<Question> listQ) {
    for(Question q : listQ) {
        if(q.getResult() == null) {
            resultB = false;
        }
    }
    return resultB;
}

private void apustuaBukatutzatEman(ApustuAnitza ap1) {
    if(ap1.getApustuak().isEmpty() && !ap1.getEgoera().equals("galduta")) {
        this.apustuaEzabatu(ap1.getUser(), ap1);
    } else if(!ap1.getApustuak().isEmpty() && ap1.irabazitaMarkatu()){
        this.ApustuaIrabazi(ap1);
    }
}

```

EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA

Egindako errefaktORIZAZIOARI dagokionez, DataAccesseko gertaerakEzabatu metodoaren konplexutasuna murriztu dut. Lehen 8 estalpen baldintza aurkitu genezaken edo “Building Maintainable Software” liburuak esaten duen bezala 8 “branch”es. Baina oraingo kodea begiratzuz 4 estalpen baldintza aurkitu daitezke. Arrazoia honakoa da: for begizta eta if baldintza bat gertaerak ezabatu metodotik atera ditut, emaitzaGuztiakJarrita metodoa sortuz. Metodo horri beharrezkoak diren parametroak pasa zaizkio kuota lista eta bueltatuko duen boolearra.

Gainera, bigarren metodo bat gertaera ezbatutik atera dut, apustuaBukatutzatEman metodoa sortuz. Metodoari beharrezkoa den parametroa pasa diot, hau da, apustu anitz bat.

Ondorioz, nahiz eta metodoaren konplexutasuna ez da guztiz jaitsi (metodoak beste metodo txikiei dei egingo dielako), metodoaren mantenuari dagokionez orain errazagoa da akatsak zuzentzea.

4. David Pintado bad smell

HASIERAKO KODEA

Hurrengo kode honetan ikusi dezakegun bezala, “write simple units of code” smell-a aurkitu dezakegu:

```

1103
1104 public void EmaizakIpini(Quote quote) throws EventNotFinished{
1105
1106     Quote q = db.find(Quote.class, quote);
1107     String result = q.getForecast();
1108
1109     if(new Date().compareTo(q.getQuestion().getEvent().getEventDate())<0)
1110         throw new EventNotFinished("Data gaurkoa baina altuagoa da");
1111
1112     Vector<Apustua> listApustuak = q.getApustuak();
1113     db.getTransaction().begin();
1114     Question que = q.getQuestion();
1115     Question question = db.find(Question.class, que);
1116     question.setResult(result);
1117     for(Quote quo: question.getQuotes()) {
1118         for(Apustua apu: quo.getApustuak()) {
1119
1120             Boolean b=apu.galdutaMarkatu(quo);
1121             if(b) {
1122                 apu.getApustuAnitza().setEgoera("galduta");
1123             }else {
1124                 apu.setEgoera("irabazita");
1125             }
1126         }
1127     }
1128     db.getTransaction().commit();
1129     for(Apustua a : listApustuak) {
1130         db.getTransaction().begin();
1131         Boolean bool=a.getApustuAnitza().irabazitaMarkatu();
1132         db.getTransaction().commit();
1133         if(bool) {
1134             this.ApustuaIrabazi(a.getApustuAnitza());
1135         }
1136     }
1137 }
1138

```

ERREFAKTORIZATUTAKO KODEA

Honela geratu da kodea behin errefaktORIZATUTA:

```

public void EmaizakIpini(Quote quote) throws EventNotFinished{

    Quote q = db.find(Quote.class, quote);
    String result = q.getForecast();

    if(new Date().compareTo(q.getQuestion().getEvent().getEventDate())<0)
        throw new EventNotFinished("Data gaurkoa baina altuagoa da");
    Vector<Apustua> listApustuak = q.getApustuak();
    db.getTransaction().begin();
    Question que = q.getQuestion();
    egoeraAdierazi(result, que);
    db.getTransaction().commit();
    for(Apustua a : listApustuak) {
        db.getTransaction().begin();
        Boolean bool=a.getApustuAnitza().irabazitaMarkatu();
        db.getTransaction().commit();
        if(bool) {
            this.ApustuaIrabazi(a.getApustuAnitza());
        }
    }
}

public void egoeraAdierazi(String result, Question que) {
    Question question = db.find(Question.class, que);
    question.setResult(result);
    for(Quote quo: question.getQuotes()) {
        for(Apustua apu: quo.getApustuak()) {

            Boolean b=apu.galdutaMarkatu(quo);
            if(b) {
                apu.getApustuAnitza().setEgoera("galduta");
            }else {
                apu.setEgoera("irabazita");
            }
        }
    }
}
}

```


EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA

Hasierako egoerako metodoa aztertuta, ikusi daiteke “write simple units of code” smell-a dagoela, guztira 6 estalpen baldintza aurkitu ditzakegu, edo “Building Maintainable Software” liburuak esaten duen bezala, 6 “branch”es. Arazo hau konpontzeko, metodoaren kode zati bat har dezakegu, eta metodo batean txertatu. Noski, bere funtzionamendua berdina izanez, metodoaren portaera mantenduz.

Horixe da, hain zuzen ere, zuzenketan ikusi daitekeena. Lehenengo for begiztan dagoena, eta goiko bi lerroak, koherentzia mantentzeko, hartu ditugu, eta hori, metodo berri batean txertatu; egoera adierazi, hain zuzen.

Honekin lortzen duguna zera da, metodoaren mantenua errazagoa izatea, akatsak zuzentzeko eta aurkitzeko. Izan ere, honekin nahiz eta metodoaren konplexutasuna ez da guztiz jaitsi (metodoak beste metodo txikiei dei egingo dielako).

DUPLICATE CODE

5. Mikel Martin bad smell

HASIERAKO KODEA

Hemen aurkitu daiteke “duplicate code” smell-a errefaktORIZATZEKO aurkitu dudak kodea:

```

: Duplication
Transaction t1 = new Transaction(reg1, apA1.getBalioa(), new Date(), 1 "ApustuaEgin");
: Duplication
Transaction t3 = new Transaction(reg2, apA4.getBalioa(), new Date(), 2 "ApustuaEgin");
: Duplication
Transaction t4 = new Transaction(reg3, apA5.getBalioa(), new Date(), 3 "ApustuaEgin");
: Duplication
Transaction t5 = new Transaction(reg4, apA3.getBalioa(), new Date(), 4 "ApustuaEgin");
: Duplication
Transaction t6 = new Transaction(reg4, apA6.getBalioa(), new Date(), 5 "ApustuaEgin");
: Duplication
Transaction t7 = new Transaction(reg1, apA7.getBalioa(), new Date(), 6 "ApustuaEgin");
: Duplication
Transaction t8 = new Transaction(reg1, apA8.getBalioa(), new Date(), 7 "ApustuaEgin");
: Duplication
Transaction t9 = new Transaction(reg2, apA9.getBalioa(), new Date(), 8 "ApustuaEgin");
: Duplication
Transaction t10 = new Transaction(reg2, apA10.getBalioa(), new Date(), 9 "ApustuaEgin");
: Duplication
Transaction t11 = new Transaction(reg3, apA11.getBalioa(), new Date(), 10 "ApustuaEgin");
: Duplication
Transaction t12 = new Transaction(reg3, apA12.getBalioa(), new Date(), 11 "ApustuaEgin");

reg1.addTransaction(t1);

```

File	Date	Description
iAccess.java		Replace this use of System.out or System.err by a logger.
iAccess.java		Replace this use of System.out or System.err by a logger.
iAccess.java		Define a constant instead of duplicating this literal "getDatabaseOpenMode:" 3 times. [+3 locations]
iAccess.java		Define a constant instead of duplicating this literal "ApustuaEgin" 12 times. [+12 locations]

DataAccess klasean aurkitu ditzakegu “ApustuaEgin” string-a hamaika aldiz kodean.

ERREFAKTORIZATUTAKO KODEA

Kodea errefaktoratuta honela geratu da:

```
String apustuEginHitza = "ApustuaEgin";
Transaction t1 = new Transaction(reg1, apA1.getBalioa(), new Date(), apustuEginHitza);
Transaction t3 = new Transaction(reg2, apA4.getBalioa(), new Date(), apustuEginHitza);
Transaction t4 = new Transaction(reg3, apA5.getBalioa(), new Date(), apustuEginHitza);
Transaction t5 = new Transaction(reg4, apA3.getBalioa(), new Date(), apustuEginHitza);
Transaction t6 = new Transaction(reg4, apA6.getBalioa(), new Date(), apustuEginHitza);
Transaction t7 = new Transaction(reg1, apA7.getBalioa(), new Date(), apustuEginHitza);
Transaction t8 = new Transaction(reg1, apA8.getBalioa(), new Date(), apustuEginHitza);
Transaction t9 = new Transaction(reg2, apA9.getBalioa(), new Date(), apustuEginHitza);
Transaction t10 = new Transaction(reg2, apA10.getBalioa(), new Date(), apustuEginHitza);
Transaction t11 = new Transaction(reg3, apA11.getBalioa(), new Date(), apustuEginHitza);
Transaction t12 = new Transaction(reg3, apA12.getBalioa(), new Date(), apustuEginHitza);
```

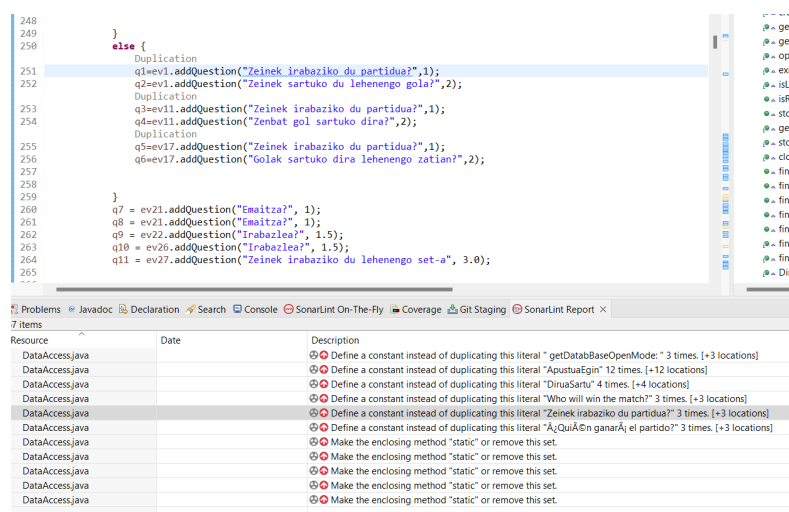
EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA

Smell hau sonar-lint-ek erakusten digu, goiko irudian agertzen den bezala. ErrefaktORIZAZIO honetan egin dudana honakoa da: "ApustuaEgin" stringa markatu eta saguko eskuineko botoia sakatuz errefaktORIZATU. Aldagai lokal bezala gorde dut apustuEginHitza stringean, lehen string-a idazten zen bakoitzean orain aldagai lokala bakarrik pasa behar da. Horrela, kodean akatsik egongo balego edota stringa aldatu nahiko bagenu behin bakarrik aldatu beharko litzateke eta ez hamaika aldiz.

6. David Pintado bad smell

HASIERAKO KODEA

Hemen aurkitu daiteke "duplicate code" smell-a errefaktORIZATZEKO aurkitu dudana kodea:



ERREFAKTORIZATUTAKO KODEA

Honela geratu da kodea behin errefaktORIZATUTA:

```

248
249     }
250     else {
251         String zeinIrabazi = "Zeinek irabaziko du partidua?";
252         Duplication
253         q1=ev1.addQuestion(zeinIrabazi,1);
254         q2=ev1.addQuestion("Zeinek sartuko du lehenengo gola?",2);
255         Duplication
256         q3=ev11.addQuestion(zeinIrabazi,1);
257         q4=ev11.addQuestion("Zenbat gol sartuko dira?",2);
258         Duplication
259         q5=ev17.addQuestion(zeinIrabazi,1);
260         q6=ev17.addQuestion("Golak sartuko dira lehenengo zatian?",2);

```

EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA

Smell hau sonar-lint-ek erakusten digu, goiko irudian agertzen den bezala. “duplicate code” smell-a dioena zera da, ez dugula kode errepikaturik eduki behar, zailtzen duelako manteinua. Konpontzeko, aldagai batean string hori sartuko dugu, eta aldagai horri deitu, string hori erabiltzen dugun bakoitzean.

KEEP UNIT INTERFACES SMALL

7. Mikel Martin bad smell

HASIERAKO KODEA

Hurrengo kode honetan ikusi dezakegun bezala, “keep unit interfaces small” smell-a aurkitu dezakegu:

```

1219 public boolean mezuaBidali(User igorlea, String hartzailea, String titulo, String test, Elkarriketa elkarriketa) {
1220     User igorle = db.find(User.class, igorlea.getUsername());
1221     User hartzaile = db.find(User.class, hartzailea);
1222     Elkarriketa elk=null;
1223     if(hartzaile==null) {
1224         return false;
1225     }else {
1226         db.getTransaction().begin();
1227         Message m = new Message(igorle, hartzaile, test);
1228         db.persist(m);
1229         elkarriketaEguneratu(titulo, elkarriketa, igorle, hartzaile, m);
1230         db.getTransaction().commit();
1231         return true;
1232     }
1233 }

```

ERREFAKTORIZATUTAKO KODEA

Honela geratu da kodea behin errefaktoratuta:

```

public boolean mezuaBidali(User igorlea, String hartzailea, ArrayList<String> mezuEgitura, Elkarriketa elkarriketa) {
    User igorle = db.find(User.class, igorlea.getUsername());
    User hartzaile = db.find(User.class, hartzailea);
    Elkarriketa elk=null;
    if(hartzaile==null) {
        return false;
    }else {
        db.getTransaction().begin();
        Message m = new Message(igorle, hartzaile, mezuEgitura.get(1));
        db.persist(m);
        elkarriketaEguneratu(mezuEgitura.get(0), elkarriketa, igorle, hartzaile, m);
        db.getTransaction().commit();
        return true;
    }
}

```

EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA

Egindako errefaktORIZAZIOAREN deskribapenari dagokionez, ikus daitekeen bezala mezuaBidali metodoak lehen 5 parametro sartzen zituen, baina oraingo honetan, aldiz, 4.

Mezuaren tituloa (gaia) eta testua stringeko arraylist batean elkartu ditut. ArrayList horren 0. posizioa beti gaia izango da eta 1. posizioan testua. Metodo honen parametro errefaktORIZAZIOAK hainbat aldaketa dakarki. Hala nola, DataAccessInterface, BLFacade eta BLFacadeImplementation klaseetan bi String pasa ordeztu string lista bat pasatzea. Gainera, mezuaBidali deitzen den GUIan aldaketa batzuk ere egin behar izan ditut, lehen pasatzen ziren bi parametroak lista batean elkartu eta lista hori parametro bezala pasa.

Ondorioz, metodoaren mantenurako abantailak dakarki; izan ere, metodo batek 4 parametro edo gutxiago dituenean errazagoa da ulertzeko.

8. David Pintado bad smell

HASIERAKO KODEA

Hurrengo kode honetan ikusi dezakegun bezala, “keep unit interfaces small” smell-a aurkitu dezakegu:

```

1224 public boolean mezuaBidali(User igorlea, String hartzailea, ArrayList<String> mezuEgitura, Elkarriketa elkarriketa)
1225 {
1226     User igorle = db.find(User.class, igorlea.getUsername());
1227     User hartzaile = db.find(User.class, hartzailea);
1228     Elkarriketa elk=null;
1229     if(hartzaile==null) {
1230         return false;
1231     }else {
1232         db.getTransaction().begin();
1233         Message m = new Message(igorle, hartzaile, mezuEgitura.get(1));
1234         db.persist(m);
1235         elkarriketaEguneratu(mezuEgitura.get(0), elkarriketa, igorle, hartzaile, m);
1236         db.getTransaction().commit();
1237         return true;
1238     }
1239 }
1240 private void elkarriketaEguneratu(String titulo, Elkarriketa elkarriketa, User igorle, User hartzaile,Message m)
1241 {
1242     Elkarriketa elk;
1243     if(elkarriketa==null) {
1244         elk = db.find(Elkarriketa.class, elkarriketa.getElkarriketaNumber());
1245     }else {
1246         elk= new Elkarriketa(titulo, igorle, hartzaile);
1247         db.persist(elk);
1248         m.setElkarriketa(elk);
1249         igorle.addElkarriketak(elk);
1250         hartzaile.addElkarriketak(elk);
1251     }
1252     elk.addMezua(m);
1253     igorle.addBidalitakoMezuak(m);
1254     hartzaile.addJasotakoMezuak(m);
1255 }

```

ERREFAKTORIZATUTAKO KODEA

Honela geratu da kodea behin errefaktoratuta:

```

1222 }
1223 }
1224 public boolean mezuaBidali(User igorlea, String hartzailea, ArrayList<String> mezuEgitura, Elkarriketa elkarriketa)
1225 {
1226     User igorle = db.find(User.class, igorlea.getUsername());
1227     User hartzaile = db.find(User.class, hartzailea);
1228     Elkarriketa elk=null;
1229     if(hartzaile==null) {
1230         return false;
1231     }else {
1232         db.getTransaction().begin();
1233         Message m = new Message(igorle, hartzaile, mezuEgitura.get(1));
1234         db.persist(m);
1235         ArrayList<User> erabiltzaileak = new ArrayList<User>();
1236         erabiltzaileak.add(igorle);
1237         erabiltzaileak.add(hartzaile);
1238         elkarriketaEguneratu(mezuEgitura.get(0), elkarriketa, erabiltzaileak, m);
1239         db.getTransaction().commit();
1240         return true;
1241     }
1242 }
1243 }
1244 private void elkarriketaEguneratu(String titulo, Elkarriketa elkarriketa, ArrayList<User> erabiltzaileak ,Message m)
1245 {
1246     Elkarriketa elk;
1247     User igorle = erabiltzaileak.get(0);
1248     User hartzaile = erabiltzaileak.get(1);
1249     if(elkarriketa==null) {
1250         elk = db.find(Elkarriketa.class, elkarriketa.getElkarriketaNumber());
1251     }else {
1252         elk= new Elkarriketa(titulo, igorle, hartzaile);
1253         db.persist(elk);
1254         m.setElkarriketa(elk);
1255         igorle.addElkarriketak(elk);
1256         hartzaile.addElkarriketak(elk);
1257     }
1258     elk.addMezua(m);
1259     igorle.addBidalitakoMezuak(m);
1260     hartzaile.addJasotakoMezuak(m);
1261 }

```

EGINDAKO ERREFAKTORIZAZIOAREN DESKRIBAPENA

Ikus daitekeen bezala elkarrizketaEguneratu metodoak lehen 5 parametro sartzen zituen, baina orain 4. Izan ere, errefaktORIZATU dugu, bi parametro lista batean sartuz. Konkreterik, argazkian agerikoa denez, User-ak dira, igorle eta hartzaile. Metodo honetara, mezuakBidali-tik deitzen denez, hor ere aldaketak egin beharko ditugu, metodoaren deia aldatuz, eta lista sortuz. Listaren 0 posizioan beti igorlea dago, eta 1ean hartzailea, beraz ez dago arazorik. Listatik atera, eta lehen bezala arituko da., Honek, metodoaren mantenurako abantailak dakarki; izan ere, metodo batek 4 parametro edo gutxiago dituenean errazagoa da ulertzeko.