ERREFAKTORIZAZIOA

Mikel Oscoz Rivas
Iñigo Gallegos Iglesias

"Write short units of code" (2. kapituloa)

storeApostuaBLFAcade - Iñigo

1-Hasierako kodea

```
@WebMethod
	public boolean storeApostua(Apostatu ap, Vector<Question> q) {
		dbManager.open(false);
		boolean emaitza = false;
		emaitza = dbManager.storeApostua(ap);
		Apostatu apostua = dbManager.lortuApostua(ap);
		for (Question quest : q) {
			int i = 0;
			boolean aurkitua = false;
			while (i < apostua.getQuote().size() && !aurkitua) {
				if
(apostua.getQuote().get(i).getQuestion().equals(quest)) {
					aurkitua = true;
					System.out.println(apostua.getApostuNumber());
					emaitza =
dbManager.addApostua(apostua.getApostuNumber(), quest);
				}
				i++;
			}
			emaitza = true;
		}
		dbManager.close();
		return emaitza;
	}
```

2- Errefaktorizatuko kodea

```
@WebMethod
	public boolean storeApostua(Apostatu ap, Vector<Question> q) {
		dbManager.open(false);
		boolean emaitza = false;
		emaitza = dbManager.storeApostua(ap);
		Apostatu apostua = dbManager.lortuApostua(ap);
		emaitza = extracted(q, emaitza, apostua);
		dbManager.close();
		return emaitza;
	}

	private boolean extracted(Vector<Question> q, boolean emaitza, Apostatu
apostua) {
		for (Question quest : q) {
			int i = 0;
			boolean aurkitua = false;
			emaitza = extracted(emaitza, apostua, quest, i, aurkitua);
			emaitza = true;
		}
		return emaitza;
	}
```

```
	private boolean extracted(boolean emaitza, Apostatu apostua, Question quest,
int i, boolean aurkitua) {
		while (i < apostua.getQuote().size() && !aurkitua) {
			if (apostua.getQuote().get(i).getQuestion().equals(quest)) {
				aurkitua = true;
				System.out.println(apostua.getApostuNumber());
				emaitza =
dbManager.addApostua(apostua.getApostuNumber(), quest);
			}
			i++;
		}
		return emaitza;
	}
    }
```

3- Egindako errefaktorizazioren deskribapena.

Hau Errefaktorizatzeko Kode zati bat atera dut metodotik ‘Extact method’ erabiliz. Ateratako kode zatia sortutako beste metodo batean sartu dut eta dei bat eginez bi metodoak komunikatzen dira.

4- Egilea

Iñigo Gallegos

returnMoney - Mikel

1-Hasierako kodea

```
public boolean returnMoney(User user, Event event) {
    boolean ok = false;
    double dirua = 0;
    try {
        User u = db.find(User.class, user);
        db.getTransaction().begin();
        for (Mugimendua m : u.getMugimenduak()) {
            if (m.getGertaera() != null) {
                if (m.getGertaera().getEventNumber() == event.getEventNumber()) {
                    dirua = m.getDiruKop();
                }
            }
        }
        u.setDirua(u.getDirua() + dirua);
        db.getTransaction().commit();
        ok = true;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return ok;
}
```

2- Errefaktorizatutako kodea

```
public boolean returnMoney(User user, Event event) {
    boolean ok = false;
    double dirua = 0;
    try {
        User u = db.find(User.class, user);
        db.getTransaction().begin();
        dirua = returnMoneyfor(event, dirua, u);
        u.setDirua(u.getDirua() + dirua);
        db.getTransaction().commit();
        ok = true;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return ok;
}

public double returnMoneyfor(Event event, double dirua, User u) {
    for (Mugimendua m : u.getMugimenduak()) {
        if (m.getGertaera() != null) {
            if (m.getGertaera().getEventNumber() == event.getEventNumber()) {
                dirua = m.getDiruKop();
            }
        }
    }
    return dirua;
}
```

3- Egindako errefaktorizazioren deskribapena

Errefaktorizazio hau egiteko “Extract method” deitzen den errefaktorizazioa erabili dut. Lehenengo for-entzako aplikatu dut, eta horrela metodotik atera dut. Parametroak begiztan erabiltzen direnak dira eta gero dirua bueltatzen da.

4- Egilea

Mikel Oscoz

"Write simple units of code" (3. kapituloa)

isLogin - Iñigo

1-Hasierako kodea

```
public Erabiltzaile isLogin(String log, String pass) {
	if (log != null && pass != null) {
		Erabiltzaile user = db.find(Erabiltzaile.class, log);
		if (user == null)
			return null;
		if (!user.getPasahitza().equals(pass))
			return null;
		return user;
	}
	return null;
}
```

2- Errefaktorizatuko kodea

```
public Erabiltzaile isLogin(String log, String pass) {
	if (log != null && pass != null) {
		return extractedisLogin(log, pass);
	}
	return null;
}
private Erabiltzaile extractedisLogin(String log, String pass) {
	Erabiltzaile user = db.find(Erabiltzaile.class, log);
	if (user == null)
		return null;
	if (!user.getPasahitza().equals(pass))
		return null;
	return user;
}
```

3- Egindako errefaktorizazioren deskribapena

Hau Errefaktorizatzeko Kode zati bat atera dut metodotik ‘Extact method’ erabiliz. Ateratako kode zatia sortutako beste metodo batean sartu dut eta dei bat eginez bi metodoak komunikatzen dira.

4- Egilea

Iñigo Gallegos

pasahitzaAldatu -Mikel

1-Hasierako kodea

```
public boolean pasahitzaAldatu(User u, String pass) {
    boolean ok = false;
    if (u != null && pass != null) {
        try {
            User us = db.find(User.class, u);
            if (us != null) {
                db.getTransaction().begin();
                us.setPasahitza(pass);
                db.persist(us);
                db.getTransaction().commit();
                ok = true;
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    return ok;
}
```

2- Errefaktorizatutako kodea

```
public boolean pasahitzaAldatu(User u, String pass) {
    boolean ok = false;
    if (u != null && pass != null) {
        ok = extractedPassAldatu(u, pass, ok);
    }
    return ok;
}

public boolean extractedPassAldatu(User u, String pass, boolean ok) {
    try {
        User us = db.find(User.class, u);
        if (us != null) {
            db.getTransaction().begin();
            us.setPasahitza(pass);
            db.persist(us);
            db.getTransaction().commit();
            ok = true;
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return ok;
}
```

3-Egindako errefaktorizazioren deskribapena

Errefaktorizazio hau, aurreko puntuan erabilitako berdina da “Extract method” erabiliz, funztio baten konplexutasun ziklomatikoa jaitsi dezakezu, eta konplexutasun ziklomatiko 4 duen funtzio bat izan ordez, bi funztio dituzu bat bikoa eta beste bat hirukoa.

4-Egilea

Mikel Oscoz

“Duplicate code” (4. kapituloa)

Define a constant instead of duplicating this literal " question= " 3 times. [+3 locations]	DataAccess.ja...
Define a constant instead of duplicating this literal "Who will win the match?" 3 times. [+3 locations]	DataAccess.ja...
Define a constant instead of duplicating this literal "Zeinek irabaziko du partidua?" 3 times. [+3 locations]	DataAccess.ja...
Define a constant instead of duplicating this literal "¿Quién ganará el partido?" 3 times. [+3 locations]	DataAccess.ja...

1)bad smell - Mikel

1-Hasierako kodea

```
public Question createQuestion(Event event, String question, float betMinimum) throws QuestionAlreadyExist {
    System.out.println(">> DataAccess: createQuestion=> event= " + event + " question= " + question + " betMinimum="
            + betMinimum);

    Event ev = db.find(Event.class, event.getEventNumber());

    if (ev.DoesQuestionExists(question))
        throw new QuestionAlreadyExist(ResourceBundle.getBundle("Etiquetas").getString("ErrorQueryAlreadyExist"));

    db.getTransaction().begin();
    Question q = ev.addQuestion(question, betMinimum);
    // db.persist(q);
    db.persist(ev); // db.persist(q) not required when CascadeType.PERSIST is added in questions
                    // property of Event class
                    // @OneToMany(fetch=FetchType.EAGER, cascade=CascadeType.PERSIST)
    db.getTransaction().commit();
    return q;

}
```

2-Errefaktorizatutako kodea

```
public Question createQuestion(EventQuestion parameterObject, float betMinimum) throws QuestionAlreadyExist {
    System.out.println(">> DataAccess: createQuestion=> event= " + parameterObject.getEvent() + quest + parameterObject.getQuestion() + " betMinimum="
            + betMinimum);

    Event ev = db.find(Event.class, parameterObject.getEvent().getEventNumber());

    if (ev.DoesQuestionExists(parameterObject.getQuestion()))
        throw new QuestionAlreadyExist(ResourceBundle.getBundle("Etiquetas").getString("ErrorQueryAlreadyExist"));

    db.getTransaction().begin();
    Question q = ev.addQuestion(parameterObject.getQuestion(), betMinimum);
    // db.persist(q);
    db.persist(ev); // db.persist(q) not required when CascadeType.PERSIST is added in questions
                    // property of Event class
                    // @OneToMany(fetch=FetchType.EAGER, cascade=CascadeType.PERSIST)
    db.getTransaction().commit();
    return q;

}
```

3- Egindako errefaktorizazioren deskribapena

Honetarako “extract local variable” errefaktorizazioa erabili dut, “question =” ordez, quest agertzen da orain, eta aldagai horren balioa String “question=” da.

4_Egilea: Mikel Oscoz

2)Bad smell - Iñigo

1-Hasierako kodea

```
if (Locale.getDefault().equals(new Locale("es"))) {
                        q1 = ev1.addQuestion("¿Quién ganará el partido?", 1);
                        q2 = ev1.addQuestion("¿Quién meterá el primer gol?", 2);
                        q3 = ev11.addQuestion("¿Quién ganará el partido?", 1);
                        q4 = ev11.addQuestion("¿Cuántos goles se marcarán?", 2);
                        q5 = ev17.addQuestion("¿Quién ganará el partido?", 1);
                        q6 = ev17.addQuestion("¿Habrá goles en la primera
parte?", 2);
                } else if (Locale.getDefault().equals(new Locale("en"))) {
                        q1 = ev1.addQuestion("Who will win the match?", 1);
                        q2 = ev1.addQuestion("Who will score first?", 2);
                        q3 = ev11.addQuestion("Who will win the match?", 1);
                        q4 = ev11.addQuestion("How many goals will be scored in
the match?", 2);
                        q5 = ev17.addQuestion("Who will win the match?", 1);
                        q6 = ev17.addQuestion("Will there be goals in the first
half?", 2);
                } else {
                        q1 = ev1.addQuestion("Zeinek irabaziko du partidua?",
1);
                        q2 = ev1.addQuestion("Zeinek sartuko du lehenengo
gola?", 2);
                        q3 = ev11.addQuestion("Zeinek irabaziko du partidua?",
1);
                        q4 = ev11.addQuestion("Zenbat gol sartuko dira?", 2);
                        q5 = ev17.addQuestion("Zeinek irabaziko du partidua?",
1);
                        q6 = ev17.addQuestion("Golak sartuko dira lehenengo
zatian?", 2);
                }
}
```

2- Errefaktorizatuko kodea

```
if (Locale.getDefault().equals(new Locale("es"))) {
                        q1 = ev1.addQuestion("¿Quién ganará el partido?", 1);
                        q2 = ev1.addQuestion("¿Quién meterá el primer gol?", 2);
                        q3 = ev11.addQuestion("¿Quién ganará el partido?", 1);
                        q4 = ev11.addQuestion("¿Cuántos goles se marcarán?", 2);
                        q5 = ev17.addQuestion("¿Quién ganará el partido?", 1);
                        q6 = ev17.addQuestion("¿Habrá goles en la primera
parte?", 2);
                } else if (Locale.getDefault().equals(new Locale("en"))) {
                        String NorkIrabaziGaldera = "Who will win the match?";
                        q1 = ev1.addQuestion(NorkIrabaziGaldera, 1);
                        q2 = ev1.addQuestion("Who will score first?", 2);
                        q3 = ev11.addQuestion(NorkIrabaziGaldera, 1);
                        q4 = ev11.addQuestion("How many goals will be scored in
the match?", 2);
```

```
                        q5 = ev17.addQuestion(NorkIrabaziGaldera, 1);
                        q6 = ev17.addQuestion("Will there be goals in the first
half?", 2);
                  } else {
                        q1 = ev1.addQuestion("Zeinek irabaziko du partidua?",
1);
                        q2 = ev1.addQuestion("Zeinek sartuko du lehenengo
gola?", 2);
                        q3 = ev11.addQuestion("Zeinek irabaziko du partidua?",
1);
                        q4 = ev11.addQuestion("Zenbat gol sartuko dira?", 2);
                        q5 = ev17.addQuestion("Zeinek irabaziko du partidua?",
1);
                        q6 = ev17.addQuestion("Golak sartuko dira lehenengo
zatian?", 2);
                  }
```

3- Egindako errefaktorizazioren deskribapena

Errefakzio hau egiteko, errepikatutako kodearen balioa atributu batean gordetzen da eta ordezkatzen da balioa agertzen den leku guztietan. Errefaktorizazio hau egiteko “extract local variable” erabili dut.

4- Egilea

Iñigo Gallegos

"Keep unit interfaces small" (5. kapituloa)

createQuote -Iñigo

1-Hasierako kodea

```
public Quote createQuote(String quote, Question question, double mult) {
		System.out.println(">> DataAccess: createQote=> Quote= " + quote
+ " question= " + question);
		Question quest = db.find(Question.class,
question.getQuestionNumber());
		Quote q;
		Integer zenb = -1;
		if (quest.doesQuoteExist(quote))
			q = new Quote(zenb, "", null, 0);
		else {
			db.getTransaction().begin();
			q = quest.addQuote(quote, mult);
			db.persist(q);
			db.getTransaction().commit();
		}
		return q;
}
```

2- Errefaktorizatuko kodea

DATA ACCESS

```
public Quote createQuote(CreateQuoteParameter parameterObject, double mult) {
		System.out.println(">> DataAccess: createQote=> Quote= " +
parameterObject.quote + " question= " + parameterObject.question);
		Question quest = db.find(Question.class,
parameterObject.question.getQuestionNumber());
		Quote q;
		Integer zenb = -1;
		if (quest.doesQuoteExist(parameterObject.quote))
			q = new Quote(zenb, "", null, 0);
		else {
			db.getTransaction().begin();
			q = quest.addQuote(parameterObject.quote, mult);
			db.persist(q);
			db.getTransaction().commit();
		}
		return q;
}
```

BLFACADE

```
import dataAccess.CreateQuoteParameter;

@WebMethod
	public Quote createQuote(String quote, Question question, double mult) {
		dbManager.open(false);
```

```
                Quote q = null;
                q = dbManager.createQuote(new CreateQuoteParameter(quote, question),
mult);
                dbManager.close();
                return q;
        }
```

KLASE BERRI BAT DATUBASEKO PAKETEAN

```
package dataAccess;
import domain.Question;
public class CreateQuoteParameter {
        public String quote;
        public Question question;
        public CreateQuoteParameter(String quote, Question question) {
                this.quote = quote;
                this.question = question;
        }
}
```

3- Egindako errefaktorizazioren deskribapena

kode zati hau errefaktorizatzeko, “Introduce parameter object” erabili dut. Errefaktorizazio honek, metodori sartutako parametro batzuei batzen ditu klase batean eta sartzen du parametro bezala, horrela, metodoak duen parametro kopurua jaisten da.

4- Egilea

Iñigo Gallegos

createQuestion - Mikel

1-Hasierako kodea

```
public Question createQuestion(Event event, String question, float betMinimum) throws QuestionAlreadyExist {
    System.out.println(">> DataAccess: createQuestion=> event= " + event + " question= " + question + " betMinimum="
            + betMinimum);

    Event ev = db.find(Event.class, event.getEventNumber());

    if (ev.DoesQuestionExists(question))
        throw new QuestionAlreadyExist(ResourceBundle.getBundle("Etiquetas").getString("ErrorQueryAlreadyExist"));

    db.getTransaction().begin();
    Question q = ev.addQuestion(question, betMinimum);
    // db.persist(q);
    db.persist(ev); // db.persist(q) not required when CascadeType.PERSIST is added in questions
                    // property of Event class
                    // @OneToMany(fetch=FetchType.EAGER, cascade=CascadeType.PERSIST)
    db.getTransaction().commit();
    return q;

}
```

2- Errefaktorizatutako kodea

```
public Question createQuestion(EventQuestion parameterObject, float betMinimum) throws QuestionAlreadyExist {
    System.out.println(">> DataAccess: createQuestion=> event= " + parameterObject.getEvent() + quest + parameterObject.getQuestion() + " betMinimum="
            + betMinimum);

    Event ev = db.find(Event.class, parameterObject.getEvent().getEventNumber());

    if (ev.DoesQuestionExists(parameterObject.getQuestion()))
        throw new QuestionAlreadyExist(ResourceBundle.getBundle("Etiquetas").getString("ErrorQueryAlreadyExist"));

    db.getTransaction().begin();
    Question q = ev.addQuestion(parameterObject.getQuestion(), betMinimum);
    // db.persist(q);
    db.persist(ev); // db.persist(q) not required when CascadeType.PERSIST is added in questions
                    // property of Event class
                    // @OneToMany(fetch=FetchType.EAGER, cascade=CascadeType.PERSIST)
    db.getTransaction().commit();
    return q;

}
```

Klase berria

```java
package dataAccess;

import domain.Event;

public class EventQuestion {
    private Event event;
    private String question;

    public EventQuestion(Event event, String question) {
        this.event = event;
        this.question = question;
    }

    public Event getEvent() {
        return event;
    }

    public void setEvent(Event event) {
        this.event = event;
    }

    public String getQuestion() {
        return question;
    }

    public void setQuestion(String question) {
        this.question = question;
    }
}
```

3- Egindako errefaktorizazioren deskribapena

Honetarako “Introduce parameter object” errefaktorizazioa, honek, bi parametro sartu ordez, adibidez objektu bat bi atributuekin sartzen du parametro bezala, horrela, parametro kopurua jaisten.

4- Egilea

Mikel Oscoz