eman ta zabal zazu

Universidad del País Vasco Euskal Herriko Unibertsitatea

REFACTORIZACIÓN

DOCUMENTACIÓN

Hasib Murib, Andrea Garcia

- **Refactorización**

"Write short units of code" (capítulo 2): Reducir las líneas de código a 15.

1.-

```
public void EmaitzakIpini(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();

    Vector<Apustua> listApustuak = q.getApustuak();
    db.getTransaction().begin();
    Question que = q.getQuestion();
    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");
            }
        }
    }
    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 EmaitzakIpini(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();

        Vector<Apustua> listApustuak = q.getApustuak();
        db.getTransaction().begin();
        Question que = q.getQuestion();
        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");
                        }
                }
        }
        db.getTransaction().commit();
        actualizarApuestas(listApustuak);
  }

  /**
   * @param listApustuak
```

```
 */
private void actualizarApuestas(Vector<Apustua> listApustuak) {
        for(Apustua a : listApustuak) {
                db.getTransaction().begin();
                Boolean bool=a.getApustuAnitza().irabazitaMarkatu();
                db.getTransaction().commit();
                if(bool) {
                        this.ApustuaIrabazi(a.getApustuAnitza());
                }
        }
}
```

2.-

```
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;
        }else if(new Date().compareTo(event.getEventDate())<0) {
                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 ap1 = db.find(ApustuAnitza.class,
apustuAnitza.getApustuAnitzaNumber());
                                db.getTransaction().begin();
                                ap1.removeApustua(quo.getApustuak().get(i));
```

```
                                        db.getTransaction().commit();
                                        if(ap1.getApustuak().isEmpty() &&
!ap1.getEgoera().equals("galduta")) {
                                                this.apustuaEzabatu(ap1.getUser(), ap1);
                                        }else if(!ap1.getApustuak().isEmpty() &&
ap1.irabazitaMarkatu()){
                                                this.ApustuaIrabazi(ap1);
                                        }
                                        db.getTransaction().begin();
                                        Sport spo =quo.getQuestion().getEvent().getSport();
                                        spo.setApustuKantitatea(spo.getApustuKantitatea()-1);
                                        db.getTransaction().commit();
                                }
                        }

                }
                db.getTransaction().begin();
                db.remove(event);
                db.getTransaction().commit();
                return true;
   }
```

```
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;
          }else if(new Date().compareTo(event.getEventDate())<0) {
                    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();
                    resoluciónApuestas(listQUO);
```

```
        }
        db.getTransaction().begin();
        db.remove(event);
        db.getTransaction().commit();
        return true;
    }

    /**
     * @param listQUO
     */
    private void resoluciónApuestas(List<Quote> listQUO) {
        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 ap1 = db.find(ApustuAnitza.class,
apustuAnitza.getApustuAnitzaNumber());
                        db.getTransaction().begin();
                        ap1.removeApustua(quo.getApustuak().get(i));
                        db.getTransaction().commit();
                        if(ap1.getApustuak().isEmpty() && !ap1.getEgoera().equals("galduta"))
{
                                this.apustuaEzabatu(ap1.getUser(), ap1);
                        }else if(!ap1.getApustuak().isEmpty() && ap1.irabazitaMarkatu()){
                                this.ApustuaIrabazi(ap1);
                        }
                        db.getTransaction().begin();
                        Sport spo =quo.getQuestion().getEvent().getSport();
                        spo.setApustuKantitatea(spo.getApustuKantitatea()-1);
                        db.getTransaction().commit();
                }
        }
    }
```

"Write simple units of code" (capítulo 3)

1.-

```
public boolean ApustuaEgin(Registered u, Vector<Quote> quote, Double balioa, Integer
apustuBikoitzaGalarazi) {
        Registered user = (Registered) db.find(Registered.class, u.getUsername());
        Boolean b;
        if(user.getDirukop()>=balioa) {
                db.getTransaction().begin();
                ApustuAnitza apustuAnitza = new ApustuAnitza(user, balioa);
                db.persist(apustuAnitza);
                for(Quote quo: quote) {
                        Quote kuote = db.find(Quote.class, quo.getQuoteNumber());
                        Apustua ap = new Apustua(apustuAnitza, kuote);
                        db.persist(ap);
                        apustuAnitza.addApustua(ap);
                        kuote.addApustua(ap);
                }
                db.getTransaction().commit();
                db.getTransaction().begin();
                if(apustuBikoitzaGalarazi==-1) {
                        apustuBikoitzaGalarazi=apustuAnitza.getApustuAnitzaNumber();
                }
                apustuAnitza.setApustuKopia(apustuBikoitzaGalarazi);
                user.updateDiruKontua(-balioa);
                Transaction t = new Transaction(user, balioa, new Date(), "ApustuaEgin");
                user.addApustuAnitza(apustuAnitza);
                for(Apustua a: apustuAnitza.getApustuak()) {
                        Apustua apu = db.find(Apustua.class, a.getApostuaNumber());
                        Quote q = db.find(Quote.class, apu.getKuota().getQuoteNumber());
                        Sport spo =q.getQuestion().getEvent().getSport();
                        spo.setApustuKantitatea(spo.getApustuKantitatea()+1);

                }
                user.addTransaction(t);
                db.persist(t);
                db.getTransaction().commit();
                for(Jarraitzailea reg:user.getJarraitzaileLista()) {
                        Jarraitzailea erab=db.find(Jarraitzailea.class,
reg.getJarraitzaileaNumber());
                        b=true;
```

```
                              for(ApustuAnitza apu: erab.getNork().getApustuAnitzak()) {

if(apu.getApustuKopia().equals(apustuAnitza.getApustuKopia())) {
                                                  b=false;
                                        }
                              }
                              if(b) {
                                        if(erab.getNork().getDiruLimitea()<balioa) {
                                                  this.ApustuaEgin(erab.getNork(), quote,
erab.getNork().getDiruLimitea(), apustuBikoitzaGalarazi);
                                        }else{
                                                  this.ApustuaEgin(erab.getNork(), quote, balioa,
apustuBikoitzaGalarazi);
                                        }
                              }
                    }
                    return true;
          }else{
                    return false;
          }

   }
```

```
public boolean ApustuaEgin(Registered u, Vector<Quote> quote, Double balioa, Integer
apustuBikoitzaGalarazi) {
          Registered user = (Registered) db.find(Registered.class, u.getUsername());
          Boolean b;
          if(user.getDirukop()>=balioa) {
                    db.getTransaction().begin();
                    ApustuAnitza apustuAnitza = new ApustuAnitza(user, balioa);
                    db.persist(apustuAnitza);
                    for(Quote quo: quote) {
                              Quote kuote = db.find(Quote.class, quo.getQuoteNumber());
                              Apustua ap = new Apustua(apustuAnitza, kuote);
                              db.persist(ap);
                              apustuAnitza.addApustua(ap);
                              kuote.addApustua(ap);
                    }
                    db.getTransaction().commit();
                    db.getTransaction().begin();
                    if(apustuBikoitzaGalarazi==-1) {
                        apustuBikoitzaGalarazi=apustuAnitza.getApustuAnitzaNumber();
                }
                apustuAnitza.setApustuKopia(apustuBikoitzaGalarazi);
                user.updateDiruKontua(-balioa);
                Transaction t = new Transaction(user, balioa, new Date(), "ApustuaEgin");
                user.addApustuAnitza(apustuAnitza);
                suficienteDinero(quote, balioa, apustuBikoitzaGalarazi, user, apustuAnitza, t);
                return true;
        }else{
                return false;
        }

   }

/**
     * @param quote
     * @param balioa
     * @param apustuBikoitzaGalarazi
     * @param user
     * @param apustuAnitza
     * @param t
     */
   private void suficienteDinero(Vector<Quote> quote, Double balioa, Integer
apustuBikoitzaGalarazi, Registered user,
                ApustuAnitza apustuAnitza, Transaction t) {
        Boolean b;
        for(Apustua a: apustuAnitza.getApustuak()) {
                Apustua apu = db.find(Apustua.class, a.getApostuaNumber());
                Quote q = db.find(Quote.class, apu.getKuota().getQuoteNumber());
                Sport spo =q.getQuestion().getEvent().getSport();
                spo.setApustuKantitatea(spo.getApustuKantitatea()+1);

        }
        user.addTransaction(t);
        db.persist(t);
        db.getTransaction().commit();
        for(Jarraitzailea reg:user.getJarraitzaileLista()) {
                Jarraitzailea erab=db.find(Jarraitzailea.class, reg.getJarraitzaileaNumber());
                b=true;
                for(ApustuAnitza apu: erab.getNork().getApustuAnitzak()) {
                        if(apu.getApustuKopia().equals(apustuAnitza.getApustuKopia())) {
                                b=false;
                        }
			}
			if(b) {
				if(erab.getNork().getDiruLimitea()<balioa) {
					this.ApustuaEgin(erab.getNork(), quote,
erab.getNork().getDiruLimitea(), apustuBikoitzaGalarazi);
				}else{
					this.ApustuaEgin(erab.getNork(), quote, balioa,
apustuBikoitzaGalarazi);
				}
			}
		}
   }
```

2.-

```
public boolean gertaerakSortu(String description,Date eventDate, String sport) {
		boolean b = true;
		db.getTransaction().begin();
		Sport spo =db.find(Sport.class, sport);
		System.out.println("El deporte existe?"+ spo);
		if(spo!=null) {
			System.out.println("Existe deporte");
			TypedQuery<Event> Equery = db.createQuery("SELECT e FROM Event e
WHERE e.getEventDate() =?1 ",Event.class);
			Equery.setParameter(1, eventDate);
			for(Event ev: Equery.getResultList()) {
				System.out.println("Entra en el for");
				if(ev.getDescription().equals(description)) {
					b = false;
				}
			}
			if(b) {
				String[] taldeak = description.split("-");
				Team lokala = new Team(taldeak[0]);
				Team kanpokoa = new Team(taldeak[1]);
				Event e = new Event(description, eventDate, lokala, kanpokoa);
				e.setSport(spo);
				spo.addEvent(e);
				db.persist(e);
			}
		}
```

```
        else {
                return false;
        }

        db.getTransaction().commit();
        return b;
  }
```

```
public boolean gertaerakSortu(String description,Date eventDate, String sport) {
        boolean b = true;
        db.getTransaction().begin();
        Sport spo =db.find(Sport.class, sport);
        System.out.println("El deporte existe?"+ spo);
        if(spo!=null) {
                System.out.println("Existe deporte");
                TypedQuery<Event> Equery = db.createQuery("SELECT e FROM Event e
WHERE e.getEventDate() =?1 ",Event.class);
                Equery.setParameter(1, eventDate);
                b = existeEvento(description, b, Equery);
                if(b) {
                        String[] taldeak = description.split("-");
                        Team lokala = new Team(taldeak[0]);
                        Team kanpokoa = new Team(taldeak[1]);
                        Event e = new Event(description, eventDate, lokala, kanpokoa);
                        e.setSport(spo);
                        spo.addEvent(e);
                        db.persist(e);
                }
        }
        else {
                return false;
        }

        db.getTransaction().commit();
        return b;
  }

  /**
   * @param description
   * @param b
   * @param Equery
  * @return
  */
 private boolean existeEvento(String description, boolean b, TypedQuery<Event> Equery) {
        for(Event ev: Equery.getResultList()) {
                System.out.println("Entra en el for");
                if(ev.getDescription().equals(description)) {
                        b = false;
                }
        }
        return b;
 }
```

“Duplicate code” (capítulo 4)

1.- KuotakIpiniGUI

```
private JLabel jLabelListOfEvents = new
JLabel(ResourceBundle.getBundle("Etiquetas").getString("ListEvents"));
```

```
private static final String ETIQUETAS = "Etiquetas";
private JLabel jLabelListOfEvents = new
JLabel(ResourceBundle.getBundle(ETIQUETAS).getString("ListEvents"));
```

2.- EmaitzakIpiniGUI

```
private JLabel jLabelListOfEvents = new
JLabel(ResourceBundle.getBundle("Etiquetas").getString("ListEvents"));
```

```
private static final String ETIQUETAS = "Etiquetas";

private JLabel jLabelListOfEvents = new
JLabel(ResourceBundle.getBundle(ETIQUETAS).getString("ListEvents"));
```

"Keep unit interfaces small" (capítulo 5)

En nuestro proyecto no hemos encontrado ningún método con más de cuatro parámetros.

- **Enlace Github**

  Enlace Github

- **Enlace Sonarcloud**

  Enlace Sonarcloud