

Санкт-Петербургский Политехнический Университет Петра Великого  
Институт компьютерных наук и технологий  
Кафедра компьютерных систем и программных технологий

# Программирование

Отчет по курсовой работе  
Приложение "Курсы валют"

**Работу выполнила:**

Власова А.В.

Группа: 23501/4

**Преподаватель:**

Вылегжанина К.Д.

Санкт-Петербург  
2016

# Содержание

<b>1</b>	<b>Проектирование приложения для отслеживания курсов валют</b>	<b>2</b>
1.1	Задание . . . . .	2
1.2	Концепция . . . . .	2
1.3	Минимально работоспособный продукт . . . . .	2
1.4	Выделенные подпроекты . . . . .	2
1.5	Описание интерфейса библиотеки . . . . .	2
1.6	Выводы . . . . .	3
<b>2</b>	<b>Реализация приложения для отслеживания курсов валют</b>	<b>3</b>
2.1	Среда разработки . . . . .	3
2.2	Выделенные классы . . . . .	3
2.3	Примеры работы консольного приложения . . . . .	3
2.4	Пример работы графического приложения . . . . .	5
2.5	Выводы . . . . .	7
<b>3</b>	<b>Процесс обеспечения качества и тестирование</b>	<b>7</b>
3.1	Просмотр кода и демонстрации . . . . .	7
3.2	Тестирование . . . . .	7
3.3	Выводы . . . . .	7
<b>4</b>	<b>Выводы</b>	<b>7</b>
<b>5</b>	<b>Приложение 1</b>	<b>7</b>

# 1 Проектирование приложения для отслеживания курсов валют

В современном мире деньги играют важную роль в жизни каждого человека. Людям, которые планируют выезжать за границу или собираются покупать валюту и размещать ее на вклад, необходимо знать текущий курс той или иной валюты. Отслеживать валютные курсы можно на финансовых телеканалах, на информационных порталах, а также с помощью специальных приложений. В связи с тем, что сейчас информация о курсах валют является востребованной, было решено создать приложение, которое позволило бы пользователям получать данные об основных мировых валютах.

## 1.1 Задание

Разработать приложение, позволяющее пользователям получать текущие курсы валют, отслеживать их изменения за определенный промежуток времени, а также конвертировать мировые валюты.

## 1.2 Концепция

Готовое приложение дает возможность пользователям получать информацию о текущих курсах валют, отслеживать динамику их изменения за указанный период и конвертировать денежные валюты. Для удобства работы пользователя программа оснащена графическим интерфейсом.

## 1.3 Минимально работоспособный продукт

Графическое приложение, предназначенное для получения текущих курсов валют, отслеживания динамики их изменения и конвертирования мировых валют.

## 1.4 Выделенные подпроекты

В процессе проектирования приложения было выделено три подпроекта.

- **Core**  
Библиотека, представляющая бизнес-логику приложения.
- **Console**  
Консольное приложение, предоставляющее возможность взаимодействия пользователя с ядром через консоль.
- **GUI**  
Приложение, предоставляющее пользователю графический интерфейс для взаимодействия с ядром.

## 1.5 Описание интерфейса библиотеки

Интерфейс библиотеки содержит в себе следующие методы:

- Метод, возвращающий курс заданной валюты на сегодня  
**double getExchange(CurrenciesNames name);**
- Метод, возвращающий курс заданной валюты на определенную дату  
**double getExchange(CurrenciesNames name, String date);**
- Метод, возвращающий курсы всех доступных валют на сегодня  
**List<Currency> getAllExchanges();**
- Метод, возвращающий курсы всех доступных валют на заданную дату  
**List<Currency> getAllExchanges(String date);**
- Метод, позволяющий конвертировать валюты  
**double convert(CurrenciesNames originalCurrency, CurrenciesNames finalCurrency, double number);**
- Метод, возвращающий статистику изменения курса валюты за указанный период  
**List<Currency> getStatistics(CurrenciesNames currency, String period);**

## 1.6 Выводы

В данном разделе рассмотрен процесс проектирования приложения для отслеживания курсов валют. Описаны концепция приложения и минимально работоспособный продукт, перечислены выделенные под-проекты, а также описаны методы, входящие в интерфейс библиотеки.

## 2 Реализация приложения для отслеживания курсов валют

### 2.1 Среда разработки

Операционная система: Windows 8  
Среда разработки: IntelliJ IDEA 2016.2.4  
Компилятор: javac, JDK 1.8.0\_102

### 2.2 Выделенные классы

В библиотеке были выделены следующие классы:

- **CurrenciesNames** - перечисление доступных валют.
- **Currency** - содержит основную информацию о валюте. Позволяет получить курс валюты и дату, когда этот курс был действителен, узнать, как изменился курс по сравнению с предыдущим днем, а также получить буквенный код валюты и ее русское название.
- **HTMLParser** - устанавливает соединение с сайтом cbt.ru. Содержит метод, возвращающий курс заданной валюты.
- **Day** - класс для работы с датами. Содержит методы, возвращающие нужную дату в отформатированном виде.
- **ExchangeRatesAPI** - интерфейс библиотеки. Описан в предыдущем разделе.
- **ExchangeRates** - реализует интерфейс библиотеки.

В подпроекте **Console** выделен единственный класс **Application**, обеспечивающий взаимодействие пользователя с ядром через консоль. Содержит в себе методы, позволяющие выводить в консоль курс заданной валюты на сегодня и на заданную дату, курсы всех валют одновременно, а также конвертировать валюты.

Проект **GUI** содержит в себе четыре класса:

- **ExchangeRatesGUI** - окно приложения, содержащее панель кнопок для переключения между панелями, реализованными в других классах.
- **Rates** - панель, на которой вырисовывается таблица с курсами валют.
- **Converter** - панель, содержащая конвертер валют.
- **Statistics** - панель, на которой отображается таблица, описывающая динамику изменения определенной валюты за указанный период.

### 2.3 Примеры работы консольного приложения

Для демонстрации работы консольного приложения ниже представлены снимки экрана работающего приложения.

```
"C:\Program Files\Java\jdk1.8.0_102\bin\java" ...
Введите команду: на сегодня
Введите код валюты: usd
Доллар США
Курс на сегодня: 60.8528
Введите команду:
```

Рис. 1: Вывод курса на сегодня

```
Введите команду: все на сегодня
USD 60.8528
EUR 63.5425
JPY 51.8095
GBP 74.6664
CHF 59.2992
CAD 45.0995
AUD 43.9053
AZN 33.9372
AMD 12.6264
BYN 31.2627
BGN 32.493
```

Рис. 2: Вывод всех курсов на сегодня

```
Введите команду: на дату
Введите код валюты: usd
Введите дату: 23.12.2016
60.8641
Введите команду:
```

Рис. 3: Вывод курса на заданную дату

```
Введите код начальной валюты: eur
Введите код конечной валюты: jpy
Введите сумму: 12
12.0 EUR = 14.72 JPY
```

Рис. 4: Конвертирование валюты

## 2.4 Пример работы графического приложения

Ниже приведены снимки экрана для демонстрации работы графического приложения.







25.12.2016	Валюта	Номинал	Наименование	Курс
	USD	1	Доллар США	60.8528
	EUR	1	Евро	63.5425
	JPY	100	Японская йена	51.8095
	GBP	1	Английский фунт	74.6664
	CHF	1	Швейцарский франк	59.2992
	CAD	1	Канадский доллар	45.0995
	AUD	1	Австралийский доллар	43.9053
	AZN	1	Азербайджанский манат	33.9372
	AMD	100	Армянский драм	12.6264
	BYN	1	Белорусский рубль	31.2627
	BGN	1	Болгарский лев	32.493

Рис. 5: Таблица с курсами

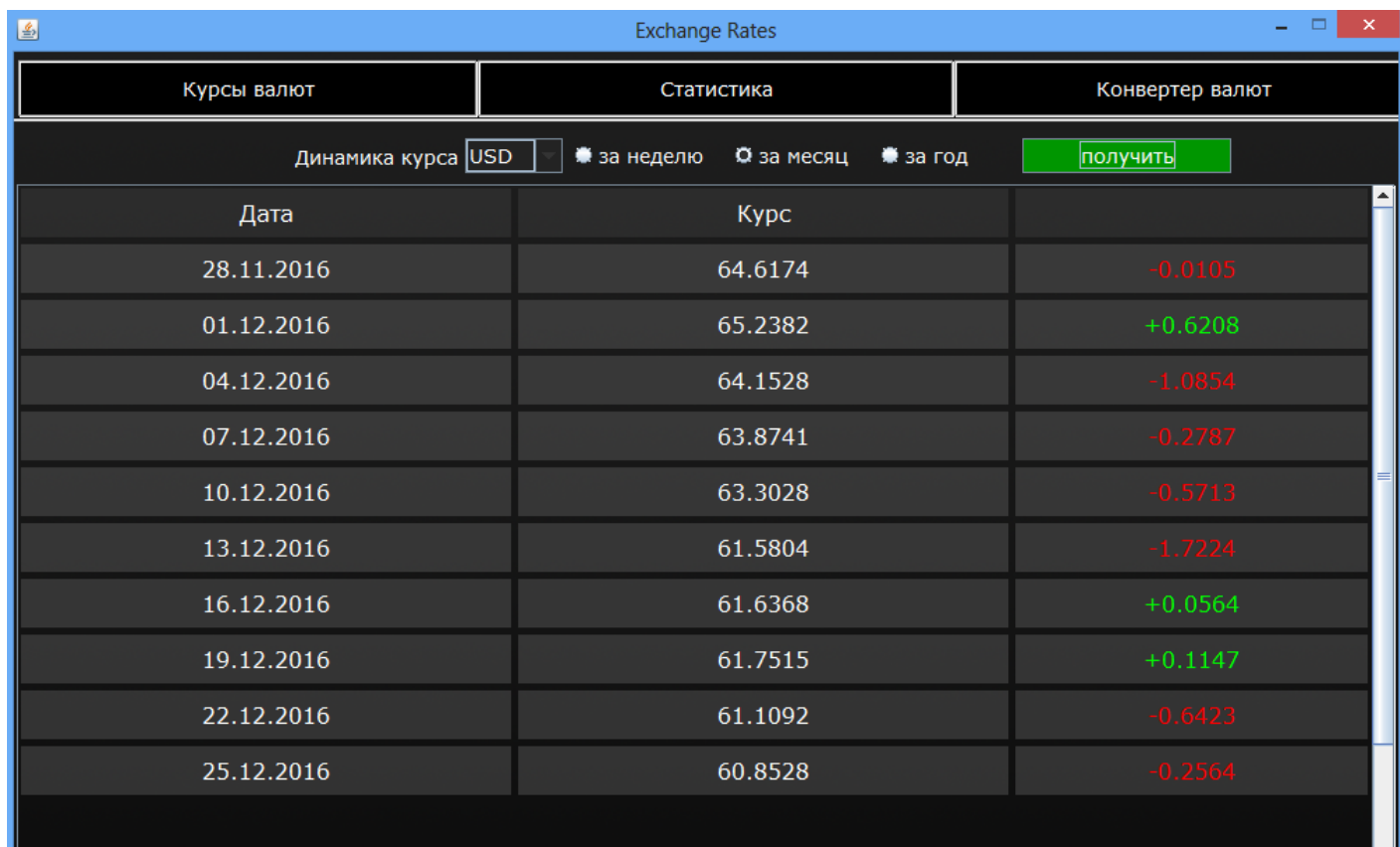


Рис. 6: Изменение курса USD

Exchange Rates

Курсы валют      Статистика      Конвертер валют

Конвертация по курсу ЦБ на сегодня

11.2    GBP    ➔    13.74    USD

Рис. 7: Конвертирование валют

## 2.5 Выводы

В данном разделе были описаны все классы, выделенные в процессе работы над проектом. Также были сделаны снимки экрана, демонстрирующие работу консольного и графического приложений.

## 3 Процесс обеспечения качества и тестирование

### 3.1 Просмотр кода и демонстрации

Для обнаружения ошибок в коде программы один раз был проведен просмотр кода, замечания по которому практически полностью исправлены. Также была осуществлена демонстрация работы графического приложения.

### 3.2 Тестирование

Для проверки работы библиотеки использовались автоматические тесты, покрывающие основную функциональность ядра. Также в процессе разработки приложения проводилось ручное тестирование программы.

### 3.3 Выводы

В данном разделе описаны просмотр кода и демонстрация, проведенные во время разработки приложения, а также процесс тестирования программы.

## 4 Выводы

В результате работы над курсовым проектом было реализовано приложение, предназначенное для получения информации о курсах валют. В процессе создания приложения получены навыки написания программ на языке Java, изучены основные особенности данного языка, а также приобретен опыт работы с библиотекой для создания графических приложений Swing.

## 5 Приложение 1

Листинг 1: ExchangeRatesAPI.java

```
1 package ru.vlasova.exchangeRates.core;
2
3 import java.util.List;
4
5 public interface ExchangeRatesAPI {
6
7     /**
8      * Получить курс валюты на сегодня
9      * @param name название валюты
10     * @return курс валюты
11     */
12     double getExchange(CurrenciesNames name);
13
14     /**
15      * Получить курс валюты на заданную дату
16      * @param name название валюты
17      * @param date дата
18      * @return курс валюты
19      */
20
21     double getExchange(CurrenciesNames name, String date);
22
23     /**
24      * Получить курсы всех валют на сегодня
25      * @return курсы валют
26      */
27     List<Currency> getAllExchanges();
28
29     /**
30      * Получить курсы всех валют на заданную дату
31      * @param date дата
```



```

32     * @return курсы валют
33     */
34
35     List<Currency> getAllExchanges(String date);
36
37     /**
38     * Конвертировать валюту
39     * @param originalCurrency начальная валюта
40     * @param finalCurrency конечная валюта
41     * @param number количество
42     * @return стоимость в конечной валюте
43     */
44     double convert(CurrenciesNames originalCurrency, CurrenciesNames finalCurrency, double
45     ↪ number);
46
47     /**
48     * Получить статистику изменения курса за указанный период
49     * @param currency валюта
50     * @param period период
51     * @return статистика
52     */
53     List<Currency> getStatistics(CurrenciesNames currency, String period);
54 }

```

## Листинг 2: ExchangeRates.java

```

1 package ru.vlasova.exchangeRates.core;
2
3 import java.math.BigDecimal;
4 import java.math.RoundingMode;
5 import java.util.ArrayList;
6 import java.util.List;
7
8 /**
9  * Реализация интерфейса приложения
10  */
11 public class ExchangeRates implements ExchangeRatesAPI {
12
13     @Override
14     public double getExchange(CurrenciesNames name) {
15         Currency currency = new Currency(name, Day.getTodayDate());
16         return currency.getExchange();
17     }
18
19     @Override
20     public double getExchange(CurrenciesNames name, String date) {
21         Currency currency = new Currency(name, date);
22         return currency.getExchange();
23     }
24
25     @Override
26     public List<Currency> getAllExchanges() {
27         List<Currency> allExchanges = new ArrayList<>();
28         for (CurrenciesNames name: CurrenciesNames.values()) {
29             allExchanges.add(new Currency(name, Day.getTodayDate()));
30         }
31         return allExchanges;
32     }
33
34     @Override
35     public List<Currency> getAllExchanges(String date) {
36         List<Currency> allExchanges = new ArrayList<>();
37         for (CurrenciesNames name: CurrenciesNames.values()) {
38             allExchanges.add(new Currency(name, date));
39         }
40         return allExchanges;
41     }
42
43     @Override
44     public double convert(CurrenciesNames originalName, CurrenciesNames finalName, double
45     ↪ number) {
46         Currency originalCurrency = new Currency(originalName, Day.getTodayDate());
47         Currency finalCurrency = new Currency(finalName, Day.getTodayDate());
48         double result = originalCurrency.getExchange() / finalCurrency.getExchange() * number;
49         return (new BigDecimal(result).setScale(2, RoundingMode.HALF_UP).doubleValue());
50     }
51 }

```

```

50
51 @Override
52 public List<Currency> getStatistics(CurrenciesNames name, String period) {
53     List<Currency> statistics = new ArrayList<>();
54     if(period.equals("за_неделю")) {
55         String firstDate = Day.getWeekAgoDate();
56         String lastDate = Day.getTodayDate();
57         statistics.add(new Currency(name, firstDate));
58         while (!firstDate.equals(lastDate)) {
59             firstDate = Day.getNextDate(firstDate);
60             statistics.add(new Currency(name, firstDate));
61         }
62     }
63
64     if(period.equals("за_месяц")){
65         String firstDate = Day.getMonthAgoDate(Day.getTodayDate());
66         String lastDate = Day.getTodayDate();
67         statistics.add(new Currency(name, firstDate));
68         int i = 1;
69         while(!firstDate.equals(lastDate)){
70             firstDate = Day.getNextDate(firstDate);
71             if((i%3)==0)
72                 statistics.add(new Currency(name, firstDate));
73             i++;
74         }
75     }
76
77     if(period.equals("за_год")){
78         String firstDate = Day.getYearAgoDate();
79         String lastDate = Day.getTodayDate();
80         statistics.add(new Currency(name, firstDate));
81         for(int i=0; i<12; i++){
82             firstDate = Day.getMonthPlusDate(firstDate);
83             statistics.add(new Currency(name, firstDate));
84         }
85     }
86     return statistics;
87 }
88 }

```

Листинг 3: CurrenciesNames.java

```

1 package ru.vlasova.exchangeRates.core;
2 //todo странная структура проекта: кажется, что Core должно быть в src/main/java и тд.
3
4 import ru.vlasova.exchangeRates.core.Exceptions.NoSuchCurrencyException;
5
6 /**
7  * Перечисление доступных валют
8  */
9 public enum CurrenciesNames {
10
11     RUB("Российский_рубль"),
12     USD("Доллар_США"),
13     EUR("Евро"),
14     JPY("Японская_йена"),
15     GBP("Английский_фунт"),
16     CHF("Швейцарский_франк"),
17     CAD("Канадский_доллар"),
18     AUD("Австралийский_доллар"),
19     AZN("Азербайджанский_манат"),
20     AMD("Армянский_драм"),
21     BYN("Белорусский_рубль"),
22     BGN("Болгарский_лев"),
23     BRL("Бразильский_реал"),
24     HUF("Венгерский_форинт"),
25     KRW("Вон_Республики_Корея"),
26     DKK("Датская_крона"),
27     KZT("Казахстанский_тенге"),
28     KGS("Киргизский_сом"),
29     CNY("Китайский_юань"),
30     MDL("Молдавский_лей"),
31     TMT("Новый_туркменский_манат"),
32     NOK("Норвежская_крона"),
33     PLN("Польский_злотый"),
34     RON("Румынский_лей"),

```

```

35     SGD( "Сингапурский_доллар" ) ,
36     TJS( "Таджикский_сомони" ) ,
37     TRY( "Турецкая_лира" ) ,
38     UZS( "Узбекский_сум" ) ,
39     UAH( "Украинская_гривна" ) ,
40     CZK( "Чешская_крона" ) ,
41     SEK( "Шведская_крона" ) ,
42     ZAR( "Южноафриканский_рэнд" ) ;
43
44     private String name;
45
46     CurrenciesNames( String name ){
47         this.name = name;
48     }
49
50     static public CurrenciesNames getName( String name ) throws NoSuchCurrencyException {
51         for( CurrenciesNames currency : CurrenciesNames.values() ) {
52             if( currency.toString().equalsIgnoreCase( name ) )
53                 return currency;
54         }
55         throw new NoSuchCurrencyException( "Неизвестная_валюта" );
56     }
57
58     public String getRussianName() {
59         return name;
60     }
61 }

```

Листинг 4: Currency.java

```

1 package ru.vlasova.exchangeRates.core;
2
3 import java.math.BigDecimal;
4 import java.math.RoundingMode;
5
6 /**
7  * Класс валюта
8  */
9 public class Currency {
10
11     private CurrenciesNames name;
12     private double exchange;
13     private String date;
14     private int numberOfUnits;
15
16     public Currency( CurrenciesNames name, String date ) {
17         this.name = name;
18         this.date = date;
19         HTMLParser parser = new HTMLParser( date );
20         exchange = new BigDecimal( parser.getExchangeByName( name ) ).setScale( 4, RoundingMode.
21         ↪ HALF_UP ).doubleValue();
22         numberOfUnits = parser.getNumberOfUnits( name );
23     }
24
25     /**
26     * Получить стоимость валюты
27     * @return стоимость
28     */
29     public double getExchange() {
30         return exchange;
31     }
32
33     /**
34     * Получить буквенный код валюты
35     * @return код
36     */
37     public CurrenciesNames getName() {
38         return name;
39     }
40
41     /**
42     * Получить русское название валюты
43     * @return название
44     */
45     public String getRussianName() {
46         return name.getRussianName();
47     }
48 }

```

```

46     }
47
48     /**
49     * Узнать, увеличилась ли стоимость валюты с предыдущего дня
50     * @return true если курс повысился
51     *         false если курс понизился или( не изменился)
52     */
53
54     public boolean isHigher() {
55         HTMLParser pastDateParser = new HTMLParser(Day.getPastDate(date));
56         return exchange > pastDateParser.getExchangeByName(name);
57     }
58
59     /**
60     * Узнать, уменьшилась ли стоимость валюты с предыдущего дня
61     * @return true если курс понизился
62     *         false если курс повысился или не изменился
63     */
64
65     public boolean isLower() {
66         HTMLParser pastDateParser = new HTMLParser(Day.getPastDate(date));
67         return exchange < pastDateParser.getExchangeByName(name);
68     }
69
70     /**
71     * Получить количество единиц валюты
72     * @return количество единиц
73     */
74
75     public int getNumberOfUnits() {
76         return numberOfUnits;
77     }
78
79     /**
80     * Получить дату
81     * @return дата
82     */
83
84     public String getDate(){
85         return date;
86     }
87
88     /**
89     * Узнать, на сколько курс изменился за день
90     * @return изменение
91     */
92
93     public double getDifference(){
94         HTMLParser pastDateParser = new HTMLParser(Day.getPastDate(date));
95         double difference = exchange - pastDateParser.getExchangeByName(name);
96         return new BigDecimal(difference).setScale(4, RoundingMode.HALF_UP).doubleValue();
97     }
98
99     /**
100    * Узнать, на сколько курс изменился за 3 дня
101    * @return изменение
102    */
103
104    public double get3DaysDifference(){
105        HTMLParser pastDateParser = new HTMLParser(Day.get3PastDate(date));
106        double difference = exchange - pastDateParser.getExchangeByName(name);
107        return new BigDecimal(difference).setScale(4, RoundingMode.HALF_UP).doubleValue();
108    }
109
110    /**
111    * Узнать, на сколько курс изменился за месяц
112    * @return изменение
113    */
114
115    public double getMonthDifference(){
116        HTMLParser pastDateParser = new HTMLParser(Day.getMonthAgoDate(date));
117        double difference = exchange - pastDateParser.getExchangeByName(name);
118        return new BigDecimal(difference).setScale(4, RoundingMode.HALF_UP).doubleValue();
119    }
120
121 }

```

## Листинг 5: Day.java

```

1 package ru.vlasova.exchangeRates.core;
2
3 import java.text.SimpleDateFormat;
4 import java.util.Calendar;
5 import java.util.Date;
6 import java.util.GregorianCalendar;
7 import java.util.StringTokenizer;
8
9 import ru.vlasova.exchangeRates.core.Exceptions.IllegalDateFormatException;
10
11 /**
12  * Класс для работы с датами
13  */
14 public class Day {
15     private static SimpleDateFormat sdf = new SimpleDateFormat("dd.MM.yyyy");
16
17     /**
18      * Получить сегодняшнюю дату
19      * @return дата
20      */
21     public static String getTodayDate() {
22         return sdf.format(Calendar.getInstance().getTime());
23     }
24
25     /**
26      * Получить отформатированную дату
27      * @param date дата
28      * @return отформатированная дата
29      * @throws IllegalDateFormatException при неверно заданной дате
30      */
31     public static String getDate(String date) throws IllegalDateFormatException {
32         try {
33             if(isCorrect(date))
34                 return date;
35             else throw new IllegalDateFormatException("Неверный_формат_даты");
36         } catch(Exception e) {
37             throw new IllegalDateFormatException("Неверный_формат_даты");
38         }
39     }
40
41     /**
42      * Проверить формат даты
43      * @param date дата
44      * @return true если формат верный
45      */
46
47     private static boolean isCorrect(String date) {
48         StringTokenizer tokenizer = new StringTokenizer(date, ".");
49         String day = tokenizer.nextToken();
50         String month = tokenizer.nextToken();
51         String year = tokenizer.nextToken();
52         if(Integer.parseInt(year)<2006 || Integer.parseInt(year)>GregorianCalendar.
53         ↪ getInstance().get(Calendar.YEAR))
54             return false;
55         int intMonth = Integer.parseInt(month);
56         if(Integer.parseInt(month) < 1 || Integer.parseInt(month)>12)
57             return false;
58         int intDay = Integer.parseInt(day);
59         if(intDay<1)
60             return false;
61         int [] days = new int []{31,29,31,30,31,30,31,31,30,31,30,31};
62         for(int i=0; i<days.length; i++){
63             if(intMonth==i && intDay>days[i-1])
64                 return false;
65         }
66         return true;
67     }
68
69     /**
70      * Получить текущий год
71      * @return год
72      */

```

```

73 public static int getYear(){
74     GregorianCalendar calendar = new GregorianCalendar();
75     return calendar.get(Calendar.YEAR);
76 }
77
78 /**
79  * Получить предыдущую дату
80  * @param date дата
81  * @return предыдущая дата
82  * @throws IllegalDateFormatException при неверно заданной дате
83  */
84
85 public static String getPastDate(String date) throws IllegalDateFormatException {
86     try {
87         Calendar calendar = Calendar.getInstance();
88         calendar.setTime(sdf.parse(date));
89         calendar.add(Calendar.DAY_OF_YEAR, -1);
90         return sdf.format(calendar.getTime());
91     } catch (Exception e) {
92         throw new IllegalDateFormatException("Неверный_формат_даты");
93     }
94 }
95
96 public static String get3PastDate(String date) throws IllegalDateFormatException{
97     try {
98         Calendar calendar = Calendar.getInstance();
99         calendar.setTime(sdf.parse(date));
100         calendar.add(Calendar.DAY_OF_YEAR, -3);
101         return sdf.format(calendar.getTime());
102     } catch (Exception e){
103         throw new IllegalDateFormatException("Неверный_формат_даты");
104     }
105 }
106
107 public static String getMonthPlusDate(String date) throws IllegalDateFormatException{
108     int[] days = new int[]{31,29,31,30,31,30,31,31,30,31,30,31};
109     try{
110         Calendar calendar = Calendar.getInstance();
111         calendar.setTime(sdf.parse(date));
112         int month = calendar.get(Calendar.MONTH);
113         if(month==11)
114             calendar.add(Calendar.DAY_OF_YEAR, 31);
115         else
116             calendar.add(Calendar.DAY_OF_YEAR, days[month]);
117         return sdf.format(calendar.getTime());
118     } catch (Exception e){
119         throw new IllegalDateFormatException("Неверный_формат_даты");
120     }
121 }
122
123 /**
124  * Получить вчерашнюю дату
125  * @return вчерашняя дата
126  */
127
128 public static String getYeaterdayDate() {
129     Calendar calendar = Calendar.getInstance();
130     calendar.add(Calendar.DAY_OF_YEAR, -1);
131     return sdf.format(calendar.getTime());
132 }
133 /**
134  * Получить следующую дату
135  * @param date дата
136  * @return следующая дата
137  * @throws IllegalDateFormatException при неверно заданной дате
138  */
139
140 public static String getNextDate(String date) throws IllegalDateFormatException {
141     try {
142         Calendar calendar = Calendar.getInstance();
143         calendar.setTime(sdf.parse(date));
144         calendar.add(Calendar.DAY_OF_YEAR, 1);
145         return sdf.format(calendar.getTime());
146     } catch (Exception e) {
147         throw new IllegalDateFormatException("Неверный_формат_даты");
148     }

```

```

149     }
150
151     public static String getWeekAgoDate() {
152         Calendar calendar = Calendar.getInstance();
153         calendar.add(Calendar.DAY_OF_YEAR, -7);
154         return sdf.format(calendar.getTime());
155     }
156
157     public static String getMonthAgoDate(String date) throws IllegalDateFormatException{
158         try {
159             int [] days = new int [] {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
160             Calendar calendar = Calendar.getInstance();
161             calendar.setTime(sdf.parse(date));
162             int month = calendar.get(Calendar.MONTH);
163             if(month==0)
164                 calendar.add(Calendar.DAY_OF_YEAR, -31);
165             else
166                 calendar.add(Calendar.DAY_OF_YEAR, -days[month - 1]);
167             return sdf.format(calendar.getTime());
168         } catch (Exception e){
169             throw new IllegalDateFormatException("Неверный_формат_даты");
170         }
171     }
172
173     public static String getYearAgoDate(){
174         Calendar calendar = Calendar.getInstance();
175         calendar.add(Calendar.DAY_OF_YEAR, -366);
176         return sdf.format(calendar.getTime());
177     }
178 }

```

Листинг 6: HTMLParser.java

```

1 package ru.vlasova.exchangeRates.core;
2
3 import org.jsoup.Jsoup;
4 import org.jsoup.nodes.Document;
5 import org.jsoup.select.Elements;
6 import ru.vlasova.exchangeRates.core.Exceptions.IllegalDateFormatException;
7
8 /**
9  * Класс для получения курсов валют с сайта cbr.ru
10 */
11 public class HTMLParser {
12     private Elements table;
13
14     public HTMLParser(String date) throws IllegalDateFormatException {
15         String url = "http://www.cbr.ru/currency_base/daily.aspx?date_req=" + Day.getDate(date
16         ↪ );
17         try {
18             Document doc = Jsoup.connect(url).get();
19             table = doc.getElementsByTag("tr");
20         } catch (Exception e) {
21             e.printStackTrace();
22         }
23     }
24
25     /**
26     * Получить строку из таблицы для указанной валюты
27     * @param name название валюты
28     * @return строка
29     */
30     private Elements getElements(CurrenciesNames name) {
31         Elements elements = null;
32         for (int i=1; i<table.size(); i++) {
33             elements = table.get(i).select("td");
34             if (elements.get(1).text().equals(name.toString()))
35                 break;
36         }
37         return elements;
38     }
39
40     /**
41     * Получить курс валюты по ее названию
42     * @param name название валюты
43     * @return курс

```

```

43     */
44     public double getExchangeByName(CurrenciesNames name) {
45         String stringExchange = null;
46         if(name != CurrenciesNames.RUB) {
47             Elements currency = getElements(name);
48             String str = currency.get(4).text();
49             stringExchange = str.replace(',', '.', '');
50         }
51         else {
52             stringExchange = "1";
53         }
54         return Double.valueOf(stringExchange);
55     }
56
57     /**
58     * Получить количество единиц валюты
59     * @param name название валюты
60     * @return количество единиц
61     */
62
63     public int getNumberOfUnits(CurrenciesNames name) {
64         int numberOfUnits;
65         if(name != CurrenciesNames.RUB) {
66             Elements currency = getElements(name);
67             numberOfUnits = Integer.valueOf(currency.get(2).text());
68         }
69         else {
70             numberOfUnits = 1;
71         }
72         return numberOfUnits;
73     }
74 }

```

Листинг 7: IllegalDateFormatException.java

```

1 package ru.vlasova.exchangeRates.core.Exceptions;
2
3 /**
4  * Created by Алина on 29.11.2016.
5  */
6 public class IllegalDateFormatException extends IllegalArgumentException {
7
8     public IllegalDateFormatException(String message) {
9         super(message);
10    }
11 }

```

Листинг 8: NoSuchCurrencyException.java

```

1 package ru.vlasova.exchangeRates.core.Exceptions;
2
3 import java.util.NoSuchElementException;
4
5 /**
6  * Created by Алина on 29.11.2016.
7  */
8 public class NoSuchCurrencyException extends NoSuchElementException {
9
10    public NoSuchCurrencyException(String message) {
11        super(message);
12    }
13 }

```

Листинг 9: CurrencyTest.java

```

1 package ru.vlasova.exchangeRates.core;
2 // todo тесты обычно находятся в src/test/java ....
3
4 import org.junit.Test;
5
6 import static org.junit.Assert.*;
7
8 /**
9  * Created by Алина on 17.10.2016.

```



```

10  */
11  public class CurrencyTest {
12
13      Currency currency1 = new Currency(CurrenciesNames.EUR, "01.01.2016");
14      Currency currency2 = new Currency(CurrenciesNames.USD, Day.getPastDate("01.01.2016"));
15      Currency currency3 = new Currency(CurrenciesNames.JPY, "01.01.2016");
16
17      @Test
18      public void testGetExchange() {
19          assertEquals(79.6395, currency1.getExchange(), 0.00001);
20          assertEquals(72.8827, currency2.getExchange(), 0.00001);
21      }
22
23      @Test
24      public void testGetName() {
25          assertEquals(CurrenciesNames.EUR, currency1.getName());
26          assertEquals(CurrenciesNames.USD, currency2.getName());
27      }
28
29      @Test
30      public void testGetRussianName() {
31          assertEquals("Евро", currency1.getRussianName());
32          assertEquals("Доллар_США", currency2.getRussianName());
33      }
34
35      @Test
36      public void testIsHigher() {
37          assertEquals(false, currency1.isHigher());
38          assertEquals(true, currency2.isHigher());
39      }
40
41      @Test
42      public void testIsLower() {
43          assertEquals(true, currency1.isLower());
44          assertEquals(false, currency2.isLower());
45      }
46
47      @Test
48      public void testGetNumberOfUnits() {
49          assertEquals(1, currency1.getNumberOfUnits());
50          assertEquals(100, currency3.getNumberOfUnits());
51      }
52 }

```

Листинг 10: DayTest.java

```

1  package ru.vlasova.exchangeRates.core;
2
3  import org.junit.Test;
4
5  import static org.junit.Assert.*;
6
7  /**
8   * Created by Алина on 24.10.2016.
9   */
10 public class DayTest {
11
12     @Test
13     public void testGetDate() {
14         assertEquals("01.08.2016", Day.getDate("01.08.2016"));
15         assertEquals("01.01.2016", Day.getDate("01.01.2016"));
16     }
17
18     @Test
19     public void testGetPastDate() {
20         assertEquals("31.07.2016", Day.getPastDate("01.08.2016"));
21         assertEquals("31.12.2015", Day.getPastDate("01.01.2016"));
22     }
23
24     @Test
25     public void testGetNextDateDay() {
26         assertEquals("02.08.2016", Day.getNextDate("01.08.2016"));
27         assertEquals("02.01.2016", Day.getNextDate("01.01.2016"));
28     }
29 }
30

```

```

31     @Test
32     public void testGetYear() {
33         assertEquals(2016, Day.getYear());
34     }
35 }

```

Листинг 11: ExchangeRatesTest.java

```

1 package ru.vlasova.exchangeRates.core;
2
3 import org.junit.Test;
4
5 import static org.junit.Assert.*;
6
7 /**
8  * Created by Алина on 18.10.2016.
9  */
10 public class ExchangeRatesTest {
11
12     ExchangeRates exchangeRates = new ExchangeRates();
13
14     @Test
15     public void testGetExchange() {
16         assertEquals(53.3701, exchangeRates.getExchange(CurrenciesNames.AUD, "01.01.2016"),
17             ↪ 0.00001);
18         assertEquals(72.9299, exchangeRates.getExchange(CurrenciesNames.USD, "01.01.2016"),
19             ↪ 0.00001);
20     }
21
22     @Test
23     public void testGetAllExchanges() {
24         assertEquals(53.3701, exchangeRates.getAllExchanges("01.01.2016").get(CurrenciesNames.
25             ↪ AUD.ordinal()).getExchange(), 0.00001);
26         assertEquals(72.9299, exchangeRates.getAllExchanges("01.01.2016").get(CurrenciesNames.
27             ↪ USD.ordinal()).getExchange(), 0.00001);
28     }
29
30     @Test
31     public void testConvert() {
32         assertEquals(1.09, exchangeRates.convert(CurrenciesNames.EUR, CurrenciesNames.USD, 1)
33             ↪ , 0.00001);
34         assertEquals(1130.8, exchangeRates.convert(CurrenciesNames.EUR, CurrenciesNames.JPY,
35             ↪ 10), 0.00001);
36     }
37 }

```

Листинг 12: HTMLParserTest.java

```

1 package ru.vlasova.exchangeRates.core;
2
3 import org.junit.Test;
4
5 import static org.junit.Assert.*;
6
7 /**
8  * Created by Алина on 17.10.2016.
9  */
10 public class HTMLParserTest {
11
12     @Test
13     public void testGetExchangeByName() {
14         HTMLParser parser = new HTMLParser("01.01.2016");
15         assertEquals(72.9299, parser.getExchangeByName(CurrenciesNames.USD), 0.00001);
16     }
17 }

```

Листинг 13: Application.java

```

1 package ru.vlasova.exchangeRates.console;
2
3 import ru.vlasova.exchangeRates.core.CurrenciesNames;
4 import ru.vlasova.exchangeRates.core.Currency;
5 import ru.vlasova.exchangeRates.core.ExchangeRates;
6

```

```

7 import java.util.InputMismatchException;
8 import java.util.List;
9 import java.util.Scanner;
10
11 /**
12  * Консольное приложение
13  */
14 public class Application {
15     private ExchangeRates exchangeRates = new ExchangeRates();
16     private Scanner in = new Scanner(System.in);
17
18     /**
19      * Считать команду пользователя
20      */
21     public void readCommand() {
22         boolean process = true;
23         while(process) {
24             System.out.print("Введите команду: ");
25             String command = in.nextLine();
26             switch (command.trim()) {
27                 case "на_сегодня":
28                     printToday();
29                     break;
30                 case "все_на_сегодня":
31                     printAllToday();
32                     break;
33                 case "на_дату":
34                     printByDate();
35                     break;
36                 case "все_на_дату":
37                     printAllByDate();
38                     break;
39                 case "перевод":
40                     convert();
41                     break;
42                 case "помощь":
43                     printHelp();
44                     break;
45                 case "выход":
46                     process = false;
47                     break;
48                 default:
49                     System.out.println("Неизвестная команда");
50                     break;
51             }
52         }
53     }
54
55     /**
56      * Вывести курс на сегодня
57      */
58     public void printToday() {
59         try {
60             System.out.print("Введите код валюты: ");
61             CurrenciesNames name = CurrenciesNames.getName(in.nextLine().trim());
62             System.out.println(name.getRussianName());
63             System.out.println("Курс на сегодня: " + exchangeRates.getExchange(name));
64         } catch (Exception e) {
65             System.out.println(e);
66         }
67     }
68
69     /**
70      * Вывести курсы всех валют на сегодня
71      */
72     public void printAllToday() {
73         List<Currency> exchanges = exchangeRates.getAllExchanges();
74         for (int i=1; i<exchanges.size(); i++) {
75             System.out.println(exchanges.get(i).getName() + " " + exchanges.get(i).getExchange
76 ↪ ());
77         }
78     }
79
80     /**
81      * Вывести курс валюты на заданную дату
82      */

```

```

82 public void printByDate() {
83     try {
84         System.out.print("Введите_код_валюты:_");
85         CurrenciesNames name = CurrenciesNames.getName(in.nextLine().trim());
86         System.out.print("Введите_дату:_");
87         String date = in.nextLine().trim();
88         System.out.println(exchangeRates.getExchange(name, date));
89     } catch (Exception e) {
90         System.out.println(e);
91     }
92 }
93
94 /**
95  * Вывести курсы всех валют на заданную дату
96  */
97 public void printAllByDate() {
98     try {
99         System.out.print("Введите_дату:_");
100         String date = in.nextLine().trim();
101         List<Currency> exchanges = exchangeRates.getAllExchanges(date);
102         for(int i =1; i<exchanges.size(); i++) {
103             System.out.println(exchanges.get(i).getName() + "_" + exchanges.get(i).
104             ↪ getExchange());
105         } catch (Exception e) {
106             System.out.println(e);
107         }
108     }
109
110 /**
111  * Конвертировать валюту
112  */
113 public void convert() {
114     try {
115         System.out.print("Введите_код_начальной_валюты:_");
116         CurrenciesNames originalName = CurrenciesNames.getName(in.nextLine().trim());
117         System.out.print("Введите_код_конечной_валюты:_");
118         CurrenciesNames finalName = CurrenciesNames.getName(in.nextLine().trim());
119         System.out.print("Введите_сумму:_");
120         double number = in.nextDouble();
121         System.out.println(number + "_" + originalName + "_=" +
122             exchangeRates.convert(originalName, finalName, number) + "_" + finalName);
123     } catch (InputMismatchException e) {
124         System.out.println("Некорректная_сумма");
125     }
126     catch (Exception e) {
127         System.out.println(e);
128     }
129 }
130
131 /**
132  * Вывести описание команд
133  */
134 public void printHelp() {
135     System.out.println("Используйте_следующие_команды:_");
136     System.out.println("----на_сегодня_--_вывести_сегодняшний_курс_заданной_валюты");
137     System.out.println("----все_на_сегодня_--_вывести_сегодняшние_курсы_всех_доступных_валют");
138     System.out.println("----на_дату_--_вывести_курс_валюты_на_заданную_дату");
139     System.out.println("----все_на_дату_--_вывести_курсы_всех_доступных_валют_на_заданную_дату");
140     System.out.println("----перевод_--_конвертировать_валюты");
141     System.out.println("----выход_--_выйти_из_приложения");
142     System.out.println("На_данный_момент_доступны_следующие_валюты:_");
143     for(CurrenciesNames name: CurrenciesNames.values()) {
144         System.out.println("----" + name + "_" + name.getRussianName());
145     }
146 }
147
148 public static void main(String[] args) {
149     Application app = new Application();
150     app.readCommand();
151 }
152 }

```