Дата API

Java 8 содержит совершенно новый API даты и времени в пакете java.time. Следующие примеры охватывают наиболее важные части этого нового API.

Класс Clock

Clock обеспечивает доступ к текущей дате и времени. Объект класса Clock знает о временной зоне и может быть использован вместо System.currentTimeMillis() , чтобы получить текущее время в миллисекундах. Такая мгновенная точка на временной линии также представлена классом Instant. Объект класса Instant может быть использован для создания традиционных объектов класса java.util.Date.

```
Clock clock = Clock.systemDefaultZone();
long millis = clock.millis();
Instant instant = clock.instant();
Date legacyDate = Date.from(instant);   // legacy java.util.Date
```

Часовые пояса (Timezones)

Часовые пояса представлены в ZoneId. Они могут быть легко получены с помощью статических фабричных методов. Часовые пояса определяют смещения между моментами, которые являются важными для преобразований между мгновениями (Instant) и локальными значениями даты и времени.

```
System.out.println(ZoneId.getAvailableZoneIds());
// prints all available timezone ids

ZoneId zone1 = ZoneId.of("Europe/Berlin");
ZoneId zone2 = ZoneId.of("Brazil/East");
System.out.println(zone1.getRules());
System.out.println(zone2.getRules());

// ZoneRules[currentStandardOffset=+01:00]
// ZoneRules[currentStandardOffset=-03:00]
```

LocalTime

LocalTime представляет время без учета часового пояса, например, 10pm или 17:30:15. В следующем примере создаются два объекта LocalTime для часовых поясов, указанных выше. Затем мы сравниваем эти объекты и вычисляем разность в часах и минутах между этими моментами времени.

```
LocalTime now1 = LocalTime.now(zone1);
LocalTime now2 = LocalTime.now(zone2);

System.out.println(now1.isBefore(now2));  // false

long hoursBetween = ChronoUnit.HOURS.between(now1, now2);
long minutesBetween = ChronoUnit.MINUTES.between(now1, now2);

System.out.println(hoursBetween);       // -3
System.out.println(minutesBetween);     // -239
```

LocalTime содержит разнообразные фабричные методы для упрощения создания новых экземпляров, включая конвертацию из строки.

```
LocalTime late = LocalTime.of(23, 59, 59);
System.out.println(late);       // 23:59:59

DateTimeFormatter germanFormatter =
        DateTimeFormatter.ofLocalizedTime(FormatStyle.SHORT)
        .withLocale(Locale.GERMAN);

LocalTime leetTime = LocalTime.parse("13:37", germanFormatter);
System.out.println(leetTime);   // 13:37
```

LocalDate

Объект LocalDate представляет конкретную дату, например, 2014-03-11. Он неизменяем и работает аналогично LocalTime. Пример демонстрирует вычисление новых дат с помощью добавления или вычитания дней, месяцев или лет. Необходимо иметь ввиду, что каждая такая манипуляция возвращает новый объект.

```
LocalDate today = LocalDate.now();
LocalDate tomorrow = today.plus(1, ChronoUnit.DAYS);
LocalDate yesterday = tomorrow.minusDays(2);

LocalDate independenceDay = LocalDate.of(2014, Month.JULY, 4);
DayOfWeek dayOfWeek = independenceDay.getDayOfWeek();
System.out.println(dayOfWeek);    // FRIDAY
```

Получить объект LocalDate из строки так же просто, как и LocalTime

```
DateTimeFormatter myFormatter =
    DateTimeFormatter.ofLocalizedDate(FormatStyle.FULL)
    .withLocale(Locale.getDefault());

LocalDate programmersDay = LocalDate.parse("12 сентября 2016 г.", myFormatter);
System.out.println(programmersDay);
```

Использование форматирования даты-времени:

Класс DateTimeFormatter используется для вывода и разбора объектов даты и времени.

Получить DateTimeFormatter можно тремя способами:

- С использованием предопределенных констант, типа ISO_LOCAL_DATE
- С использованием шаблонных букв типа uuuu-MMM-dd
- С использованием локализованных стилей, таких, как long или medium

Основные классы даты-времени предоставляют два метода – один для форматирования, т.е. преобразования даты-времени в строку – `format(DateTimeFormatter formatter)`, и еще один – для разбора, т.е. получения даты-времени из строки,
`parse(CharSequence text, DateTimeFormatter formatter)`.

```
  LocalDate date = LocalDate.now();
  String text = date.format(formatter);
  LocalDate parsedDate = LocalDate.parse(text, formatter);
```

Предопределенные стили форматирования

Стиль форматирования	Описание	Пример
`ofLocalizedDate(dateStyle)`	Стиль даты, соотв. локали	'2011-12-03'
`ofLocalizedTime(timeStyle)`	Стиль времени, соотв.локали	'10:15:30'
`ofLocalizedDateTime(dateTimeStyle)`	Стиль даты и времени, соотв. локали	'3 Jun 2008 11:05:30'
`ofLocalizedDateTime( dateStyle,timeStyle)`	Стиль даты и стиль времени, соотв.локали	'3 Jun 2008 11:05'
`BASIC_ISO_DATE`	Дата в стиле ISO	'20111203'
`ISO_LOCAL_DATE`	Локальная ISO дата	'2011-12-03'
`ISO_OFFSET_DATE`	ISO дата со смещением	'2011-12-03+01:00'
`ISO_DATE`	ISO дата со смещением или без него	'2011-12-03+01:00'; '2011-12-03'
`ISO_LOCAL_TIME`	Время без смещения	'10:15:30'
`ISO_OFFSET_TIME`	Время со смещением	'10:15:30+01:00'
`ISO_TIME`	Время со смещением или без него	'10:15:30+01:00'; '10:15:30'
`ISO_LOCAL_DATE_TIME`	Локальные дата и время ISO	'2011-12-03T10:15:30'
`ISO_OFFSET_DATE_TIME`	Дата и время со смещением	2011-12-03T10:15:30+01:00'

Шаблоны для форматирования и разбора

Шаблоны основаны на простых последовательностях букв и символов. Шаблон используется для создания объекта форматирования с использованием методов ofPattern(String) и ofPattern(String, Locale). Например, "d MMM uuuu" будет форматировать 2016-09-12 как '12 сен 2016'. Объект форматирования, созданный из шаблона, может быть использован столько раз, сколько нужно, поскольку он неизменяем и потокобезопасен.

Таблица значений шаблонных символов (как в описании)

Symbol	Meaning	Presentation	Examples
G	era	text	AD; Anno Domini; A
u	year	year	2004; 04
y	year-of-era	year	2004; 04
D	day-of-year	number	189
M/L	month-of-year	number/text	7; 07; Jul; July; J
d	day-of-month	number	10
Q/q	quarter-of-year	number/text	3; 03; Q3; 3rd quarter
Y	week-based-year	year	1996; 96
w	week-of-week-based-year	number	27
W	week-of-month	number	4
E	day-of-week	text	Tue; Tuesday; T
e/c	localized day-of-week	number/text	2; 02; Tue; Tuesday; T
F	week-of-month	number	3
a	am-pm-of-day	text	PM
h	clock-hour-of-am-pm (1-12)	number	12
K	hour-of-am-pm (0-11)	number	0
k	clock-hour-of-am-pm (1-24)	number	0
H	hour-of-day (0-23)	number	0
m	minute-of-hour	number	30
s	second-of-minute	number	55
S	fraction-of-second	fraction	978
A	milli-of-day	number	1234
n	nano-of-second	number	987654321
N	nano-of-day	number	1234000000
V	time-zone ID	zone-id	America/Los_Angeles; Z; -08:30
z	time-zone name	zone-name	Pacific Standard Time; PST
O	localized zone-offset	offset-O	GMT+8; GMT+08:00; UTC-08:00;
X	zone-offset 'Z' for zero	offset-X	Z; -08; -0830; -08:30; -083015; -08:30:15;
x	zone-offset	offset-x	+0000; -08; -0830; -08:30; -083015; -08:30:15;
Z	zone-offset	offset-Z	+0000; -0800; -08:00;
p	pad next	pad modifier	1
'	escape for text	delimiter	
''	single quote	literal	'
[	optional section start		
]	optional section end		
#	reserved for future use		
{	reserved for future use		
}	reserved for future use		