

JAVA

TEMA 09:

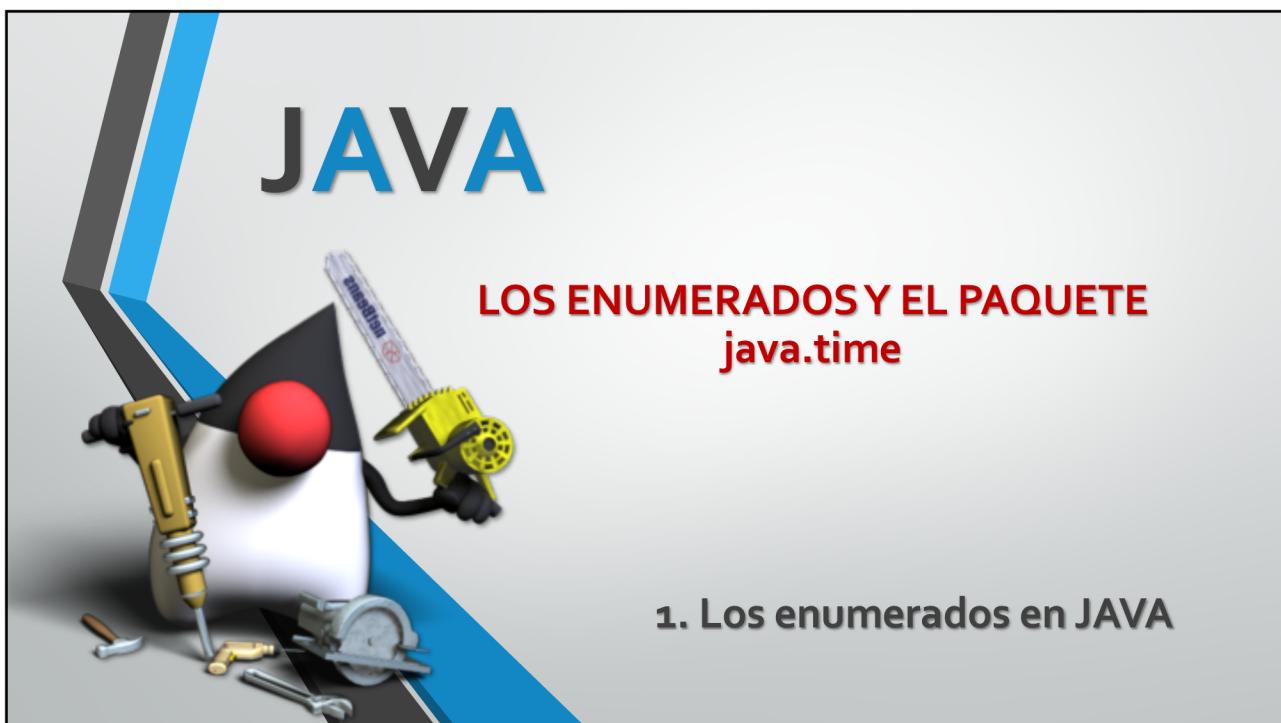
LOS ENUMERADOS Y EL PAQUETE java.time



ÍNDICE

1. Los enumerados en JAVA
2. Introducción al paquete java.time
3. Enumerados de mes y de dia de la semana
4. Las clases del paquete java.time
5. Ejercicios de consolidación





1. Los enumerados en JAVA

Logos for RO (Regional Operative Program), Unión Europea (European Union), and IES Valle del Jerte. A small text box states: "Una manera de hacer Europa". Cofinanciación a cargo del Programa Operativo del FSE 2014-2020 para Extremadura gastos de Ciclos Formativos de Grados Medio y Superior.

1.- Los enumerados en JAVA

- Un enumerado (o Enum) es una clase "especial" que limitan la creación de objetos a los especificados explícitamente en la implementación de la clase.
- La única limitación que tienen los enumerados respecto a una clase normal es que, si tiene constructor, este debe de ser privado para que no se puedan crear nuevos objetos.
- Vamos a ver como añadirlos a nuestro proyecto:

Oscar Laguna García
Ana M. Arribas Arjona

4
PROGRAMACIÓN

UNIÓN EUROPEA
Fondo Social Europeo
Los retos de tu futuro

"Una manera de hacer Europa". Cofinanciación a cargo del Programa Operativo del FSE 2014-2020 para Extremadura gastos de Ciclos Formativos de Grados Medio y Superior.

VIES Valle del Jerte
diferentes perspectivas

1.- Los enumerados en JAVA

Oscar Laguna García
Ana M. Arribas Arjona

PROGRAMACIÓN

5

UNIÓN EUROPEA
Fondo Social Europeo
Los retos de tu futuro

"Una manera de hacer Europa". Cofinanciación a cargo del Programa Operativo del FSE 2014-2020 para Extremadura gastos de Ciclos Formativos de Grados Medio y Superior.

VIES Valle del Jerte
diferentes perspectivas

1.- Los enumerados en JAVA

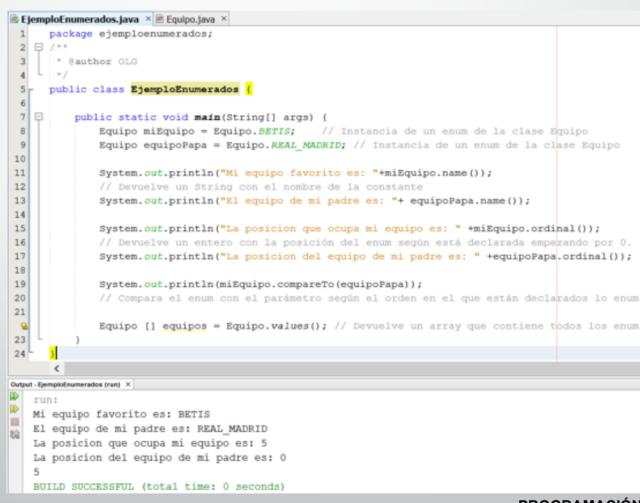
Oscar Laguna García
Ana M. Arribas Arjona

PROGRAMACIÓN

6

1.- Los enumerados en JAVA

- Es muy importante entender que un "Enum" en java es realmente una clase (cuyos objetos solo pueden ser los definidos en esta clase: REAL_MADRID, BARCELONA..., BETIS, VILLAREAL)
- Hereda de la clase **"java.lang.Enum"**, y por tanto los enumerados tienen una serie de métodos heredados de esa clase padre.
- Veamos un ejemplo:



```

1 EjemploNumerados.java x Equipo.java x
2 package ejemploenumerados;
3
4 /**
5  * @author OLG
6  */
7 public class EjemploNumerados {
8
9     public static void main(String[] args) {
10         Equipo miEquipo = Equipo.BETIS; // Instancia de un enum de la clase Equipo
11         Equipo equipoPapa = Equipo.REAL_MADRID; // Instancia de un enum de la clase Equipo
12
13         System.out.println("Mi equipo favorito es: " + miEquipo.name());
14         // Devuelve un String con el nombre de la constante
15         System.out.println("El equipo de mi padre es: " + equipoPapa.name());
16
17         System.out.println("La posicion que ocupa mi equipo es: " + miEquipo.ordinal());
18         // Devuelve un entero con la posición del enum según está declarada empezando por 0.
19         System.out.println("La posicion del equipo de mi padre es: " + equipoPapa.ordinal());
20
21         System.out.println(miEquipo.compareTo(equipoPapa));
22         // Compara el enum con el parámetro según el orden en el que están declarados lo enum
23
24         Equipo [] equipos = Equipo.values(); // Devuelve un array que contiene todos los enum
}

```

Output: EjemploNumerados [run]

- RUN:
- Mi equipo favorito es: BETIS
- El equipo de mi padre es: REAL_MADRID
- La posicion que ocupa mi equipo es: 5
- La posicion del equipo de mi padre es: 0
- 5

BUILD SUCCESSFUL (total time: 0 seconds)

7

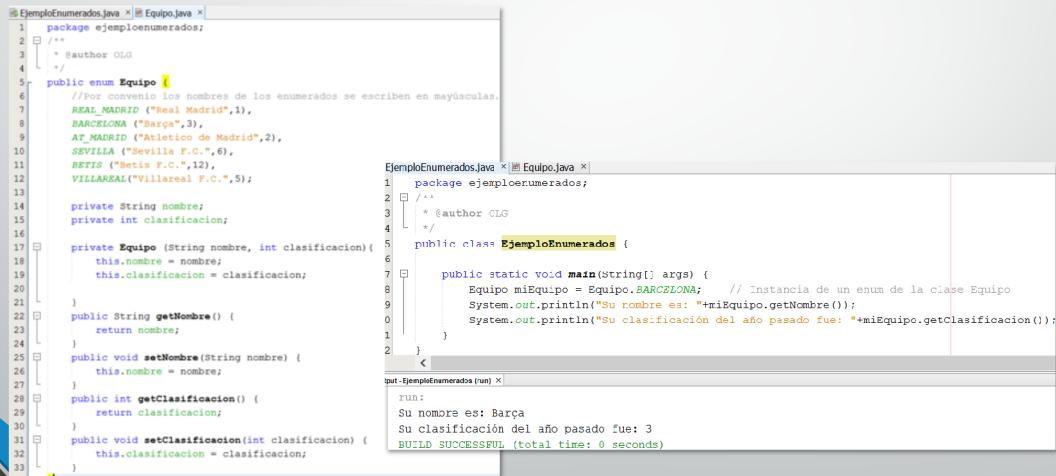
PROGRAMACIÓN

1.- Los enumerados en JAVA

- Como ya se ha dicho, un enum es una clase especial que limita la creación de objetos a los especificados en su clase. Por ello, su constructor es privado, como vamos a ver en el siguiente fragmento de código.
- Además estos objetos pueden tener atributos como cualquier otra clase. En la siguiente declaración de la clase, vemos un ejemplo en la que definimos un enumerado "Equipo" que va a tener dos atributos; el nombre y el puesto en el que quedaron en la liga el año pasado:

8

1.- Los enumerados en JAVA



```

EjemploEnumerados.java | Equipo.java |
1 package ejemploenumerados;
2 /**
3  * @author CLG
4 */
5 public enum Equipo {
6     /**
7      * Por convenio los nombres de los enumerados se escriben en mayúsculas.
8     REAL_MADRID ("Real Madrid",1),
9     BARCELONA ("Barça",3),
10    AT_MADRID ("Atletico Madrid",2),
11    SEVILLA ("Sevilla F.C.",6),
12    BETIS ("Betis F.C.",12),
13    VILLAREAL ("Villareal F.C.",5);
14
15    private String nombre;
16    private int clasificacion;
17
18    private Equipo (String nombre, int clasificacion) {
19        this.nombre = nombre;
20        this.clasificacion = clasificacion;
21    }
22    public String getNombre() {
23        return nombre;
24    }
25    public void setNombre(String nombre) {
26        this.nombre = nombre;
27    }
28    public int getClasificacion() {
29        return clasificacion;
30    }
31    public void setClasificacion(int clasificacion) {
32        this.clasificacion = clasificacion;
33    }
34}

```

```

EjemploEnumerados.java | Equipo.java |
1 package ejemploenumerados;
2 /**
3  * @author CLG
4 */
5 public class EjemploEnumerados {
6
7     public static void main(String[] args) {
8         Equipo miEquipo = Equipo.BARCELONA; // Instancia de un enum de la clase Equipo
9         System.out.println("Su nombre es: "+miEquipo.getNombre());
0         System.out.println("Su clasificación del año pasado fue: "+miEquipo.getClasificacion());
1     }
2 }

```

```

put-EjemploEnumerados (run) |
run:
Su nombre es: Barça
Su clasificación del año pasado fue: 3
BUILD SUCCESSFUL (total time: 0 seconds)

```

 Oscar Laguna García
 Ana M. Arribas Arjona

9

PROGRAMACIÓN

JAVA

LOS ENUMERADOS Y EL PAQUETE `java.time`



2. Introducción al paquete `java.time`

2.- Introducción al paquete java.time

- A partir de la versión Java 8, el manejo de las fechas y el tiempo cambió en Java.
- Desde esta versión, se ha creado una nueva API para el manejo de fechas y tiempo en el paquete `java.time`, que resuelve distintos problemas que se presentaban con el manejo de fechas y tiempo en versiones anteriores (con las clases `java.util.Date`, la clase `java.util.Calendar` y `GregorianCalendar`).

2.- Introducción al paquete java.time

- Por ello, el paquete `java.time`, es el principal API para el manejo de fechas, horas, instantes y duraciones.
- Las clases definidas en este paquete representan los principales conceptos de fecha - hora, incluyendo instantes, fechas, horas, periodos, zonas de tiempo, etc. Están basados en el sistema de calendario ISO, el cual es el calendario mundial *de-facto* que sigue las reglas del calendario Gregoriano

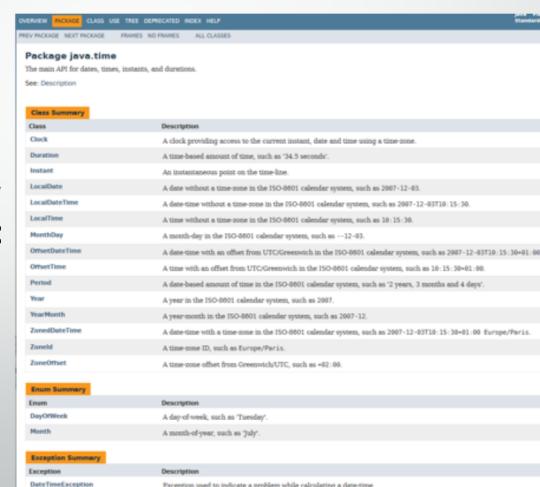
2.- Introducción al paquete java.time

- Algunas de las clases incluidas dentro del paquete java.time para el manejo de tiempos y fechas son: Clock, Duration, Instant, LocalDate, LocalTime, MonthDay, OffsetDateTime, OffsetTime, Period, Year, YearMonth, ZonedDateTime, ZoneId y ZoneOffset).
- Estas clases contienen multitud de métodos para trabajar con fechas. A diferencia de la clase Math, estas clases no tienen métodos estáticos, por lo que para utilizar sus métodos deberemos instanciar un objeto.

13

PROGRAMACIÓN

2.- Introducción al paquete java.time



The screenshot displays the Java API documentation for the `java.time` package. The main content area is titled "Package `java.time`". It starts with a brief description: "The main API for dates, times, instants, and durations." Below this, there's a "See: Description" section. The main focus is the "Class Summary" table, which lists various classes with their descriptions:

Class	Description
<code>Clock</code>	A clock providing access to the current instant, date and time using a time zone.
<code>Duration</code>	A time-based amount of time, such as "34.5 seconds".
<code>Instant</code>	An instantaneous point on the time-line.
<code>LocalDate</code>	A date without a time-zone in the ISO-8601 calendar system, such as 2007-12-03.
<code>LocalDateTime</code>	A date-time without a time-zone in the ISO-8601 calendar system, such as 2007-12-03T10:15:30.
<code>LocalTime</code>	A time without a time-zone in the ISO-8601 calendar system, such as 10:15.
<code>MonthDay</code>	A month-day in the ISO-8601 calendar system, such as -12-31.
<code>OffsetDateTime</code>	A date-time with an offset from UTC/Greenwich in the ISO-8601 calendar system, such as 2007-12-03T10:15:30+01:00.
<code>OffsetTime</code>	A time with an offset from UTC/Greenwich in the ISO-8601 calendar system, such as 10:15:30+01:00.
<code>Period</code>	A date-based amount of time in the ISO-8601 calendar system, such as "2 years, 3 months and 4 days".
<code>Year</code>	A year in the ISO-8601 calendar system, such as 2007.
<code>YearMonth</code>	A year-month in the ISO-8601 calendar system, such as 2007-12.
<code>ZonedDateTime</code>	A date-time with a time-zone in the ISO-8601 calendar system, such as 2007-12-03T10:15:30+01:00 Europe/Paris.
<code>ZoneId</code>	A time-zone ID, such as Europe/Paris.
<code>ZoneOffset</code>	A time-zone offset from Greenwich/UTC, such as +02:00.

Below the class summary, there are sections for "Enum Summary" (listing `DayOfWeek` and `Month`) and "Exception Summary" (listing `DateException` and `ParseException`).

14

PROGRAMACIÓN



"Una manera de hacer Europa". Cofinanciación a cargo del Programa Operativo del FSE 2014-2020 para Extremadura gastos de Ciclos Formativos de Grados Medio y Superior.



3.- Enumerados mes y día de semana

- Existe un enum en `java.time` donde se definen todos los días de la semana, lo cual tiene sentido hacerlo enum porque siempre habrán siete días de la semana.
- Este enum se llama `java.time.DayOfWeek`
- Vamos a ver un ejemplo con alguno de sus métodos más interesantes:

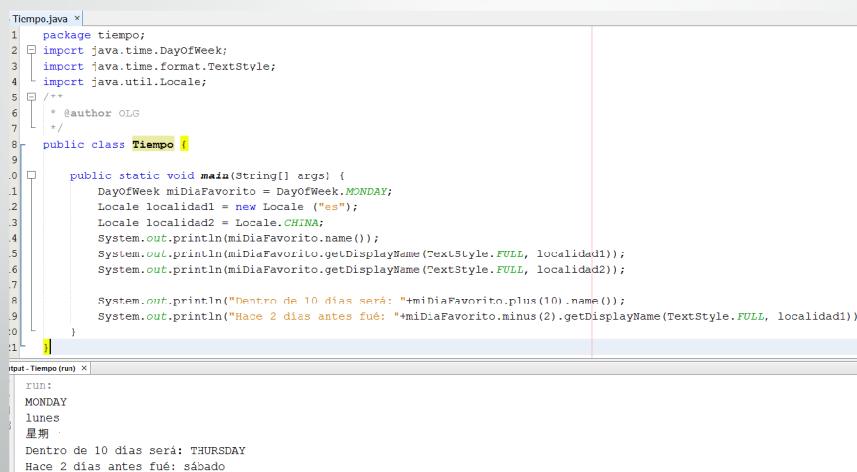
Oscar Laguna García
Ana M. Arribas Arjona

16
PROGRAMACIÓN

 "Una manera de hacer Europa". Cofinanciación a cargo del Programa Operativo del FSE 2014-2020 para Extremadura gastos de Ciclos Formativos de Grados Medio y Superior.

3.- Enumerados mes y día de semana



```

Tiempo.java x
1 package tiempo;
2 import java.time.DayOfWeek;
3 import java.time.format.TextStyle;
4 import java.util.Locale;
5 /**
6 * @author OLG
7 */
8 public class Tiempo {
9
10    public static void main(String[] args) {
11        DayOfWeek miDiaFavorito = DayOfWeek.MONDAY;
12        Locale localidad1 = new Locale("es");
13        Locale localidad2 = Locale.CHINA;
14        System.out.println(miDiaFavorito.name());
15        System.out.println(miDiaFavorito.getDisplayName(TextStyle.FULL, localidad1));
16        System.out.println(miDiaFavorito.getDisplayName(TextStyle.FULL, localidad2));
17
18        System.out.println("Dentro de 10 días será: " + miDiaFavorito.plus(10).name());
19        System.out.println("Hace 2 días antes fue: " + miDiaFavorito.minus(2).getDisplayName(TextStyle.FULL, localidad1));
20    }
21 }

```

Input - Tiempo (run) x

```

RUN:
MONDAY
lunes
星期一
Dentro de 10 días será: THURSDAY
Hace 2 días antes fue: sábado
BUILD SUCCESSFUL (total time: 0 seconds)

```

Oscar Laguna García
Ana M. Arribas Arjona

17

PROGRAMACIÓN

 "Una manera de hacer Europa". Cofinanciación a cargo del Programa Operativo del FSE 2014-2020 para Extremadura gastos de Ciclos Formativos de Grados Medio y Superior.

3.- Enumerados mes y día de semana

- Para los meses, existe el enum **java.time.Month** que básicamente hace lo mismo:

Oscar Laguna García
Ana M. Arribas Arjona

18

PROGRAMACIÓN

3.- Enumerados mes y día de semana

```
Tiempo.java x
1 package tiempo;
2 import java.time.Month;
3 import java.time.format.TextStyle;
4 import java.util.Locale;
5 /**
6 * @author OLG
7 */
8 public class Tiempo {
9     public static void main(String[] args) {
0         Locale localidad = new Locale("pt"); //probamos con portugues
1         Month miMesFavorito = Month.FEBRUARY;
2         System.out.println("Dos meses más y será: " + miMesFavorito.plus(2).name());
3         System.out.println("Hace 1 mes fué: " + miMesFavorito.minus(1).name());
4         System.out.println("Este mes tiene " + miMesFavorito.maxLength() +" dias");
5
6         System.out.println("Mi mes favorito en portugues es: " + miMesFavorito.getDisplayName(TextStyle.FULL, localidad));
7     }
8 }
```

Output - Tiempo (run) x

```
run:
Dos meses más y será: APRIL
Hace 1 mes fué: JANUARY
Este mes tiene 29 dias
Mi mes favorito en portugues es: Fevereiro
BUILD SUCCESSFUL (total time: 0 seconds)
```

 Oscar Laguna García
 Ana M. Arribas Arjona

19

PROGRAMACIÓN

JAVA

LOS ENUMERADOS Y EL PAQUETE java.time



4. Las clases del paquete java.time

4.- Las clases del paquete java.time

- La clase **LocalDate** nos permite establecer **una fecha determinada con la que trabajar** (método **of**) o la fecha actual (método **now**):

```

LocalDate date = LocalDate.of(1986, Month.FEBRUARY, 22);
DayOfWeek dia=date.getDayOfWeek();
System.out.println("El dia que naci fue el "+date+" y era un "+dia);

-fechas (run) X
run:
El dia que naci fue el 1986-02-22 y era un SATURDAY

LocalDate date = LocalDate.now();
DayOfWeek dia=date.getDayOfWeek();
System.out.println("Hoy es "+date+" y es "+dia);

-fechas (run) X
run:
Hoy es 2015-05-25 y es MONDAY

```

21

PROGRAMACIÓN

4.- Las clases del paquete java.time

- Para representar el mes de un año específico, usamos la clase **YearMonth**. Podemos obtener la cantidad de días de ese mes, útil cuando manejamos los años bisiestos:

```

YearMonth mes = YearMonth.now();
System.out.println("Mes actual "+mes+" y tiene "+mes.lengthOfMonth()+" dias");
mes = YearMonth.of(2004, Month.FEBRUARY);
System.out.printf("El mes %s tuvo %d dias. %n", mes, mes.lengthOfMonth());

-fechas (run) X
run:
Mes actual 2015-05 y tiene 31 dias
El mes 2004-02 tuvo 29 dias.

```

22

PROGRAMACIÓN

4.- Las clases del paquete java.time

- La clase **LocalTime** es muy útil para representar horas y tiempos de un día, tales como la hora de inicio de una película o el horario de atención de una biblioteca:

```

LocalTime ahora = LocalTime.now();
System.out.println("En este momento son las "+ahora.getHour()+" horas ");
System.out.println("con "+ahora.getMinute()+" minutos y "+ahora.getSecond()+" segundos");

```

- fechas (run) ×

run:
En este momento son las 18 horas
con 36 minutos y 4 segundos

4.- Las clases del paquete java.time

- La clase **LocalDateTime** es usada para representar la fecha (año, mes, día) junto con la hora (hora, minuto, segundo, nanosegundo), siendo la combinación de LocalDate y LocalTime.

```

LocalDateTime ahora = LocalDateTime.now();
System.out.println("La fecha y hora completa es: "+ahora);
System.out.println("Hace seis meses fue: "+LocalDateTime.now().minusMonths(6));

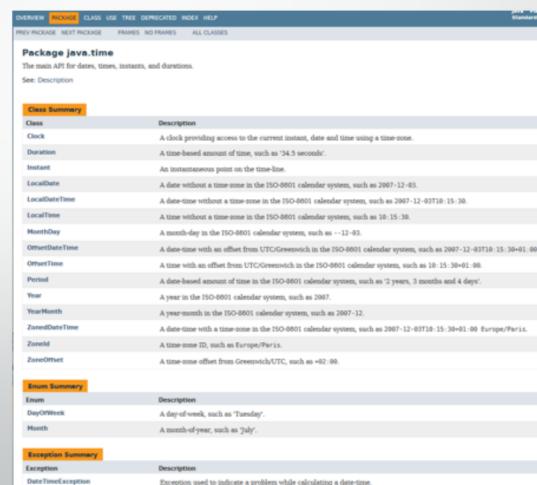
```

- fechas (run) ×

run:
La fecha y hora completa es: 2015-05-25T00:43:04.936
Hace seis meses fue: 2014-11-25T00:43:04.983

4.- Las clases del paquete java.time

- Para más información, consulta la API de JAVA:
<https://docs.oracle.com/javase/8/docs/api/java/time/package-summary.html>



Class	Description
Clock	A clock providing access to the current instant, date and time using a time-zone.
Duration	A time-based amount of time, such as "34.5 seconds".
Instant	An instantaneous point on the time-line.
LocalDate	A date without a time-zone in the ISO-8601 calendar system, such as 2007-12-43.
LocalDateTime	A date-time without a time-zone in the ISO-8601 calendar system, such as 2007-12-03T10:15:30.
LocalTime	A time without a time-zone in the ISO-8601 calendar system, such as 10:15:30.
MonthDay	A month-day in the ISO-8601 calendar system, such as -12-43.
OffsetDateTime	A date-time with an offset from UTC/Greenwich in the ISO-8601 calendar system, such as 2007-12-03T10:15:30+01:00.
OffsetTime	A time with an offset from UTC/Greenwich in the ISO-8601 calendar system, such as 10:15:30+01:00.
Period	A date-based amount of time in the ISO-8601 calendar system, such as "2 years, 3 months and 4 days".
Year	A year in the ISO-8601 calendar system, such as 2007.
YearMonth	A year-month in the ISO-8601 calendar system, such as 2007-12.
YearWeek	A date-time with a time-zone in the ISO-8601 calendar system, such as 2007-12-03T10:15:30+01:00 Europe/Paris.
ZoneId	A time-zone ID, such as Europe/Paris.
ZoneOffset	A time-zone offset from Greenwich/UTC, such as +02:00.

Enum	Description
DayOfWeek	A day-of-week, such as "Tuesday".
Month	A month-of-year, such as "July".

Exception	Description
DateFormatException	Exception used to indicate a problem while calculating a date-time.

25

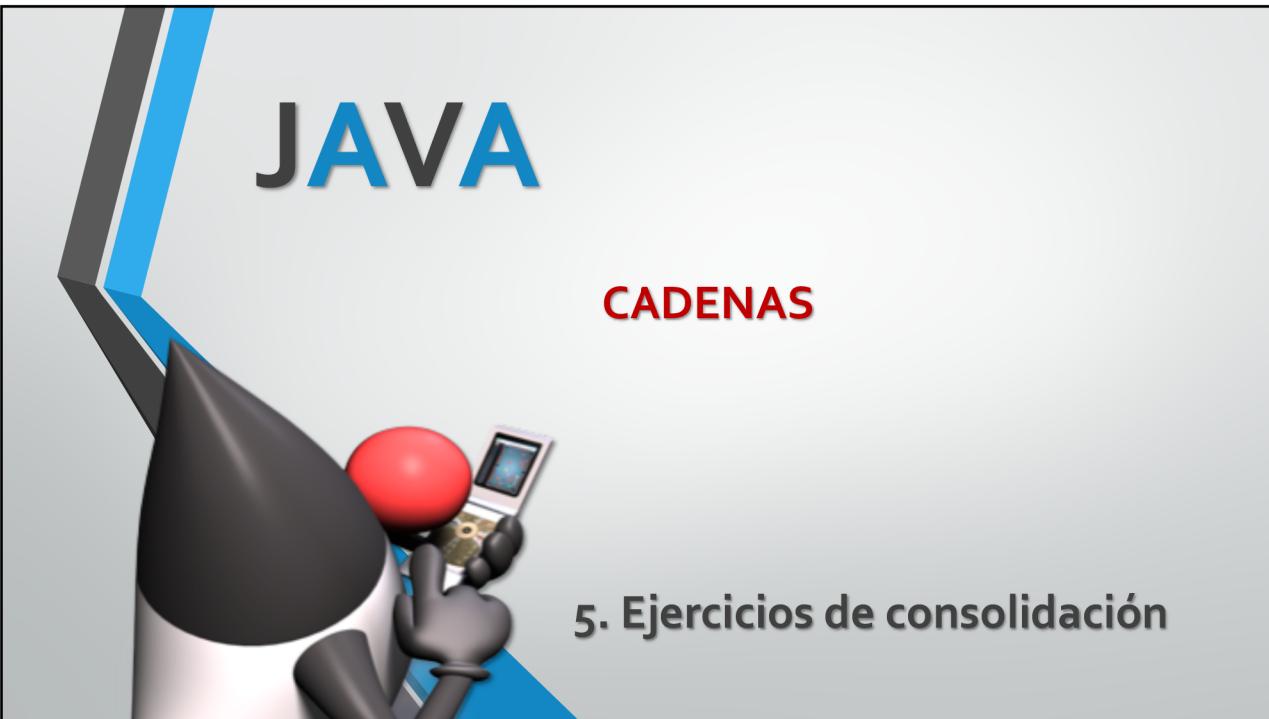
PROGRAMACIÓN

Oscar Laguna García
 Ana M. Arribas Arjona

JAVA

CADENAS

5. Ejercicios de consolidación



EJERCICIOS

- **Ejercicio 01.- (OPTATIVO)** Crea un programa que te pida una fecha de nacimiento y te calcule cual es su signo del zodiaco.
- Pistas: Utiliza un switch y averigua los intervalos de los signos en función del día del año de nacimiento (1 – 365).

EJERCICIOS

- **Ejercicio 02.- (OBLIGATORIO)** Crea un programa que te pedirá tu fecha de nacimiento y te mostrará un menú que te permitirá las siguientes opciones:
 1. Te calcula tu edad exacta en años, meses y días.
 2. Te dice en que día de la semana naciste.
 3. Te dice que estación del año era.
 4. Te dice cuantos días llevas vividos.
 5. Te dice que año podrás/pudiste conducir.
- Recuerda utilizar subprogramas y el paquete java.time
- Pistas:



EJERCICIOS

- **Pistas:**

- *Para calcular el periodo que hay entre dos fechas necesitaras utilizar métodos de la clase **Period**.*
- *Para calcular cuantos días llevas vividos necesitarás utilizar métodos de la clase **ChronoUnit**.*



EJERCICIOS

- **Ejercicio 03.- (OPTATIVO)** Crea un programa que te pida tu edad y la fecha de expedición de tu DNI y te calcule en que fecha tienes que renovarlo, utilizando el paquete `java.time`.
- **PISTA:** Plazos de validez del DNI
- *Con carácter general, el DNI tendrá un período de validez, a contar desde la fecha de la expedición o de cada una de sus renovaciones, de:*
 - *2 años, cuando el niño sea menor de 5 años.*
 - *5 años, cuando el titular no haya cumplido los 30 en el momento de la expedición o renovación.*
 - *10 años, cuando el titular haya cumplido los 30 y no haya alcanzado los 70.*
 - *Permanente, cuando el titular haya cumplido los 70 años.*



UNIÓN EUROPEA
Fondo Social Europeo
"Una manera de hacer Europa". Cofinanciación a cargo del Programa Operativo del FSE 2014-2020 para Extremadura gastos de Ciclos Formativos de Grados Medio y Superior.



EJERCICIOS

- **Ejercicio 04.- (OBLIGATORIO)** Teniendo en cuenta que el instituto abre a las 08:00 de la mañana, y cierra a las 21:20, realiza un programa que, utilizando el paquete *java.time* te calcule:
 1. Cuantas horas y minutos lleva abierto hasta la hora actual, o si ya está cerrado.
 2. Cuantas horas y minutos quedan para que cierre, o si ya está cerrado.