# **Expresiones Reguleras**

## **Objetivo**

Validación de DNI y NIE mediante expresiones regulares.

No te olvides de sonreir: estás programando.

## **Requisitos funcionales**

La solución que programamos hace unos días para chequear si un DNI era válido sólo funciona correctamente para “españoles con documento nacional de identidad (DNI) asignado por el Ministerio del Interior”.

Nos encargan ahora que extendamos el programa para que valide también el NIE o número de identidad extranjero, documento para “extranjeros residentes en España e identificados por la Policía”.

El formato del NIE es:

* Letra X + 7 números + dígito de control
* Letra Y + 7 números + dígito de control
* Letra Z + 7 números + dígito de control

según el [Número de Identificación Fiscal en la Wikipedia.](https://es.wikipedia.org/wiki/N%C3%BAmero_de_identificaci%C3%B3n_fiscal)

Ejemplos:

"X1234567T", "Y1234567T", "Z1234567T"

Además, como nuestros programadores/as vienen de la FP Dual del Borja Moll y practican la mejora e integración continua, han decidido optimizar el código del programa DNI para chequear la validez del DNI o NIE introducido utilizando expresiones regulares.

## **Expresiones regulares**

### **Reglas de escritura**

Primero has de comprender qué es una expresión regular y cuáles son las reglas mediante las cuales se escriben. Luego, has de aprender cómo utilizarlas en Java.

Las expresiones regulares son patrones que seleccionan (filtran) determinadas cadenas de caracteres cuando se aplican sobre un string.

Ejemplos:

[abc] selecciona un caracter que será a, b, o c.

[^xyz] selecciona un carácter excepto x, y, z.

[a-p] selecciona un carácter comprendido entre a y p.

[a-cx-z] selecciona un carácter comprendido entre a y c, o entre x y z, lo cual incluiría a, b, c, x, y, z.

[a-z&&[^aeiou]] selecciona una letra miníscula menos las vocales. En otras palabras, una consonante en minúscula.

[a-zA-Z]+ selecciona cualquier palabra en un string. Por ejemplo: “*Todas las palabras en esta frase se seleccionan*“.

\b\w{4}\b selecciona palabras de 4 letras en un string. Por ejemplo: “drink *beer*, it’s *very* *nice*!”

\b\d{3}[-.]?\d{3}[-.]?\d{4}\b selecciona teléfonos en un string con los formatos: 444-555-1234, 246.555.8888, 1235554567.

* En esta [herramienta web RegExr](http://regexr.com/) puedes testear tus expresiones regulares. Tiene muchos ejemplos. Aquí aprenderás cuáles son sus reglas.

### **Clases Java**

* Has de utilizar las clases Pattern y Matcher.
* En la clase Matcher presta especial atención al método find().
* Los métodos group(), start() y end() de la clase Matcher son muy útiles para nuestros propósitos.

En el capítulo 14 del libro [Beginning Java 8 Fundamentals](https://drive.google.com/open?id=0B68CprMt4M4PREViRUkyNUx2MVU) encontrarás todo lo que necesitas conocer sobre cómo utilizar expresiones regulares en Java, y también sobre sus reglas de escritura.

También puedes leer este [Java Tutorials: Regular Expressions](https://docs.oracle.com/javase/tutorial/essential/regex/intro.html) de Oracle, donde aprenderás a escribir código para utilizarlas en Java (y también sobre sus reglas de escritura).

## **Tu programa**

### **Parte I: validación de la sintaxis de DNI y NIE.**

Escribe un programa Java que, utilizando una (o varias) expresión regular que diseñes, valide si la sintáxis de un DNI o un NIE es correcta. Es decir, has de chequear que:

* El DNI posee 8 dígitos y una letra mayúscula que no sea una I Ñ O U.
* El NIE posee una letra inicial X, Y o Z seguida de 7 dígitos, y una letra final que no sea una I Ñ O U.

NO!! has de chequear si la letra es la que corresponde a ese DNI o NIE según el algoritmo que estudiamos en clase.

1. Utiliza estos casos test DNI correctos (testéalos):

"78484464T","72376173A","01817200Q","95882054E","63587725Q",  
"26861694V","21616083Q","26868974Y","40135330P","89044648X",

1. Genera de manera automática 10 casos test DNI incorrectos y testéalos.
2. Genera de manera automática 10 casos test NIE correctos y testéalos.
3. Genera de manera automática 10 casos test NIE incorrectos y testéalos.

### **Envío del código**

* Envíame el proyecto Eclipse en un correo electrónico, indicando quiénes sois los autores/as.
* Indica también los enlaces a los recursos web que hayas consultado si son distintos a los que te he proporcionado.
* Sube el código a tu cuenta en GitHub y envíame el enlace.

### **Parte II: validación de la letra del DNI/NIE.**

Si dispones de tiempo, refactoriza la clase DniCif del programa DNI en mi [GitHub](https://github.com/dfleta/Java/tree/master/DNI) utilizando lo que has aprendido sobre expresiones regulares, para validar tanto la sintáxis de un DNI /NIE como el dígito de control que le corresponde.

Las reglas de validación del dígito de control de un NIE se encuetran explicadas en el artículo [Número de Identificación Fiscal en la Wikipedia](https://es.wikipedia.org/wiki/N%C3%BAmero_de_identificaci%C3%B3n_fiscal).

Reutiliza la clase TablaAsignacion así como el código que necesites de los casos test en la clase Main.