

**Instituto Tecnológico de Costa Rica**



Actividad 03

Curso: Programación orientada a Objetos

Profesora:

Ing. Ericka Solano Fernández

Grupo #11

Integrantes:

Natifpee Duran Campos

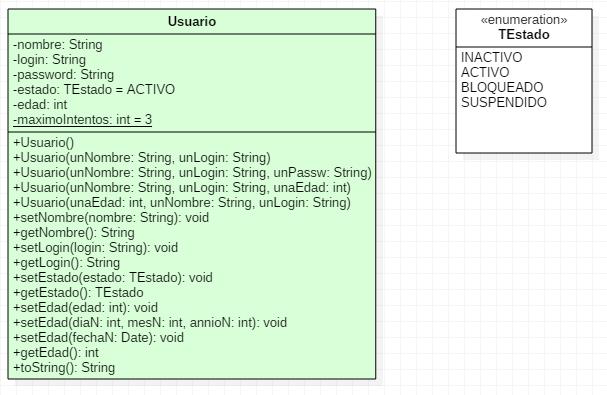
Kevin Salazar Valles

Danny Jiménez Sevilla

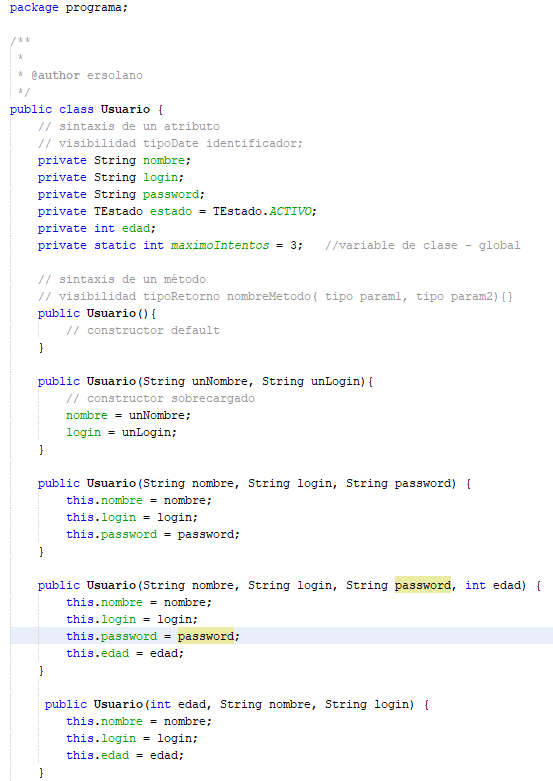
Segundo Semestre, 2021

## Actividad 03: Modelaje en UML y Traducción de un objeto a Java

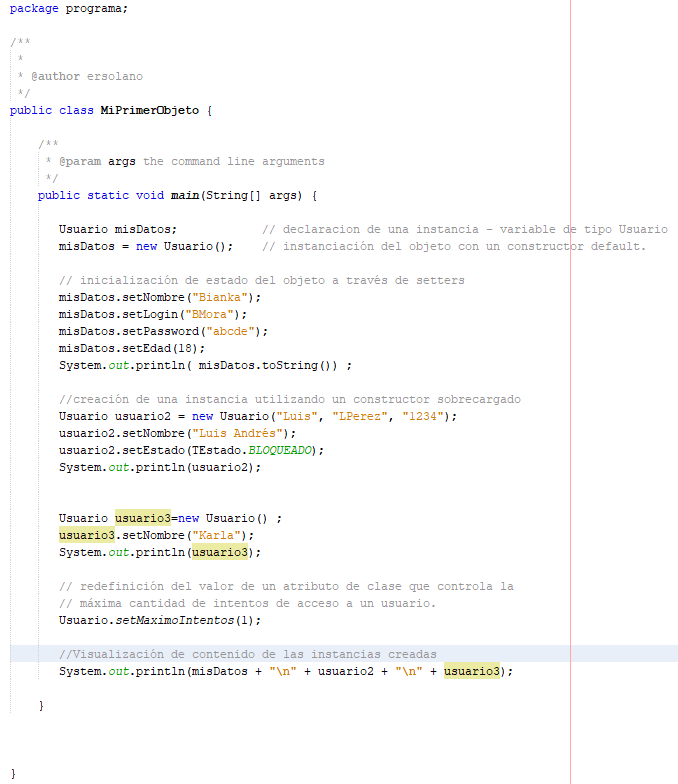
Durante la semana 4 se trabajó en el modelaje de una entidad a la que denominamos **Usuario** y que representamos su estructura (atributos y métodos) a través de un diagrama de clases utilizando la notación UML, en el caso de la clase sincrónica se utilizó StarUML como herramienta. Se agregó además el concepto de tipo de dato enumerado para definir un tipo de dato **TEstado**con valores predeterminados, que establecen una delimitación en el rango de valores posibles que puede adquirir una variable en un contexto dado.



Adicionalmente se produjo la traducción del modelo conceptual al computacional haciendo uso del lenguaje de programación JAVA, utilizando un ambiente de desarrollo integrado, que en la sesión sincrónica se utilizó Netbeans 12.3



Se aportó además un archivo que contiene un programa principal que permitió demostrar la forma de declaración, instanciación y acceso a los servicios públicos ofrecidos por el objeto Usuario a través de métodos constructores, accesores (getters-setters) y toString.



Tanto el proyecto en Java como el diagrama se encuentra en la carpeta del canal General de Teams de este curso.

Se solicita realizar en los grupos de trabajo:

1. En el diagrama se propone la determinación y asignación de la edad del usuario por medio del método sobrecargado *setEdad*, pero sólo fue implementada la versión que recibe un valor entero y se asigna al atributo de manera directa, quedando pendientes otras dos versiones:

|  |  |
| --- | --- |
| setEdad(dia:int, mes:int, annio:int):void | Recibe los datos de dia mes y año de nacimiento en formato de valor para determinar la cantidad de años cumplidos. |
| setEdad(fNac:Date):void | Recibe una fecha de nacimiento en formato Date para determinar la cantidad de años cumplidos. |

1. Implementar la lógica que resuelve estas dos formas de definir la edad en años del objeto Usuario. Para el segundo caso se sugiere investigar sobre los objetos Date y Calendar disponibles en Java como manera de representar una fecha en este lenguaje.
2. Incorporar en el programa principal las instrucciones necesarias para comprobar el funcionamiento correcto de los métodos *setEdad* implementados por el equipo de trabajo.

***Text

Description automatically generated***

***Text

Description automatically generated***

***Text

Description automatically generated***

1. Investigar sobre el concepto de expresiones regulares (regex), que se utilizan por ejemplo, para la comprobación de formatos en datos como hileras de caracteres. En Java existe esta facilidad a través de los objetos Pattern y Matcher.
   1. Describa en no más de una página el funcionamiento y sintaxis de este tipo de servicios y adjunte una imagen que refleje su funcionamiento.

No olvide mencionar las referencias consultadas.

***Aporte aquí lo solicitado***

Las expresiones regulares (o regex) son funciones de búsqueda y reemplazo de contenido textual. “Puede decirse que son el gran subestimado en la edición de texto, puesto que aun ofreciendo una capacidad enorme a la hora de buscar situaciones variables, pocos programadores las utilizan por la complejidad de definir estas expresiones.” (Arteco Consulting SI, 2021)

Las expresiones regulares se usan mediante el paquete de biblioteca de clases java.util.regex, que usa patrones personalizados por las mismas para hacer coincidir cadenas. Incluye dos clases: Pattern y Matcher.

“En java contamos con dos clases importantes para esto:

Pattern es una representación de la expresión regular que ha sido ‘compilada’ para ser utilizada. Es decir, es una expresión que define un patrón de búsqueda para nuestra cadena de caracteres. Matcher es un objeto que surge a partir del patrón (Pattern) previamente creado al cual le pasamos la cadena en dónde deseamos realizar la búsqueda de nuestra expresión.” (Peiretti, S.f)

La clase Pattern es usada para crear una expresion regular, estas se crean en un método de construcción privado.

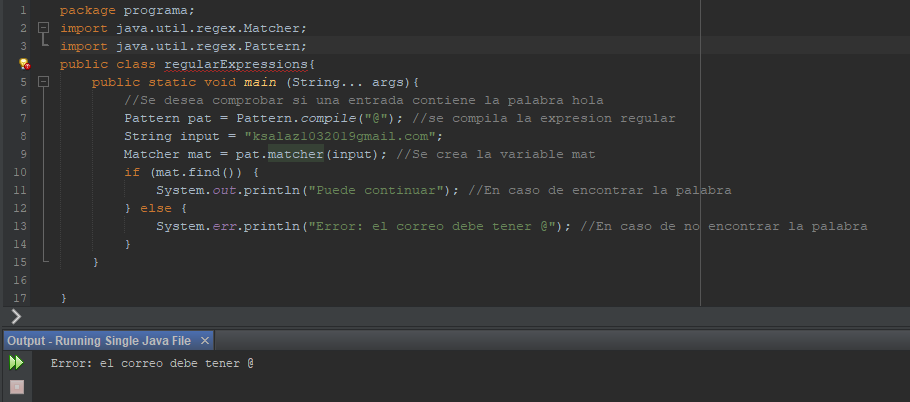
Sintaxis de creación: Pattern nombre = Pattern.compile (String regex)

La clase Matcher proporciona soporte de agrupación para expresiones regulares y soporte de coincidencia múltiple para expresiones regulares.

Sintaxis de creación: Pattern x = Pattern.compile(String regex);

String parametro = “… “

Matcher y = x.matcher(Parametro);

Ejemplo básico del uso de las funciones Pattern y Matcher (Hecho en Netbeans)

Bibliografía

Anónimo. (S.f). Programmerclick. Recuperado el 23 de Agosto de 2021, de <https://programmerclick.com/article/2720823067/>

Arteco Consulting SI. (10 de Agosto de 2021). Arteco Consulting. Recuperado el 23 de Agosto de 2021, de [www.arteco-consulting.com/tutorial-java-regexp/](http://www.arteco-consulting.com/tutorial-java-regexp/)

Peiretti, G. (S.f). GustavoPeiretti. Recuperado el 23 de Agosto de 2021, de <https://gustavopeiretti.com/introduccion-usando-pattern-y-matcher/>

1. Incluya dos atributos nuevos en el objeto Usuario que permitan describir un correo electrónico y un número celular de contacto representados como hileras de caracteres.

El correo deberá cumplir con las reglas de construcción de una dirección electrónica de correo y el celular deberá tener únicamente 8 dígitos numéricos separados por un guión (-) en dos grupos de 4 valores.

* 1. Refleje su cambio en el modelo de UML tanto a nivel de atributos como de métodos accesores.

***Diagram

Description automatically generated with low confidence***

* 1. Defina los atributos y cree los métodos accesores setters y getters en el proyecto Java donde el equipo decida construir el ejercicio (Netbeans, Visual Code Studio, IntelliJ, etc), pero en el caso de los métodos para asignar el correo y el celular, lleve a cabo la comprobación necesaria dentro del método, para saber si el parámetro recibido cumple con el formato adecuado, en caso de ser correcto se procede a asignar el valor, de lo contrario se asigna la expresión “N/A”.

***Text

Description automatically generated***

* 1. Agregue los atributos dentro del método toString() del objeto Usuario.

***Text

Description automatically generated***

1. Incorpore en el programa principal las instrucciones necesarias para comprobar el funcionamiento correcto de los métodos accesores *setCorreo y setCelular* implementados por el equipo de trabajo.

***Text

Description automatically generated***

Este documento modificado por el equipo de trabajo debe ser entregado de manera **grupal** en la **actividad 03** del TEC Digital, junto con el proyecto donde se construyó el objeto Usuario y su archivo de prueba en un archivo comprimido denominado **Act03\_EquipoX\_ObjetoUsuario.zip,** donde X es el número de equipo conformado y debe entregarse a más tardar el martes 24 de agosto al final del día.