**Universidad Catolica De Honduras**

**Nuestra Señora de la Paz**

**Campus San Isidro**

**Catedratico**

**ING Anthony Gainor**

**Alumno**

**David Delcid 0101199702515**

**Asignatura**

**Programacion Orientada A Objetos 18:01**

**Trabajo Practico**

**La Ceiba Atlantida 1 de Noviembre 2017**

**Expresiones Regulares**

Una expresión regular define un patrón de búsqueda para cadenas de caracteres.La clase Pattern (segun la documentación del jdk1.4) es la representación compilada de una expresion regular, o lo que es lo mismo, representa a la expresión regular, que en el paquete java.util.regex necesita estar compilada. En castellano significa patrón.

La clase Matcher es un tipo de objeto que se crea a partir de un patrón mediante la invocación del método Pattern.matcher. Este objeto es el que nos permite realizar operaciones sobre la secuencia de caracteres que queremos validar o la en la secuencia de caracteres en la que queremos buscar. En castellano lo mas parecido a esto es la palabra encajador.

El paquete java.util.regex esta formado por dos clases, la clase Matcher y la clase Pattern y por una excepción, PatternSyntaxException.

La podemos utilizar para comprobar si una cadena contiene o coincide con el patrón. El contenido de la cadena de caracteres puede coincidir con el patrón 0, 1 o más veces.

**Expresión Regular Usada**

\ parametro \g

no es necesario especificar el parametro \g.

Ejemplo dav lo que significa todas las personas cuyo nombre o apellido contengan con “dav” son validas , la aplicación buscara en cada cadena la subcadena que coincida con el parametro de busqueda mencionado.

**Procedimiento Usado**

public static String buscarNombre(String busqueda) {

String persona = "";

for (int a = 0; a < *lista*.length; a++) {

Pattern valida = Pattern.*compile*(busqueda,Pattern.*DOTALL*);

Matcher comprobacion = valida.matcher(*lista*[a].nombre + " " + *lista*[a].apellido);

if (comprobacion.find()) {

persona += *lista*[a].nombre + " " + *lista*[a].apellido + " " + *lista*[a].edad + "\n";

}

}

return persona;

}

El Primer parametro es búsqueda este recibe el valor del textfield para la búsqueda atreves del método getText().

El segundo parametro es Pattern.DOTALL que habilita Unicode-aware case folding.

Compile es un método que se encarga de representar la expresión regular ya compilada para su posterior uso , además se asigna al objeto valida para tener una referencia mas clara y elegante .

Matcher es una clase que se encarga de tomar la expresión regular compilada para su posterior comparación, con el parametro de busqueda

Luego se usa el metodo find de la clase Matcher, el metodo find va buscando subcadenas dentro de la cadena de caracteres que cumplan el patrón compilado , si encuentra alguna devuelve un valor tipo booleano que puede ser True si cumple con la expresión regular especificada, o False si no cumple

Cuando se detecta que una cadena coincide con el patrón , se acumula junto con otros resultados que también puedan coincidir ejemplo

David Martinez

David Delcid

Estas dos personas tienen el mismo nombre pero distinto apellido , si se manda como criterio de búsqueda ¨Dav¨ la aplicación debería de devolver las dos personas antes mencionadas, para mejor interacción del usuario se ha usado "KeyEvent " se debe establecer un objeto de tipo KeyEvent como parámetro para permitir escuchar cualquier tipo de desencadenador, en este caso se ha usado el desencadenador KeyRealised dándole el efecto al usuario de una búsqueda en tiempo real.

KeyEvent toma cualquier evento del teclado

KeyRealized

Toma en cuenta una accion del teclado cuando se ha presionado una tecla y esta afecta el atributo seleccionado