Práctica 7

1 FICHA TRABAJO

Autor	Sergio Martín Santana
Fecha entrega y defensa	13/01/2016
Tiempo Necesario para realizar el trabajo (Aproximado)	
Realización de trabajo de búsqueda de información	4 h
Codificación del problema	12 h
Realización del informe	2 h
Total	18 h

2 INTRODUCCIÓN

En este informe veremos el uso de dos clases: Observable y Observer, las cuales utilizaremos en un ejemplo practico y sencillo para explicar el funcionamiento de estas. Supongamos que tenemos una clase que queremos que sea monitoreada (Observable) por otras clases (Observer) en la arquitectura de tu programa, y que cuando esta clase monitoreada tenga modificaciones estas sean difundidas a las clases que la observan, observadores. Este problema es resuelto por este par de clases (observable y observer) encontradas en el paquete java.util.* el cual facilita esta tarea, asi que vamos a el ejemplo para entender los observadores en java, comencemos.

Lo primero que haremos antes de empezar a destripar las clases es describir un poco las clases de java que utilizaremos:

Inteface Observer:

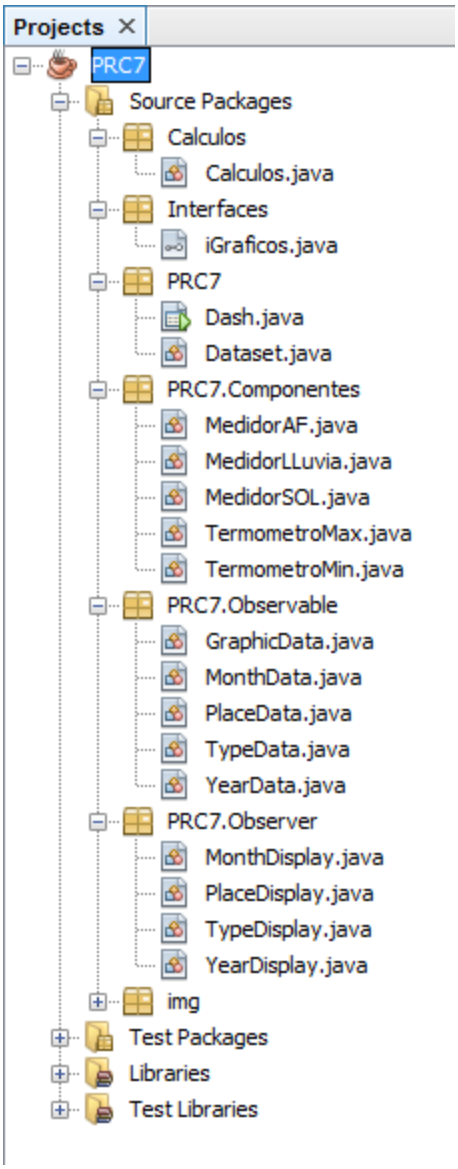
Esta interfaz se implementa para ser notificado de cambios sobre un objeto observable. Esta interfaz luce de la siguiente manera una vez implementada:

```
1
2  package jonathanmelgoza.com.blog;
3
4  import java.util.Observable;
5  import java.util.Observer;
6
7  /**
8   *
9   * @author Jonathan
10  */
11 public class Observador implements Observer{
12
13     @Override
14     public void update(Observable o, Object arg) {
15
16     }
17
18 }
19
```

Clase Observable

Esta clase representa un objeto que puede ser observado o monitoreado de acuerdo al paradigma modelo-vista, representa un objeto que la aplicacion quiere observar y ser notificada de cambios, un objeto observable puede ser observado por uno o mas observadores. Despues de que un objeto observable produce cambios entonces se llama al metodo notifyObservers que causa que todos los observadores sean notificados del cambio llamando a su metodo update. El codigo de la clase que sera monitoreada llamada Observado que hereda de Observable.

```
1
2  package jonathanmelgoza.com.blog;
3
4  import java.util.Observable;
5
6  /**
7   *
8   * @author Jonathan
9   */
10 public class Observado extends Observable{
11     String mensaje;
12
13     public Observado(){
14         mensaje = "Objeto Observado Iniciado";
15     }
16
17     public void cambiarMensaje(String m){
18         mensaje = m;
19         //Marcamos el objeto observable como objeto que ha cambiado
20         setChanged();
21         //Notificamos a los observadores y le enviamos el nuevo valor
22         notifyObservers(mensaje);
23         //notifyObservers(); Este metodo solo notifica que hubo cambios en el objeto
24     }
25 }
26
```



Una vez Entendido en que consiste las clases Observable y Observer, pasaremos a comentar un poco el código que nos repercute en este trabajo.

3 CÓDIGO CREADO

A la izquierda aparece la estructura del proyecto PRC7, que cntiene todas las clases necesarias para realizar el conjunto de funcionalidades requeridas.

Empezaremos describiendo el Paquete Calculos, el cual se compone de algunas funciones de tipo “Static” que no permiten realizar operaciones que bien son necesarias para este proyecto, pero no tanto como para incluirlas como responsabilidad de una clase.

Luego existe un apartado para las interfaces, en el solo existe un interfaz, la iGraficos, que nos permite tener control sobre la expansión de la aplicación, ya que todo elemento gráfico deberá de implementar esta interfaz, lo que nos asegura que no habrá que modificar las clases actuales en caso de querer modificar un gráfico, o añadir uno nuevo.

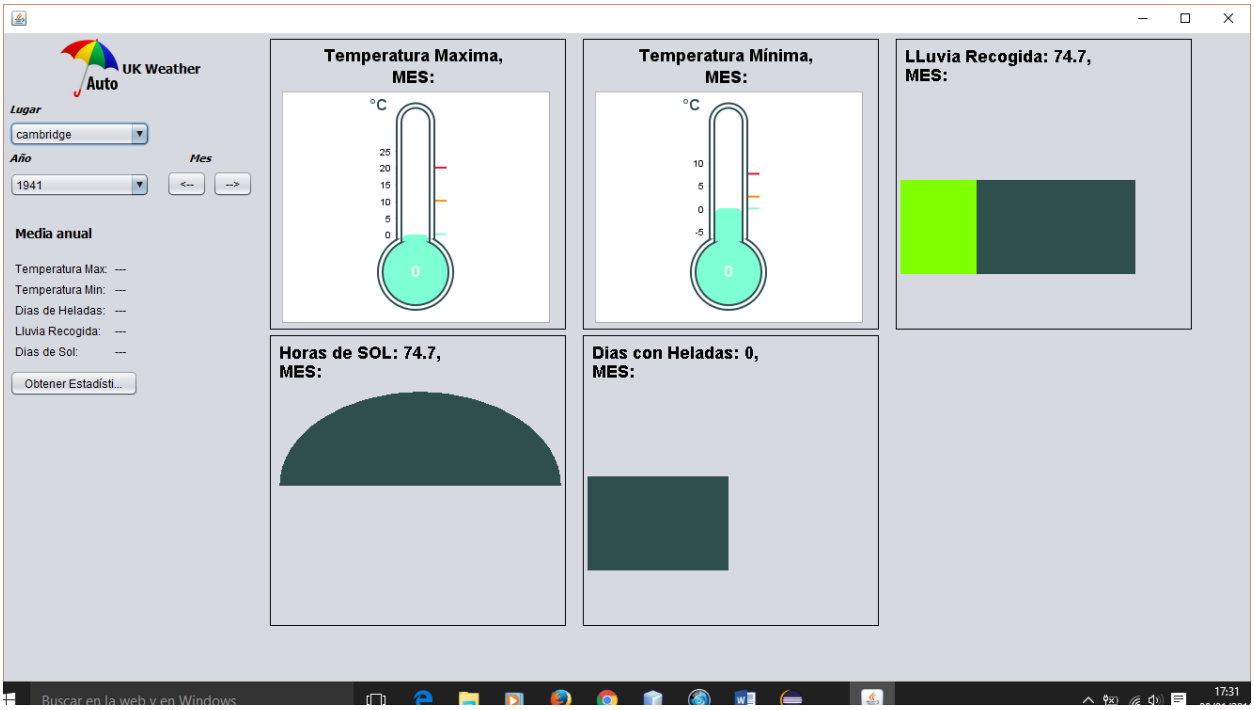
Debajo aparece el paquete principal, donde se recogen la ventana que ejecuta la aplicación y la clase responsable de la carga de conjuntos de datos.

El paquete Componentes, engloba a todos los graficos que implementan iGraficos, y que son los encargados de pintar los diferentes graficos en la pantalla.

Como penúltimo paquete se encuentra el paquete observable, este es uno de los más “imporantes” dentro del proyecto, ya que son uno de los componentes que generan la “magia”. Este grupo de componentes, son los encargados de ser Observados por los observadores, y deberemos tener tantos Observados como componentes observados querramos tener, en nuestro caso hemos determinado que son necesarios 5.

Y por último aparece el paquete Observer que cuenta con los observadores del proyecto, estos componentes se encargan de esperar a que existan cambios en los observadores que tienen adjidicados. En caso de que estos varien realizan una funcion que se encuentra localizada en update(). En nuestro caso, solo el propio dash realiza trabajo funcional en la GUI repitando todos los graficos en cada cambio.

4 INTERFAZ


UK Weather Auto
Lugar
cambridge
Año
1941
Mes
<--
-->
Media anual
Temperatura Max: ---
Temperatura Min: ---
Dias de Heladas: ---
Lluvia Recogida: ---
Dias de Sol: ---
Obtener Estadísti...
Temperatura Maxima, MES:
Temperatura Mínima, MES:
LLuvia Recogida: 74.7, MES:
Horas de SOL: 74.7, MES:
Dias con Heladas: 0, MES:
°C