SpinnerDateModel extends 예제

ere is the code for a JSpinnerModel that handles Date objects in simple format (e.g. ‘dd/MM/yyyy’). It extends a SpinnerDateModel so most of the code is based on that class. This class assumes that values are separated by a slash (/). The reason why I created this was based on a client’s requirement where I could only display the date in simple format and select the calendar using a JSpinner:

/\*\*

\* A SpinnerSimpleDateModel for sequences of formatted Dates.

\* in simple format(e.g. "dd/MM/yyyy")

\*

\* @author William Mora

\* @version 1.0

\*/

public class SpinnerSimpleDateModel extends SpinnerDateModel {

private Comparable start, end;

private Date value;

private int calendarField;

private SimpleDateFormat dateFormat;

private boolean calendarFieldOK(int calendarField) {

switch (calendarField) {

case Calendar.DATE:

case Calendar.YEAR:

case Calendar.MONTH:

return true;

default:

return false;

}

}

/\*\*

\* Creates an instance of SpinnerSimpleDateModel

\*

\* @param value Initial date for the model

\* @param start Minimum date for the model

\* @param end Maximum date for the model

\* @param calendarField The interval desired for next and previous values.

\* The options are:

\* Calendar.DATE - Day of the month

\* Calendar.MONTH - Month

\* Calendar.YEAR - Year

\* @param dateFormat A format specified by an instance of SimpleDateFormat

\*/

public SpinnerSimpleDateModel(Date value, Comparable start, Comparable end,

int calendarField, SimpleDateFormat dateFormat) {

this.dateFormat = dateFormat;

if (value == null) {

throw new IllegalArgumentException("value is null");

}

if (!calendarFieldOK(calendarField)) {

throw new IllegalArgumentException("invalid calendarField");

}

if ( !(((start == null) || (start.compareTo(value) <= 0)) &&

((end == null) || (end.compareTo(value) >= 0)))){

throw new IllegalArgumentException("(start <= value <= end) is false");

}

this.value = value;

this.start = start;

this.end = end;

this.calendarField = calendarField;

}

@Override

public void setStart(Comparable start) {

if ((start == null) ? (this.start != null) : !start.equals(this.start)) {

this.start = start;

fireStateChanged();

}

}

@Override

public void setEnd(Comparable end) {

if ((end == null) ? (this.end != null) : !end.equals(this.end)) {

this.end = end;

fireStateChanged();

}

}

@Override

public void setCalendarField(int calendarField) {

if (!calendarFieldOK(calendarField)) {

throw new IllegalArgumentException("invalid calendarField");

}

if (calendarField != this.calendarField) {

this.calendarField = calendarField;

fireStateChanged();

}

}

@Override

public Comparable getEnd() {

return end;

}

@Override

public Object getNextValue() {

Calendar cal = Calendar.getInstance();

cal.setTime(value);

cal.add(calendarField, 1);

Date next = cal.getTime();

return (end == null) || (end.compareTo(next) >= 0) ? next: null;

}

@Override

public Object getPreviousValue() {

Calendar cal = Calendar.getInstance();

cal.setTime(value);

cal.add(calendarField, -1);

Date prev = cal.getTime();

return ((start == null) || (start.compareTo((prev)) <= 0)) ? prev : null;

}

@Override

public Object getValue() {

return value;

}

@Override

public void setValue(Object value) {

if (value == null) {

throw new NullPointerException("Date is null!");

} else {

if ((value instanceof String) && ((String) value).matches("[0-9]+[/][0-9]+[/][0-9]+")) {

String date = ((String) value);

int day = Integer.parseInt(date.substring(0, 2));

int month = Integer.parseInt(date.substring(3, 5));

int year = Integer.parseInt(date.substring(6));

Calendar cal = Calendar.getInstance();

//Month-1 because is zero-based

cal.set(year, month - 1, day);

this.value = cal.getTime();

}

if (value instanceof Date) {

if (!dateFormat.format((Date) value).equals(dateFormat.format(this.value))) {

String temp = dateFormat.format((Date) value);

this.setValue(temp);

fireStateChanged();

}

}

}

}

}