بسم الله الرحمن الرحيم
نظرة عن قرب في مكتبة الـ JasperReport

ترجمة التقرير:

هذه المرحلة هي أولى مراحل تحويل التقرير إلى شكله النهائي وفيها يتم تحويل قالب التقرير من ملف مكتوب بالـXML إلى ملف التصميم بإمتداد Jasper أو إلي كائن(object) من الفئة JasperReport.

وتتم هذه المرحلة عن طريق الفئة **net.sf.jasperreports.engine.JasperCompileManager**.

هذه الفئة تحتوي علي الطرق الأتية(Methods):-

1. compileReport() **:** تختص بترجمة ملف القالب إلي كائن(object) من الفئة JasperReport.
2. CompileReportToFile: تختص بترجمة ملف قالب التقرير إلي ملف أخر بإمتداد jasper.
3. CompileReportToStream: تختص بترجمة ملف قالب التقرير تيار بيانات (Stream Data).

الطريقة CompileReport:-

كما وضحنا من قبل هذه الطريقة(method) تختص بترجمة ملف القالب إلي كائن(object) من الفئة JasperReport .وهي تعمل عن طريق أخذ مسار ملف القالب أو عن طريق تيار بيانات (inputStream) لملف القالب أو عن طريق كائن من الفئة (JasperDesign) أو مسار الملف في صورة نص لذا فهذه الطريقة محملة (override) بثلاث طرق(methods) .

```
JasperReport jr = JasperCompileManager.compileReport(inputStream);
JasperReport jr =JasperCompileManager.compileReport(jasperDesign);
JasperReport jr =JasperCompileManager.compileReport(sourceFileName);
```

الطريقة compileReportToFile:-

هذه الطريقة تختص بترجمة ملف القالب إلى ملف أخر بإمتداد Jasper. وهي لها ثلاث تحميلات:

1. تأخذ مسار ملف القالب وفي هذه الحالة يكون الملف الناتج في نفس مجلد القالب.
2. تأخذ مسار ملف القالب ومسار الملف الناتج.
3. تاخذ كائن من الفئة JasperDesign ومسار الملف الناتج.

```
JasperCompileManager.compileReportToFile(sourceFileName);
JasperCompileManager.compileReportToFile(jasperDesign, destFileName);
JasperCompileManager.compileReportToFile(sourceFileName, destFileName);
```

الطريقة CompileReportToStream :-

هذه الطريقة تختص بترجمة تيار البيانات لملف القالب (inputStream) أو كائن من الفئة JasperDesign إلى تيار بيانات (outputStream)

```
JasperCompileManager.compileReportToStream(inputStream, outputStream);
JasperCompileManager.compileReportToStream(jasperDesign, outputStream);
```

تعريف الفئة JasperDesign :-

الكائن المنشئ من هذه الفئة يعتبر قالب للتقرير أو المادة الخام للتقرير. ففي اولى مراحل ترجمة يتم

تحويل الملف المكتوب بالـ XML إلي كائن من هذه الفئة ثم إلي كائن من الفئة JasperReport. و يفضل إستخدام هذه الكائنات المنشئة من هذه الفئة مباشرة في حالة ما إذا كنت ستنشئ التقرير في زمن التنفيذ(runtime execution).

ملحوظة : غالبا لن تتعرض لهذه المرحلة لان iReport يقوم بها عنك.

ربط التقرير بمصدر البيانات:-

في هذه المرحلة والتي تعتبر المرحلة الأخيرة قبل عرض التقرير يتم دمج الناتج من المرحلة السابقة مع مصدر البيانات و إنتاج النسخة القابلة للطباعة.تتم هذة المرحلة عن طريق الفئة ((Class **net.sf.jasperreports.engine.JasperFillManager** .

تحتوي هذه الفئة على علي الطرق(methods) التالية:-

الطريقة fillReport :-

هذه الطريقة(method) تختص بملء قالب التقرير المترجم بالبيانات من مصدر البيانات و تحوله إلي كائن من الفئة JasperPrint.

```
JasperPrint jr =
        JasperFillManager.fillReport(InputStream inputStream,Map parameters);
JasperPrint jr =
        JasperFillManager.fillReport(JasperReport jasperReport,Map parameters);
JasperPrint jr =
        JasperFillManager.fillReport(String jasperFilePath,Map parameters);
JasperPrint jr =
        JasperFillManager.fillReport(InputStream inputStream,Map parameters,Connecetion conn);
JasperPrint jr =
        JasperFillManager.fillReport(JasperReport jasperReport,Map parameters,Connection conn);
JasperPrint jr =
        JasperFillManager.fillReport(String jasperFilePath,Map parameters,Connection conn);
JasperPrint jr =
        JasperFillManager.fillReport(InputStream inputStream,Map parameters,JRDataSource ds);
JasperPrint jr =
        JasperFillManager.fillReport(JasperReport jasperReport,Map parameters,,JRDataSource ds);
JasperPrint jr =
        JasperFillManager.fillReport(String jasperFilePath,Map parameters,,JRDataSource ds);
```

الطريقة fillReportToFile :-

هذه الطريقة(method) تختص بملء قالب التقرير المترجم بالبيانات من مصدر البيانات و تحوله إلي ملف بإمتداد .jprint

```
JasperFillManager.fillReportToFile(JasperReport jasperReport,
                                   String destFileName,Map  parameters);
 JasperFillManager.fillReportToFile(String sourceFilePath,String  destFilePath,Map  parameters);
 JasperFillManager.fillReportToFile(JasperReport jasperReport,
                                   String destFileName,Map parameters, Connection con);
JasperFillManager.fillReportToFile(String sourceFilePath,
                                   String  destFilePath,Map  parameters, Connection conn);
JasperFillManager.fillReportToFile(JasperReport jasperReport,
                                   String destFileName,Map parameters, JRDataSource ds);
JasperFillManager.fillReportToFile(String sourceFilePath,
                                   String  destFilePath,Map  parameters, JRDataSource ds);
```

مصادر البيانات:-

قبل أن نبدأ شرح كيفية إستعمال الفئة السابقة لنتكلم قليلاً عن مصادر البيانات التي تتعدمها هذه المكتبة. غالبا ما تكون مصادر البيانات عبارة عن كائنات من فئات (Classes)أو واجهات(Interfaces) معظم هذه الفئات او الواجهات تطبق الواجهة **net.sf.jasperreports.engine.JRDataSource**.عند تطبيق هذه الوجهة فإنك تطبق طريقتين هما:

public boolean next() throws JRException;
public Object getFieldValue(JRField jrField) throws JRException;

الطريقة الأولى تعمل نفس عمل الطريقة next في الواجهة ResultSet.
الطريقة الثانية تستخدم لجلب قيمة الحقل المعطى في الصف الحالي لمصدرالبيانات

أنواع مصادر البيانات:-

مصادر البيانات التى تعتمدها المكتبة كالأتي:

1. JDBC
2. POJOs:Plain Old Java Objects
3. Map-Based
4. TableModel
5. XML
6. CSV : comma-separated value
7. مصادر بيانات خالية

مصدر بيانات من نوع JDBC :-

عادة ما يكون هذا النوع عبارة كائن من نوع java.sql.Connection وهو يعبر عن وصلة لقاعدة البيانات. ويستخدم هذا النوع من المصادر في حالة ما إذا كنت تصدر تقريرك مباشرة من قاعدة البيانات عن طريق عبارة إستعلام SQL. هذا النوع هو الوحيد الذي لا يطبق الواجهة **net.sf.jasperreports.engine.JRDataSource**.

مصدر بيانات من نوع POJOs:Plain Old Java Objects:-

وهو يعرف أيضاً بإسم ***JavaBeans Data Source*** و يعمل هذا النوع من المصادر عن طريق فئتين هما:-

net.sf.jasperreports.engine.data.JRBeanArrayDataSource
net.sf.jasperreports.engine.data.JRBeanCollectionDataSource

الفئة(Class) الاولى تعمل عن طريق إحتواء مصفوفة(Array) من الكائنات(JavaBeans objects) التي تعبر عن صفوف التقرير أما الفئة (Class)الثانية تعمل طريق إحتواء مجموعة(Collection) من الكائنات (JavaBeans objects) التى أيضاً تعبر عن صفوف التقرير. كل كائن (object)- والتى هى من فئة واحدة -داخل المصفوفة (Array) أو المجموعة (Collection) سيعتبر صف من صفوف التقرير.

يراعى عند توصيف الفئة التي تعبر عن صف البيانات في التقرير يجب مراعاة ان كل حقل من حقول التقرير يجب أن يقابله كائن بنفس الإسم ونفس النوع ونراعي أيضا وجود المحددات (setters) و النوائل(getters).

مثال:-
تقرير عن بيانات أشخاص (تعريف مصدر البيانات)

```
public class Person {
	private String name;
	private String address;
	private String phone;

	public Person(String name,String address,String phone) {
		this.name = name;
		this.address = address;
		this.phone = phone;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getName() {
		return name;
	}

	public void setAddress(String address) {
		this.address = address;
	}

	public String getAddress() {
		return address;
	}

	public void setPhone(String phone) {
		this.phone = phone;
	}

	public String getPhone() {
		return phone;
	}

}
```

الفئة المعرفة في الشفيرة السابقة هي التي تنشئ منها الكائنات التي تعبر عن الصفوف في التقرير. يجب مراعاة أن خصائص (Properties) الفئة (Person) تطابق الحقول في التقرير .لتهيئة مصدر البيانات :-

1. ننشئ المصفوفة (Array)أو المجموعة(Collection)

```
private Collection<Person> initializeCollection (){
		ArrayList<Person> coll = new ArrayList<Person>();
		coll.add(new Person("Ahmad","Egypt","123456789"));
		coll.add(new Person("Ali","KSA","123456987"));
		coll.add(new Person("mohammed","PALESTINE ","98654721"));
		return coll;
	}
```

في الشفيرة السابقة أعددنا مجموعة (Collection) من الكائنات التى تعبر عن الأشخاص ومعلوماتهم

2. نعرف مصدر البيانات

```
RBeanCollectionDataSource dataSource = new
	JRBeanCollectionDataSource(initializeCollection());
```

مصدربيانات من نوع Map-Based:-

طريقة العمل مع هذا الانوع تشبه النوع السابق وذلك لوجود فئتين من نفس نوع الفئتين في النوع السابق وهما :-

net.sf.jasperreports.engine.data.JRMapArrayDataSource
net.sf.jasperreports.engine.data.JRMapCollectionDataSource

الفئة(Class) الاولى تعمل عن طريق إحتواء مصفوفة(Array) من الكائنات(java.util.Map objects) التي تعبر عن صفوف التقرير أما الفئة (Class)الثانية تعمل طريق إحتواء مجموعة(Collection) من الكائنات (java.util.Map objects) التى أيضاً تعبر عن صفوف التقرير. كل كائن من المصفوفة(Array) أوالمجموعة (Collection) هو كائن من الواجهة java.util.Map.

مثال:-

```
private Collection<Map> intializeMapCollection(){
            ArrayList<Map> coll = new ArrayList<Map>();
            HashMap row1 = new HashMap();
            row1.put("name", "Ahmad");
            row1.put("address", "Egypt");
            row1.put("phone", "123456789");

            HashMap row2 = new HashMap();
            row2.put("name", "Ali");
            row2.put("address", "KSA");
            row2.put("phone", "123456987");

            HashMap row3 = new HashMap();
            row3.put("name", "mohammed");
            row3.put("address", "Egypt");
            row3.put("phone", "98654721");

            coll.add(row1);
            coll.add(row2);
            coll.add(row3);

            return coll;
      }
```

الطريقة (method) السابقة توصف مجموعة كائنات من java.util.Map كل كائن منهم يمثل صف من صفوف التقرير. وفي كل كائن منهم لابد من وضع قيمة كل حقل عن طريق put().

Mapobject.put(Fieldname,Value);

ملحوظة: لا بد أن تستطيع التعامل مع الواجهة java.util.Map والفئات التي تطبقها إذا كنت ستتعامل مع هذا النوع من المصادر

تعريف مصدر البيانات

```
private JRMapCollectionDataSource dataSource = new
     JRMapCollectionDataSource(intializeMapCollection());
```

مصدر البيانات من النوع Table Model:-

في بعض تطبيقات سطح المكتب تكون بيانات التقرير مهيئة في صورة كائن من javax.swing.table.TableModel. الكائنات من هذه الواجهة تستخدم في تهيئة قالب الجدول من أعمدة و صفوف ثم يطبق على الكائن من javax.swing.JTable ليظهر الجدول بصورته النهائية.

مكتبة JasperReport تستطيع إنشاء تقارير إعتمادا على كائن من javax.swing.table.TableModel كمصدر للبيانات وذلك عن طريق الفئة net.sf.jasperreports.engine.data.JRTableModelDataSource.

```
JRTableModelDataSource datasource = new JRTableModelDataSource(TableModelObject);
```

هناك طريقتين لإستخدام هذا النوع من مصادر البيانات. الاولى- وهي عادة ما تستخدم- إنشاء حقل لكل عمود بنفس الإسم . الطريقة الثانية ربط الحقول مع الأعمدة عن طريق الدليل الصفري(Zero Index based). مثلاً إذا كنت ستربط عمود رقم 4 في الجدول (Index = 3) يكون إسم الحقل في التقرير COLUMN_3.

مصادر البيانات الخالية:-

تستخدم هذه المصادر لعرض شكل التقرير من دون بيانات

```
private JREmptyDataSource dataSource = new JREmptyDataSource();
```

إخراج التقرير بالصورة النهائية:-

هناك ثلاثة طرق لإخراج التقرير في صورته النهائية :-

1. عرضه في نافذة مستقله.
2. طباعته مباشر ة.
3. تصديره إلي هيئات أخرى

عرض التقرير:-

لعرض التقرير في نافذة مستقلة نستخدم الفئة **net.sf.jasperreports.view.JasperViewer**. وهذه الفئة عبارة عن تطبيق بسيط ليعرض التقرير في صورته النهائية في نافذة مستقلة.

```
JasperViewer.viewReport(JasperPrint jp);
JasperViewer.viewReport(JasperPrint jp,boolean Exit_on_Close);
JasperViewer.viewReport(String SourceFile,boolean isXml);
JasperViewer.viewReport(String SourceFile,boolean isXml,boolean exit_on_close);
JasperViewer.viewReport(InputStream inputStream,boolean isXml);
JasperViewer.viewReport(InputStream inputStream,boolean isXml,boolean exit_on_close);
```

exit_on_close : يحدد هذا العلم حالة التطبيق بعد غلق نافذة التقرير. عندما يكون true يغلق التطبيق عند غلق نافذة التقرير. False غلق نافذة التقرير فقط

isXML : يحدد هذا العلم ماذا كان الملف المعطى مكتوب بالـ XML

طباعة التقرير:-

لطباعة التقرير مباشرة بدون الحاجه إلى عرضه نستخدم الفئة **net.sf.jasperreports.engine.JasperPrintManager**. وتحتوي هذه الفئة عل الطرق الأتية:-

لطباعة صفحة معينة في التقرير:-

```
JasperPrintManager.printPage(InputStream inputStream,int pageIndex,
                                                    boolean withPrintDialog);
JasperPrintManager.printPage(String sourceFilePath,int pageIndex,
                                                    boolean withPrintDialog);
JasperPrintManager.printPage( JasperPrint jp, pageIndex,boolean withPrintDialog);
```

لتحويل صفحة إلى صورة :-

```
Image image =  JasperPrintManager.printPageToImage(InputStream inputStream,int pageIndex,
                                                    boolean withPrintDialog);
Image image =  JasperPrintManager.printPageToImage(String sourcefilepath,int pageIndex,
                                                    boolean withPrintDialog);
Image image =  JasperPrintManager.printPageToImage(JasperPrint jasperPrint,int pageIndex,
                                                    boolean withPrintDialog);
```

لطبع عدة صفحات متتابعة :-

```
JasperPrintManager.printPages( InputStream inputStream,int firstPageIndex,int lastPageIndex,
                                                    boolean withPrintDialog);
JasperPrintManager.printPages( String sourcefilepath,int firstPageIndex,int lastPageIndex,
                                                    boolean withPrintDialog);
JasperPrintManager.printPages( JasperPrint jasperPrint,int firstPageIndex,int lastPageIndex,
                                                    boolean withPrintDialog);
```

لطباعة التقرير بالكامل:-

```
JasperPrintManager.printReport(inputStream,boolean withPrintDialog);
JasperPrintManager.printReport(String sourcefilepath, boolean withPrintDialog);
JasperPrintManager.printReport(JasperPrint,boolean withPrintDialog);
```

العلم withPrintDialog يحدد ما إذا كنت تريد نافذة حوار الطباعة أن تظهر

التصدير لهيئات أخرى:-

تصدير التقرير لهيئات أخري يعد من مزايا هذه المكتبة وتتم عن طريق الفئة **net.sf.jasperreports.engine.JasperExportManager**. وتحوي على الطرق الأتية:-

التصدير لهيئة PDF :-

```
        String path = JasperExportManager.exportReportToPdfFile(String fileSourcePath);
       JasperExportManager.exportReportToPdfFile(JasperPrint jp,String destfile);
       JasperExportManager.exportReportToPdfFile(String filesourcePath,,String destfilePath);
       JasperExportManager.exportReportToPdfStram(JasperPrint jp,OutStream outStream);
       JasperExportManager.exportReportToPdfStram(InputStream inStram,OutStream
outStream);
```

ملحوظة : هناك مشكلة عند التصدير إلى PDF واللغة العربية وتم حلها في الرابط الأتي
http://www.arabteam2000-forum.com/index.php?showtopic=142388&hl=jasper++%C7%E1%DA%D1%C8%ED%C9

التصدير لهيئة HTML:-

```
String path = JasperExportManager.exportReportToHtmlFile(String fileSourcePath);
        JasperExportManager.exportReportToPdfFile(JasperPrint jp,String destfile);
       JasperExportManager.exportReportToPdfFile(String filesourcePath,,String destfilePath);
```

التصدير لهيئة XML:

```
String path = JasperExportManager.exportReportToXMLFile(String fileSourcePath);
       JasperExportManager.exportReportToXMLFile(JasperPrint jp,String destfile);
       JasperExportManager.exportReportToXMLFile(String filesourcePath,,String destfilePath);
       JasperExportManager.exportReportToXMLStram(JasperPrint jp,OutStream outStream);
      JasperExportManager.exportReportToXMLStram(InputStream inStram,
                                                  OutStreamoutStream);
```