http://www.ponpo.com/jrs/jpn/intoro.html
http://jasperforge.org/jaspersoft/opensource/business_intelligence/ireport/index.php
http://itextpdf.sourceforge.net/

iReport の起動オプション

```
-cp:p D:\data\Development_tools\java\JDBC\oracle\ojdbc6.jar
```

でクラスパスを指定して起動できる。

iReport & JasperReports で日本語を表示する方法

フォントを PDF に埋め込まない方法

■注意点

半角文字列の長さが正しく計算されないことがあるので注意。
個人的にはフォントを PDF に埋め込む方法を推奨する。

■ iTextAsian.jar を取得

http://itextpdf.sourceforge.net/
から iTextAsian.jar をダウンロード
※必要に応じて iTextAsianCmaps.jar もダウンロードする

■ iReport で Pdf Font name を設定する

--	--
HeiseiKakuGo-W5	ゴシック
HeiseiMin-W3	明朝

のいずれか

■ iReport で Pdf Encoding の設定する

--	--
UniJIS-UCS2-H	Adobe-Japan1 の Unicode エンコーディング
UniJIS-UCS2-V	UniJIS-UCS2-H の縦書きエンコーディング
UniJIS-UCS2-HW-H	UniJIS-UCS2-H と同じ、ただし英文字を半角に置き換える
UniJIS-UCS2-HW-V	UniJIS-UCS2-HW-H の縦書きエンコーディング

のいずれか

PDF にフォントを埋め込む方法（Font extention を使う）

■説明

http://d.hatena.ne.jp/dojum/20121108
http://dev.classmethod.jp/server-side/java/jasperreports-tutorial/

PDF にフォントを埋め込む方法で現在推奨されている方法。
PDF font name や Pdf encoding は非推奨になっている。

■ iReport へのフォント追加

・ツール ->　オプションを選択
・iReport -> Fonts　タブを選択

・フォントを選択する

・PDF 用フォントの設定

ここで選んだフォントは iReport のディレクトリ以下の fonts ディレクトリへコピーされる。

■ iReport でフォントを指定する

PDF Font Name や PDF Encoding はいじらなくてよい。

■ jasperreports Font extention の設定

以下の２つのファイルを作成する。
これらのファイルは iReport の fonts ディレクトリに作成されているので、
自分で作成せずにコピーして編集しても良い。

jasperreports_extension.properties を作成して、

内容は、以下のとおり。
フォントを定義している XML のパスを指定する。

・jasperreports_extension.properties

```
net.sf.jasperreports.extension.registry.factory.fonts=net.sf.jasperreports.engine.fonts.SimpleFontExtensionsRegistryFa
```

フォント定義の XML を作成する。
記載内容は以下の通り。

・irfonts.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<fontFamilies>
   <fontFamily name="Consolas_Migu1M">
       <normal><![CDATA[Consolas_Migu1M-Regular.ttf]]></normal>
       <bold><![CDATA[Consolas_Migu1M-Bold.ttf]]></bold>
       <italic><![CDATA[Consolas_Migu1M-Italic.ttf]]></italic>
       <boldItalic><![CDATA[Consolas_Migu1M-BoldItalic.ttf]]></boldItalic>
       <pdfEncoding><![CDATA[Identity-H]]></pdfEncoding>
       <pdfEmbedded><![CDATA[true]]></pdfEmbedded>
   </fontFamily>

   <fontFamily name="IPA P明朝 ">
       <normal><![CDATA[ipamp.ttf]]></normal>
       <pdfEncoding><![CDATA[Identity-H]]></pdfEncoding>
       <pdfEmbedded><![CDATA[true]]></pdfEmbedded>
   </fontFamily>

</fontFamilies>
```

PDF にフォントを埋め込む方法（PDF Font Name を直接指定する方法）

※推奨されていない

■ iReport で Pdf font name を設定する

フォントを埋め込む場合は、 Pdf font name に 埋め込むフォントのファイル名を書く
例

```
ipag.ttf
```

フォントは、クラスパスからクラスローダによって検索されるので
フォントのパスが

```
project/font/ipag.ttf
```

にあって、クラスパスが

```
project
```

に通っている場合、フォント名は

```
font/ipag.ttf
```

と指定する。

■ iReport で Pdf embedded を設定する

Pdf embedded を

```
true
```

に設定する

■ iReport で Pdf encoding を設定する

Pdf encoding を

--	--
Identity-H	横書き
Identity-V	縦書き

に設定する

使い方

サンプル

■新規レポートの設定

Expression の記述言語を指定する。
もし、Expression の記載言語を Java にするには、レポートのプロパティの Language を Java にする。

■基本

1. ソースからもらうパラメータ、フィールドを必要に応じて追加する
2. 追加したパラメータ、フィールドを使って TextField やサブフォーム、グラフを追加

※ソースコンパイル時にクラスが無いって表示されたときは、iReport のディレクトリ以下から jar ファイルを探して、それっぽい jar ファイルをクラスパスに追加。

1. 罫線は、各 TextField を右クリックして、border を設定すると楽
    1.Line を引いたり、rectangle で罫線を付けてもいいけど、border のほうが楽

■ソースからデータを渡す１

２次元配列からデータを出力するデータソースを作る

出力する

■ソースからデータを渡す２

3.7.6 には ListOfArrayDataSource が追加された。

```
String[] header = new String[] {"name","cost"};
Vector vec = new Vector();
vec.add(new Object[]{"test", "100"});
vec.add(new Object[]{"test1","200"});
vec.add(new Object[]{" あいうえお ","200"});

ListOfArrayDataSource ds = new ListOfArrayDataSource(vec,header );
JasperPrint jasperPrint = JasperFillManager.fillReport(jasperReport,
```

```
        parameters, ds);
```

と言う風に使えるはず・・・。

■ソースからデータを渡す３

```
Vector vec = new Vector();
HashMap map = new HashMap();
map.put("BU","001");
map.put("KA","A");
map.put("NAME","hoge");
map.put("COST",new Double(2));
vec.add(map);

map = new HashMap();
map.put("BU","001");
map.put("KA","A");
map.put("NAME","hoge2");
map.put("COST",new Double(2));
vec.add(map);

JRMapCollectionDataSource ds = new JRMapCollectionDataSource(vec);
JasperPrint jasperPrint = JasperFillManager.fillReport(jasperReport,
     parameters, ds);
```

※ $F{BU} や $F{KA} などを使えるように、iReport でフィールドを追加する

■サブレポート用のデータをソースから渡す

パラメータ用の Map に含めて渡してしまえばいい。

```
parameters.put("subdata",new JRMapCollectionDataSource(vec))
```

$P{subdata} を使えるように、iReport でパラメータを追加する

■グラフの使い方

円グラフの場合はこんな感じ。

```
vec = new Vector();
map = new HashMap();
map.put("項目","項目1");
map.put("value",1);
vec.add(map);

map = new HashMap();
map.put("項目","項目2");
map.put("value",2);
vec.add(map);
```

折れ線グラフの場合は

```
vec = new Vector();
map = new HashMap();
map.put("項目","項目1");
map.put("カテゴリ","カテゴリ１");
map.put("value",1);
vec.add(map);

map = new HashMap();
map.put("項目","項目1");
map.put("カテゴリ","カテゴリ２");
map.put("value",2);
vec.add(map);

map = new HashMap();
```

```
map.put("項目","項目2");
map.put("カテゴリ","カテゴリ１");
map.put("value",1);
vec.add(map);

map = new HashMap();
map.put("項目","項目2");
map.put("カテゴリ","カテゴリ２");
map.put("value",4);
vec.add(map);
```

な感じでデータを送る。Series に項目 n を設定すると、項目毎にグループ化を行って出力される。

■プリンタ直印刷

```
        PrintRequestAttributeSet atts = new HashPrintRequestAttributeSet();
        atts.add(new NumberUp(2)); // Ｎアップの指定
        atts.add(new Copies(5)); // 部数の指定
        atts.add(MediaSizeName.ISO_A4); // サイズの指定
        atts.add(OrientationRequested.PORTRAIT); // 縦方向
        atts.add(SheetCollate.COLLATED); // ドキュメントを部単位で印刷コピー機でいうところのソート機能
でしょうか？
        atts.add(Sides.DUPLEX); // 両面印刷

        PrintServiceAttributeSet atts2 = new HashPrintServiceAttributeSet();
        atts2.add(new PrinterName("PDF Printer",Locale.getDefault()));

        JRExporter exporter = new JRPrintServiceExporter();
        exporter.setParameter(JRExporterParameter.JASPER_PRINT, jasperPrint);
        exporter.setParameter(JRPrintServiceExporterParameter.PRINT_REQUEST_ATTRIBUTE_SET, atts);
        exporter.setParameter(JRPrintServiceExporterParameter.DISPLAY_PAGE_DIALOG, Boolean.FALSE);
        exporter.setParameter(JRPrintServiceExporterParameter.DISPLAY_PRINT_DIALOG, Boolean.TRUE);
        exporter.setParameter(JRPrintServiceExporterParameter.PRINT_SERVICE_ATTRIBUTE_SET, atts2);
        exporter.exportReport();
```

■ページ数表示

http://74.125.153.132/search?q=cache:OrH57gRZFk4J:d.hatena.ne.jp/w650/20051101+jasperreports+%E3%83%9A%E3%83%BC%E3%82%B8%E6%95%B0&cd=1&hl=ja&ct=clnk&gl=jp

JasperReport にて各ページに“ Page X of Y ”（ex.1/2）みたいな記述を入れたい場合には、textField 要素を二つ用意してそれぞれに PAGE_NUMBER 変数を設定する。

そして現在ページ数の textField には、evaluationTime="Now" とし、全体ページ数の textField には evaluationTime="Report" を指定する。

■複数のレポートを１つレポートに結合する

こんな感じで、ページを追加をする。
ただし、１ページ目のサイズに固定されるので注意。

■サイズの違うレポートを１つのＰＤＦに結合する

http://gendosu.jp/redmine/projects/25/wiki/JasperReports-%E3%82%B5%E3%82%A4%E3%82%BA%E3%81%AE%E9%81%95%E3%81%86%E7%94%A8%E7%B4%99%E3%82%84%E3%80%81%E7%94%A8%E7%B4%99%E6%96%B9%E5%90%91%E3%81%AE%E9%81%95%E3%81%86%E7%89%A9%E3%82%92%E5%8D%98%E4%B8%80%E3%81%AEPDF%E3%81%A8%E3%81%97%E3%81%A6%E5%87%BA%E5%8A%9B%E3%81%99%E3%82%8B%E3%81%AB%E3%81%AF?version=2

■テキストフィールドを自動的に折り返す

テキストフィールドで WordWrap 機能を有効にすることができる。
http://stackoverflow.com/questions/10631806/jasperreport-wrap-text-to-show-long-text-in-textfield

```
Stretch With Overflow
```

を有効にすれば、自動的に折り返す。

また、枠（rectangle）に対しても stretchType　を RelativeToBandHeight にすることで自動的に高さを変えることができる。

For rectangle :

```
<rectangle>
    <reportElement stretchType="RelativeToBandHeight" ... />
</rectangle>
```

For textField :

```
<textField isStretchWithOverflow="true">
    ...
</textField>
```

■改行位置がずれる場合

http://kenichiro22.hatenablog.com/entry/20101208/1291779030
PDF を出力する際に、文字の改行位置がおかしくなることがある。
フォントやバージョンに左右されるような気がするが、詳しいことは不明。

解決策としては JRPdfExporter のパラメータ "FORCE_LINEBREAK_POLICY" を true する。
（バージョンによってはパラメータの設定方法が変わっているかも）

```
JRExporter exporter = new JRPdfExporter();
exporter.setParameter(JRPdfExporterParameter.FORCE_LINEBREAK_POLICY, Boolean.TRUE);
```

または、jrxml ファイルに property を追加します。

```
<property name="net.sf.jasperreports.export.pdf.force.linebreak.policy" value="true"/>
```