

ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická
Katedra počítačů (K 13136)

Výukový kurz použití JasperReports frameworku

Semestrální práce do předmětu Softwarové technologie (X36SWT)

Tomáš Procházka

Vedoucí práce: Ing. Ladislava Smítková Janků

©Tomáš Procházka, FEL ČVUT Praha, 2006

Abstrakt

Práce se zabývá výukově orientovaným popisem knihovny JasperReports pro tvorbu tiskových sestav, analýz a zpráv.

Klíčová slova

JasperReports, iReports, iText, JFreeChart, Java, tisková sestava, vizualizace dat, grafy, tabulky, PDF

Obsah

1. Úvod	4
1.1. Knihovna JasperReports	4
1.2. Co je potřeba	5
1.3. Seznámení s možnostmi JasperReport knihovny	5
2. Základy tvorby šablon	7
2.1. Úvod	7
2.2. Struktura šablony	7
2.3. Parametry (Parameters)	8
2.4. Zdroje dat (Data Sources)	9
2.5. Datová pole (Fields)	10
2.6. Výrazy (Expressions)	10
2.7. Proměnné (Variables)	11
2.8. Sekce (Report Sections, Bands)	12
2.9. Skupiny (Groups)	13
2.10. Rámce (Frames)	14
2.11. Fonty a Unicode podpora	14
2.12. Styly (Styles)	14
2.13. Scriptlety (Scriptlets)	15
2.14. Subreporty (Subreports)	15
2.15. I18N (vícejazyčné reporty)	16
2.16. Poddotazy (SubDatasets)	17
2.17. Grafy (Charts)	17
2.18. Křížové tabulky (Crosstabs)	18
2.19. Další možnosti a čárové kódy	18
3. iReport (WYSIWYG editor)	19
3.1. Úvod	19
3.2. Popis prostředí	19
3.3. Popis menu	21
3.4. Zdroje dat (Data Source)	22
3.5. Průvodce tvorbou reportu (Report Wizard)	22
3.6. Scriptlety	24
4. API (použití šablon)	26
4.1. API JasperReportu	26
4.2. Kompilace reportu	27
4.3. Vložení dat	27
4.4. Export a zobrazení reportu	28
4.5. Vlastní zdroje dat (User Datasources)	28
4.6. Scriptlety (Scriptlets)	29
5. Dodatky	30
5.1. Opakovací test	30
5.2. Licenční ujednání	30
5.3. Další informace	30
Literatura	31

Kapitola 1

Úvod

1.1. Knihovna JasperReports

JasperReports™ (dále jen JR) je rozsáhlý open source nástroj pro tvorbu reportů, tedy tiskových sestav. Kromě zobrazení a tisku je schopen poskytnout výstup ve formátu PDF, HTML, RTF, XLS, CSV a samozřejmě také XML. JR je kompletně napsaný v Javě ve formě balíčku, který umožňuje libovolné Java aplikaci poměrně snadno generovat dynamické a flexibilní tiskové sestavy a výstupy.

Jako zdroj dat je možné použít například JDBC ovladač nebo nově je k dispozici také rozhraní pro Hibernate a CSV. Dotazovat na data se tedy dá prostřednictvím jazyka SQL, podporován je také HibernateQL nebo XPath pro vstup v podobě XML souboru.

Sestavy (reporty) se pak generují na základě XML souboru s koncovkou `.jrxml`, který představuje šablonu pro sestavu, definuje tedy vzhled a umístění vkládaných dat. Šablona se před použitím kompiluje do binární verze šablony (`.jasper`), což dává možnost zamezit uživateli modifikovat výslednou sestavu a urychluje zpracování. Šablona se naplní daty a vygeneruje se soubor `.jrprint`, což je univerzální tiskový výstup, který je dále možné konvertovat na PDF, HTML, atd. nebo jej lze odeslat do tiskárny. `.jrprint` je také možné zobrazit prohlížečem, který je součástí knihovny.

JR knihovna je šířena pod LGPL¹ licenci.

Aby se usnadnilo použití, existuje hned několik nástrojů, které zjednodušují tvorbu tiskových sestav. Nejznámější a nejpokročilejším nástrojem je iReports™. Jedná se o samostatný program šířený pod GPL² licenci, který umožňuje vizuální editaci XML šablon. Ukázky vzhledu programu lze najít na webu projektu⁴ nebo v kapitole 3 – „iReport (WYSIWYG editor)“.

Ukázky výsledných tiskových sestav:

- Bar3DChartReport.pdf⁵

¹ <http://www.oss.cz/pdf/lgpl-cz.pdf>

³ V případě potřeby je možné koupit komerční licenci, která je svými možnostmi podobná LGPL.

² <http://www.oss.cz/pdf/gpl-cz.pdf>

⁴ <http://jasperforge.org/sf/wiki/do/viewPage/projects.ireport/wiki/Screenshots>

⁵ [preview/Bar3DChartReport.pdf](http://jasperforge.org/sf/wiki/do/preview/Bar3DChartReport.pdf)

- JChartsReport.pdf⁶
- Pie3DChartReport.pdf⁷
- QueryReport.pdf⁸
- SubDatasetChartReport.pdf⁹

1.2. Co je potřeba

Především je potřeba SDK distribuce Javy a to minimálně verze 1.4. Vše bude funkční také pod 1.5 nebo pod připravovanou 1.6. Javu lze získat na adrese <http://java.sun.com/j2se/1.5.0/>.

Pro snadné spouštění ukázek je potřeba mít nainstalovaný ANT¹⁰, který je možné získat na <http://ant.apache.org/>. Musí být navíc nainstalovaný tak, aby jej bylo možné spouštět z příkazové řádky zadáním příkazu **ant**. Toho lze docílit prostým přidáním bin složky, která je uvnitř binární distribuce Antu, do systémové proměnné PATH. Dále musí být správně nastavena systémová proměnná JAVA_HOME.

Samotný JasperReports je možné získat na <http://jasperforge.org/sf/projects/jasperreports/> a iReport na <http://jasperforge.org/sf/projects/ireport/>.

1.3. Seznámení s možnostmi JasperReport knihovny

Plná distribuční verze, tedy archiv s názvem ve tvaru `jasperreports-1.x.x-project.zip`, obsahuje kromě zdrojových kódů a dokumentace velké množství vzorových příkladů na nichž si lze snadno vyzkoušet všechny možnosti JR, aniž by to vyžadovalo jakékoliv znalosti konfiguračního souboru nebo API rozhraní. Lze si tak snadno udělat představu o tom, co JR umožňuje.

Ukázky se nacházejí ve složce `demo/samples/`. Všechny se dají spouštět pomocí Ant skriptu. Některé ukázky vyžadují spuštěnou databázi, které se nachází ve složce `demo/hsqldb/`. Databáze se spustí pouhým spuštěním příkazu **ant** v dané složce.

⁶ [preview/JChartsReport.pdf](#)

⁷ [preview/Pie3DChartReport.pdf](#)

⁸ [preview/QueryReport.pdf](#)

⁹ [preview/SubDatasetChartReport.pdf](#)

¹⁰ Nástroj umožňující automatické dávkové zpracování složitých úloh jako například sestavení aplikace ze zdrojových kódů.

Samotné ukázky se pak spouští stejně, s tím že vyžadují ještě zadání parametru, který zajistí spuštění příslušného ant tasku, ten automaticky provede určitou činnost. Výpis všech parametrů lze získat zadáním spuštěním **ant -p**.

Bohužel ukázky jsou naspané tak, že je nutné jednotlivé úlohy spouštět ručně. Zpravidla se musí nejprve zavolat **ant javac**, což zkompileje zdrojové kódy ukázky, pak se zavolá **ant compile**, čímž se zkompileje samotný report (.jrxml na .jasper), dále **ant fill**, tím se vytvoří univerzální tiskový výstup (.jrprint) a pak je vždy na výběr několik možností exportu nebo zobrazení, například **ant pdf** a **ant view**.

Kapitola 2

Základy tvorby šablon

2.1. Úvod

Šablona je tvořena XML konfiguračním souborem s koncovkou `.jrxml`. Pomocí tohoto souboru se definuje vzhled výsledného dokumentu, uspořádání jednotlivých prvků, ale také způsob, jakým se získají data pro sestavení reportu¹.

Cílem této kapitoly není kompletní popis všech XML elementů a jejich atributů. Pro tvorbu šablon je mnohem lepší použít aplikaci iReport, která umožňuje grafický návrh. Popisem této aplikace se zabývá další kapitola. Pro pochopení principu tvorby šablon je však nutné znát některé základní vlastnosti a principy, na kterých je šablona postavena. Pokud pak víte, co od programu máte očekávat, je s ním mnohem snazší práce.

Popis vychází z verze 1.2.2.

Oficiální dokumentace k JR je bohužel placená², volně je k dispozici jen stručný tutoriál³ a referenční příručka⁴, která je však hodně bohatým zdrojem informací a tento text se na ní velmi často odkazuje.

2.2. Struktura šablony

Základní kostra XML souboru z definice reportu vypadá takto:

```
<?xml version="1.0"?>
<!DOCTYPE jasperReport
  PUBLIC "-//JasperReports//DTD Report Design//EN"
  "http://jasperreports.sourceforge.net/dtds/jasperreport.dtd">

<jasperReport name="name_of_the_report" ... >
...
</jasperReport>
```

¹Report je oficiální název pro tiskovou sestavu, který se užívá v dokumentaci a v API rozhraní Jasper Reports knihovny.

²<http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/MoreDocs>

³<http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/Tutorial>

⁴<http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/QuickReference>

Za hlavním elementem `jasperReport` následuje celá řada atributů, které definují rozměr stránky reportu, orientaci a další obecné vlastnosti. Podrobný popis všech atributů je k dispozici na:

<http://jasperreports.sourceforge.net/reference/index1.html#jasperReport>⁵.

Pak následují elementy definující parametry reportu, importy Java tříd, dále pak způsob získání dat a jednotlivé atributy získané ze zdroje dat. Potom proměnné, které se během generování reportu vypočítají. A pak hlavní a nejsložitější část definující vzhled, rozmístění a uspořádání jednotlivých datových a grafických elementů.

Podrobnějším popisem významných částí se zabývají následující kapitoly.

2.3. Parametry (Parameters)

Parametry jsou reference na objekty, které se šabloně předají z vnějšku, ještě před naplněním daty. Lze pomocí nich parametrizovat chování šablony a zobecnit tak její uplatnění.

Každý parametr musí být v XML souboru deklarován:

```
<parameter name="ReportTitle" class="java.lang.String"/>
<parameter name="MaxOrderID" class="java.lang.Integer"/>
<parameter name="SummaryImage" class="java.awt.Image"/>
```

U jednotlivých parametrů je také možné vyžadovat jejich zadání, určit výchozí hodnotu a doplnit je o komentář.

```
<parameter name="age" isForPrompting="true" class="java.lang.Integer">
  <defaultValueExpression ><![CDATA[new Integer(0)]]></defaultValueExpression>
  <parameterDescription><![CDATA[Věk]]></parameterDescription>
</parameter>
```

Parametry se pak vkládají pomocí zápisu `#{ReportTitle}`.

Například do SQL dotazu:

```
SELECT * FROM Orders WHERE OrderID <= #{MaxOrderID}
```

K dispozici je také několik vestavěných parametrů, které se nastaví automaticky:

- `REPORT_PARAMETERS_MAP` – mapa se všemi vestavěnými a uživatelsky definovanými parametry
- `REPORT_CONNECTION` – `java.sql.Connection` objekt z JDBC zdrojem dat

⁵ <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/J>

- `REPORT_DATA_SOURCE` – instance třídy `JRDataSource`, která představuje jeden z vestavěných nebo uživatelem definovaných zdrojů dat (lze tak jako zdroj dat použít cokoliv)
- `REPORT_MAX_COUNT` – číselná hodnota, která umožňuje omezit počet záznamů získaných ze zdroje dat
- `REPORT_SCRIPTLET` – instance třídy `JRAbstractScriptlet` obsahující uživatelský scriptlet
- `REPORT_LOCALE` – instance třídy `java.util.Locale` s informacemi o národním prostředí
- `REPORT_RESOURCE_BUNDLE` – instance třídy `java.util.ResourceBundle` s lokalizačními texty (pro vícejazyčné reporty)
- `REPORT_VIRTUALIZER` – `net.sf.jasperreports.engine.JRVirtualizer` objekt, který se použije pro vytváření stránek
- `REPORT_CLASS_LOADER` – `java.lang.ClassLoader` objekt, který načítá třídy, které se používají v reportu ve výrazech
- `IS_IGNORE_PAGINATION` – `java.lang.Boolean` hodnota, pomocí které je možné vypnout stránkování reportu (vytvoří se jedna dlouhá stránka)

2.4. Zdroje dat (Data Sources)

Účelem reportu je nějakým způsobem reprezentovat větší množství dat, tyto data je potřeba šabloně poskytnout z vnějšku.

JR podporuje velké množství zdrojů dat pomocí speciálního rozhraní `JRDataSource`. K tomuto rozhraní existuje mnoho základních implementací umožňující jako zdroj dat použít Hibernate⁶, XML soubor, JavaBeans objekt a CSV. K dispozici je pak několik speciálních implementací, které umožňují data předat prostřednictvím polí nebo map. A je také samozřejmě možné napsat si vlastní rozhraní.

Za základní implementaci lze považovat třídu `JRResultSetDataSource`, která obaluje `ResultSet` objekt a umožňuje tak jako zdroj dat použít jakoukoliv relační databázi prostřednictvím JDBC ovladače.

Pro JDBC zdroje dat pak existuje ještě jedna speciální varianta, která umožňuje reportu předat přímo `Connection` objekt a samotný SQL dotaz se pak specifikuje přímo v reportu pomocí xml entity `<queryString>`.

```
<queryString><![CDATA[SELECT id,jmeno FROM portal_users]]></queryString>
```

⁶Framework realizující objektově-relační mapování, více na: <http://www.hibernate.org>

Dotaz je samozřejmě možné parametrizovat pomocí parametrů popsaných v předešlé kapitole. Je tak možné z databáze získat v podstatě cokoliv bez nutnosti programovat něco navíc. Entita `<queryString>` má také volitelný parametr `language`, kterým lze specifikovat jazyk dotazu. Výchozí je SQL. Podrobnější popis na: <http://jasperreports.sourceforge.net/reference/index2.html#queryString>⁷.

Podrobněji se zdrojům dat věnuje kapitola 4 – „API (použití šablon)“.

2.5. Datová pole (Fields)

Datová pole úzce souvisí se zdrojem dat. Jde v podstatě jen o definici jednotlivých polí, které zdroj dat poskytuje a o určení jejich typu. Pro SQL dotaz uvedený v předchozí sekci by to mohlo vypadat například takto:

```
<field name="id" class="java.lang.Long"/>
<field name="jmeno" class="java.lang.String"/>
```

2.6. Výrazy (Expressions)

Ne vždy je vhodné přímo zobrazovat data poskytnutá ze zdroje dat. Za tímto účelem umožňuje JR vyhodnocovat složitější výrazy. Platí pro ně stejná pravidla jako pro psaní výrazů v Javě, je možné používat i Java třídy (například pro matematické výpočty).

Ve výrazech je možné používat již dříve zmíněné parametry, pomocí `#{Nazev}` nebo datová pole `#{Nazev}` a stejně tak již dříve vyhodnocené proměnné `#{Nazev}`.

V konfiguračním XML souboru existuje několik entit, které mohou obsahovat výrazy: `<variableExpression>`, `<initialValueExpression>`, `<groupExpression>`, `<printWhenExpression>`, `<imageExpression>` and `<textFieldExpression>`. První z uvedených entit jsou podrobněji popsány v části 2.7 – „Proměnné (Variables)“.

Ukázka složitějšího výrazu pro zobrazení jako řetězec v textovém poli:

```
<textFieldExpression>
  #{FirstName} + " " + #{LastName} + " byl naposledy přihlášen " +
  (new SimpleDateFormat("dd.MM.yyyy")).format(#{lastLoginDate}) + "."
</textFieldExpression>
```

Aby předchozí ukázka fungovala, je nutné, aby v XML konfiguračním souboru reportu byly naincludované třídy, například takto: `<import value="java.util.*" />`

Kromě Javy jde výrazy psát pomocí jazyka Groovy⁸.

⁷ <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/Q>

⁸ <http://jasperforge.org/sf/wiki/do/viewPage/projects.ireport/wiki/IReportAndGroovy>

2.7. Proměnné (Variables)

Proměnné jsou v reportech chápány jako objekty, které vzniknou jako výsledek vyhodnocení výrazu. V těchto výrazech je možné používat již dříve deklarované proměnné. Je tedy důležité pořadí jejich deklarace v XML.

Pro usnadnění práce obsahuje JR několik vestavěných výpočtových mechanismů, které řeší často prováděné výpočty s datovými poli, známé z relačních databází. Například minimální a maximální hodnotu, průměr, počet prvků, součet, atd.

Jednoduchá ukázka, která definuje proměnnou obsahující součet všech záznamů ve sloupci Quantity :

```
<variable name="QuantitySum" class="java.lang.Double" calculation="Sum">
  <variableExpression>${F{Quantity}}</variableExpression>
</variable>
```

U proměnných lze dále určit spoustu dalších parametrů, jako například automatická inkrementace a automatické nulování. Lze tedy například zajistit počítadlo stránek nebo na každé straně vytvářet mezisoučty, jak ukazuje následující příklad:

```
<variable name="QuantitySum" class="java.lang.Double"
  resetType="Page" calculation="Sum">
  <variableExpression>${F{Quantity}}</variableExpression>
  <initialValueExpression>new Double(0)</initialValueExpression>
</variable>
```

Parametr **resetType="Page"** zajistí, že se proměnná na každé nové stránce vynuluje. V ukázce je také určena výchozí hodnota, na kterou se proměnná nastaví ještě před provedením prvního výpočtu.

Podrobný popis všech atributů a parametrů lze najít na <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/V#variable>

K dispozici je také několik vestavěných typů proměnných, které se nastavují automaticky:

- PAGE_NUMBER – číslo aktuální stránky
- PAGE_COUNT – celkový počet stran
- COLUMN_NUMBER – číslo sloupce (v datovém poli)
- COLUMN_COUNT – počet datových sloupců
- REPORT_COUNT – TODO
- GroupName_COUNT – počet pojmenovaných skupin v reportu (TODO: ověřit)

Je třeba dávat pozor na to, že proměnná `#{PAGE_COUNT}` je k dispozici až po vytvoření celého reportu. Předem se neví, kolik bude mít report stran. Aby však bylo možné na každou stránku vložit textovou informaci o počtu stran, je možné u elementu `<textFiled>` zadat atribut `evaluationTime` s hodnotou "Report", čímž se zajistí, že hodnota pole se vypočte až po vytvoření celého reportu, kdy již bude znám počet stran.

2.8. Sekce (Report Sections, Bands)

Po obecném seznámení se dostáváme k samotné grafické stránce šablony.

Každý report je rozdělen do několika sekcí, jedná se o horizontální bloky definované těmito XML entitami: `<background>`, `<title>`, `<pageHeader>`, `<columnHeader>`, `<groupHeader>`, `<detail>`, `<groupFooter>`, `<columnFoter>`, `<pageFooter>`, `<lastPageFooter>`, `<summary>`.

Význam jednotlivých entit je následující:

- `<background>` – definuje společné pozadí pro každou stránku
- `<title>` – definuje nadpis reportu (uvedený jen na první straně)
- `<pageHeader>` a `<pageFooter>` – definuje záhlaví a zápatí, které se opakuje na každé stránce
- `<columnHeader>` a `<columnFoter>` – definuje záhlaví a zápatí tabulky (pokud má report formu tabulky)
- `<groupHeader>` a `<groupFooter>` – definuje hlavičku a patičku GROUP BY skupiny (více v 2.9 – „Skupiny (Groups)“)
- `<detail>` – definuje jeden datový řádek (opakuje se dle počtu záznamu získaných ze zdroje dat)
- `<summary>` – definuje závěrečné shrnutí celého reportu

Report nemusí obsahovat vždy všechny uvedené sekce.

Definice jedné sekce může vypadat takto:

```
<pageHeader>
  <band height="30">
    <rectangle>
      <reportElement x="0" y="0" width="555" height="25"/>
      <graphicElement/>
    </rectangle>
    <staticText>
      <reportElement x="0" y="0" width="555" height="25"/>
      <textElement textAlignment="Center">
```

```

        <font fontName="Helvetica" size="18"/>
    </textElement>
    <text>Nadpis stránky</text>
</staticText>
</band>
</pageHeader>

```

V í c e
na <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/B#band>.

2.9. Skupiny (Groups)

Pomocí skupin lze seskupovat získaná data na základě změny nějakého sloupce nebo více sloupců, podobně jako GROUP BY v relačních databázích. Zjednodušeně to vypadá takto:

```

<group name="state">
  <groupExpression><![CDATA[{$F{state}}]></groupExpression>
  <groupHeader>
  <band height="50">
    ...
  </band>
  </groupHeader>
  <groupFooter>
  <band height="50">
    ...
  </band>
  </groupFooter>
</group>
<group name="city">
  <groupExpression><![CDATA[{$F{city}}]></groupExpression>
  <groupHeader>
  <band height="50" isSplitAllowed="true" >
    ...
  </band>
  </groupHeader>
  <groupFooter>
  <band height="50" isSplitAllowed="true" >
    ...
  </band>
  </groupFooter>
</group>

```

V ukázce se seskupují položky podle státu a pak podle města. Každé seskupení má svou vlastní hlavičku a patičku. Hlavička a patička se zobrazí vždy, když dochází ke změně hodnoty výrazu zadaného v <groupExpression>. Je proto důležité pamatovat na to, že aby seskupení správně fungovalo, musí být data správně seříděna, tak aby stejné položky byly u sebe. Pro výše uvedenou ukázkou by tedy SQL dotaz musel vypadat takto: SELECT * FROM users ORDER BY state,city.

2.10. Rámce (Frames)

Rámce umožňují vytvářet seskupení více objektů, jejichž poloha a atributy `positionType` a `stretchType` se vztahují k rámcům jejichž jsou součástí. Rámec může mít nastavenou také barvu pozadí a okraj.

Více informací na <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/F#frame>.

2.11. Fonty a Unicode podpora

Pro fonty jsou k dispozici dva xml elementy a to ``⁹ a `<reportFont>`¹⁰. První z nich určuje písmo, které se použije pro konkrétní textový element a druhý definuje výchozí písmo pro celý report. Oba elementy mohou kromě názvu písma definovat i typ písma pro PDF (pomocí atributu `pdfFontName`) a také typ kódování v PDF (pomocí atributu `pdfEncoding`). Umožňuje také určit, zda se písmo bude vkládat do PDF (pomocí atributu `isPdfEmbedded`).

Pro export do PDF využívá JR knihovnu `iText`¹¹.

2.12. Styly (Styles)

Pomocí stylů se dají definovat společné vzhledové parametry více elementu.

Ukázka:

```
<style
  name="myStyle"
  isDefault="false"
  pen="2Point"
  fill="Solid"
  isBold="true"
  pdfFontName="Helvetica"
  isPdfEmbedded="false"
/>
```

Prostřednictvím názvu definovaného atributem `name`, se pak na styl odkazuje. Například:

```
<staticText>
  <reportElement
    style="myStyle"
    x="376" y="12" width="82" height="13"
    key="staticText-1"/>
```

⁹ <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/F#font>

¹⁰ <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/R#reportFont>

¹¹ <http://www.lowagie.com/iText/>

```
<text><![CDATA[Static text]]></text>
</staticText>
```

V í c e
na <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/S#style>.

Poměrně novou vlastností JR je možnost definovat podmíněné styly. Realizují se prostřednictvím elementu `<conditionalStyle>`, který dále obsahuje dva podelementy: `<conditionExpression>` a `<style>`. Pokud je kladně vyhodnocen zadaný výraz, aplikují se definice stylů uvedené v podmíněném stylu. Podmíněných stylů může být v jedné definici stylu více, ale nelze je vnořovat do sebe.

V í c e
na <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/C#conditionalStyle>.

2.13. Scriptlety (Scriptlets)

Scriptlet je Java třída, která dovoluje realizovat složitější operace s daty, které by se pomocí standardních nástrojů JR, jako jsou proměnné a výrazy, realizovaly velmi špatně.

Reportu lze přiřadit jen jeden scriptlet. Reference na příslušnou třídu se vkládá přímo do hlavní entity `<jasperReport>`, jako atribut `scriptletClass` jehož hodnota odpovídá názvu třídy. Samotná třída pak může být uložena v samostatném souboru ve stejné složce jako report. Pak se kompiluje společně s reportem. V takovém případě musí být v XML souboru reportu nastaveno `<property name="ireport.scriptlethandling" value="1" />`. Pokud se místo hodnoty 1 použije 2, nebude se třída scriptletu hledat ve stejné složce jako scriptlet, ale bude se hledat standardně v classpath. Může tedy být součástí aplikace, z které se report generuje.

Scriptlet může obsahovat řadu metod, které se volají během tvorby jednotlivých částí reportu, jako je začátek stránky, skupiny nebo detailu. Navíc má každá metoda dvě varianty, jedna se volá před a druhá po vybrané události.

Podrobnější popis tvorby scriptletů v části 4.6 – „Scriptlety (Scriptlets)“.

2.14. Subreporty (Subreports)

Subreporty umožňují rozdělit složitější report na několik částí. Nejde v podstatě o nic jiného, než že se do jednoho reportu vloží jiný na určené místo. Subreport přitom může převzít databázové spojení nebo i parametry s mateřského reportu.

Vložení subreportu se provádí pomocí xml entity `<subreport>` a může vypadat například takto:

```

<subreport>
  <reportElement x="0" y="0" width="530" height="170" key="sreport-1"/>
  <connectionExpression><![CDATA[{$P{REPORT_CONNECTION}}]></connectionExpression>
  <parametersMapExpression><![CDATA[{$P{REPORT_PARAMETERS_MAP}}]></parametersMapExpression>
  <subreportExpression class="java.lang.String">
    <![CDATA["Example_subreport0.jasper"]]>
  </subreportExpression>
</subreport>

```

Element `<connectionExpression>` zajišťuje předání spojení na zdroj dat a využívá k tomu parametr popsany v 2.3 – „Parametry (Parameters)“, který obsahuje aktuální spojení reportu. Stejným způsobem se předají do subreportu i všechny parametry pomocí `<parametersMapExpression>`. Element `<subreportExpression>` pak určuje, odkud se načte samotný subreport. V tomto případě se načte ze souboru zadaného řetězcem, tedy z již zkompilevané podoby reportu.

Více na `#subreport`¹².

2.15. I18N (vícejazyčné reporty)

JR umožňuje přiřadit k reportu `java.util.ResourceBundle` objekt, který pak poskytuje jazykově závislé řetězce.

Lokalizační objekt je buďto možné předat pomocí atributu `resourceBundle` v elementu `<jasperReport>` nebo pomocí parametru `REPORT_RESOURCE_BUNDLE` z prostředí Java aplikace. V prvním případě se JR pokusí načíst standardní `.properties` souboru ze stejné složky jako report. Nejprve se pokusí načíst soubor, jehož název odpovídá zadanému názvu, pak k názvu přidá koncovku dle výchozí lokalizace vašeho OS. Pokud nastavíte atribut `resourceBundle` na "MyReport" a máte systém nastavený na český jazyk, pokusí se JR načíst nejdrive soubor `MyReport.properties` a pak `MyReport_cs.properties` nebo `MyReport_cs_CZ.properties`. Pokud tedy nenajde jazykový soubor pro systémem používaný jazyk, použijí se výchozí hodnoty definované v základním `properties` souboru, stejně tak pokud jazykový soubor existuje, ale něco v něm chybí.

Properties soubor má standardní tvar, tedy na každém řádku je identifikátor = hodnota.

Ve výrazech pak lze používat `${identifikator}` nebo ekvivalent `str("identifikator")`. Často je potřeba do řetězců vkládat například čísla. Za tímto účelem existuje vestavěná funkce `msg()`, která se předá jako první argument maska a jako další tři volitelné argumenty proměnné, které se budou vkládat do masky. Formát masky je shodný se standardní třídou `java.text.MessageFormat`.

¹² <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/S#subreport>

Třída `java.text.MessageFormat` od verze Javy 1.5 umožňuje sice zadat neomezený počet argumentů, ale knihovna JR je stále kompatibilní s Javou 1.4

JR knihovna také podporuje tisk zprava doleva pro jazyky, které to vyžadují.

Ukázku použití vícejazyčných reportů můžete najít v oficiální distribuci JR ve složce `demo/samples/i18n`.

2.16. Poddotazy (SubDatasets)

Poddotazy vznikly především kvůli grafům a tabulkám, které někdy potřebují speciální data, získaná na základě poddotazu. Jde tedy v podstatě o to, že na úrovni generování sekce "detail" (viz. 2.8 – „Sekce (Report Sections, Bands)“) se provede další dotaz, přičemž položky z hlavního dotazu lze použít pro parametrizaci dotazu. Díky poddotazům tak lze řešit situace, které by jinak vyžadovali použití subreport, a přitom se jedná o jednoduché věci.

Krásnou ukázkou použití poddotazů, lze najít v oficiálních distribuci JR ve složkách `demos/samples/charts` a `demos/samples/crosstabs`. Jedná se zároveň o ukázky použití grafů a "křížových" tabulek.

Více na <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/S#subDataset>.

2.17. Grafy (Charts)

JR obsahuje vestavěnou podporu grafů.

Popis všech možností a typů grafů, které JR podporuje je na <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/C#charts>.

TODO: Bude doplněno.

JR využívá pro tvorbu grafů LGPL knihovnu JFreeChart¹³.

Ukázku použití poddotazů lze najít v oficiálních distribuci JR ve složce `demos/samples/charts`.

¹³ <http://www.jfree.org/jfreechart/>

2.18. Křížové tabulky (Crosstabs)

Jedná se o speciální typ tabulek, které umožňují dynamicky seskupovat data jak ve sloupcích, tak v řádcích. Na seskupená data pak lze aplikovat libovolnou agregační funkci.

Shipments to Argentina

Shipments total freight		1997					1998					Total	
		I	II	V	X	XII	1997 Total	I	II	III	IV		1998 Total
No region	Buenos Aires	29,8	38,8	25,3	22,6	1,1	117,7	19,8	93,7	146,1	217,9	477,4	595,1
	Total	29,8	38,8	25,3	22,6	1,1	117,7	19,8	93,7	146,1	217,9	477,4	595,1
Argentina Total		29,8 (5%)	38,8 (7%)	25,3 (4%)	22,6 (4%)	1,1 (0%)	117,7 (20%)	19,8 (3%)	93,7 (16%)	146,1 (25%)	217,9 (37%)	477,4 (80%)	595,1 (100%)

Obrázek 2.1. Ukázka Crosstabs

Ukázku použití lze najít v oficiálních distribuci JR ve složce demos/samples/crosstabs.

2.19. Další možnosti a čárové kódy

JR dále umožňuje do tiskových sestav vkládat nejrůznější grafické prvky, jako obdélník, čáru, kružnici, elipsu, statický text, obrázek. Obrázky mohou být i dynamicky generované.

Dynamicky generovaný obrázek lze realizovat například takto:

```
<image ... >
  <reportElement .../>
  <graphicElement stretchType="NoStretch" pen="None" fill="Solid" />
  <imageExpression class="java.awt.Image">
    <![CDATA[it.businesslogic.ireport.barcode.BcImage.getBarcodeImage(8, "0815", true, ▶
false, "bar", 0, 0)]]>
  </imageExpression>
</image>
```

Třída použita v ukázce generuje čárový kód a je součástí programu iReport, který také přímo nabízí podporu pro nastavení typu a hodnoty čárového kódu.

Kapitola 3

iReport (WYSIWYG editor)

3.1. Úvod

Program iReport lze získat na <http://jasperforge.org/sf/projects/ireport>. Pro Windows je k dispozici instalátor nebo je možné si stáhnout zip archiv, který stačí rozbalit a spustit `iReport.exe`. Pro spuštění je nutné mít nainstalovanou minimálně Javu SDK 1.4 nebo novější.

Je částečně lokalizovaný do češtiny. V současnosti asi z 53%. Čeština je ale bohužel na mnoha místech spíše matoucí a nepřesná, takže se budu v popisu držet angličtiny, která se navíc shoduje i s pojmy, které již znáte z předchozí kapitoly. (Narazil jsem však na to, že program si alespoň v mnou používané verzi nepamatuje nastavení jazyka a pořád se přepíná na češtinu.)

Oficiální dokumentace k iReportu je bohužel placená¹. Na stejné stránce jsou však odkazy na volně dostupný tutoriál, FAQ a několik dalších odkazů na články zabývající se touto tematikou.

Avšak pokud již znáte základy, není vůbec problém vyzkoušet si příklady dodávané v oficiální distribuci JasperReports knihovny ve složce `demo/samples/`. Ty vás seznámí se vším, co JR knihovna dovede.

3.2. Popis prostředí

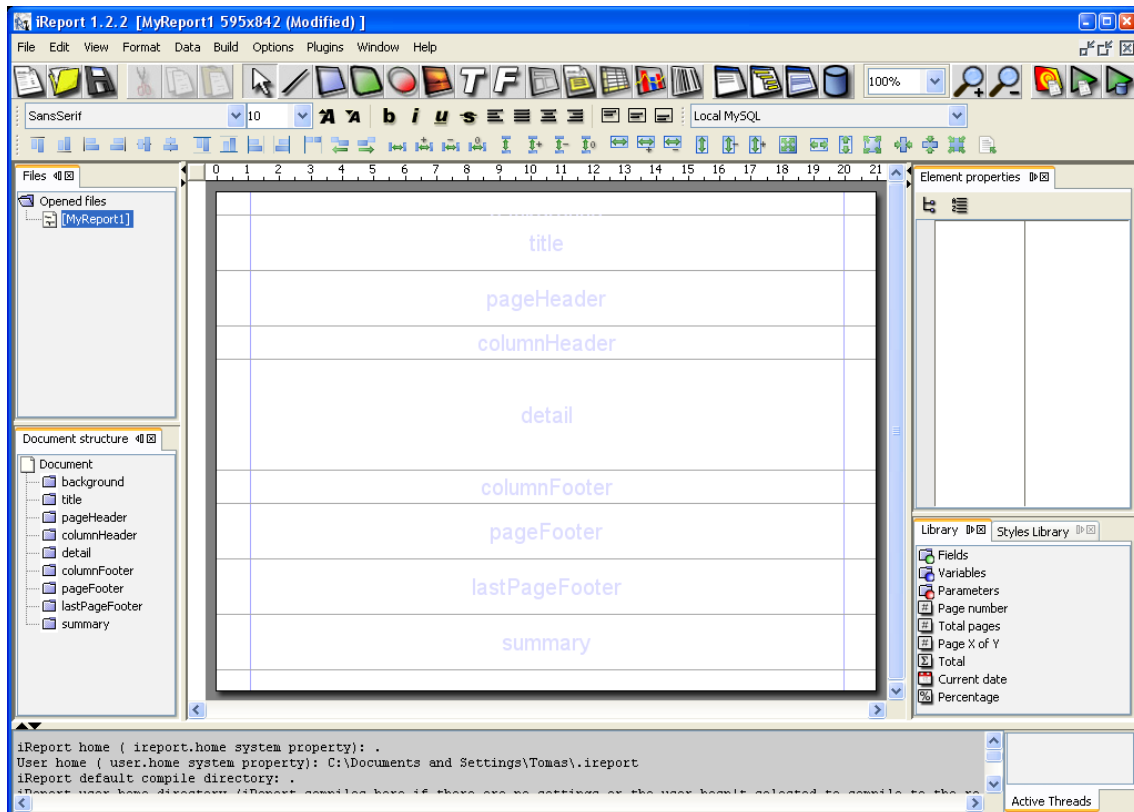
Hlavní obrazovka má standardní vzhled. Nahoře je nástrojové menu, které obsahuje všechny často používané nástroje jako je ukládání, načítání, vkládání různých elementů do návrhu reportu, kompilace a spuštění reportu. Pod hlavní nástrojovou lištou je pak řada menších ikon, které slouží k formátování a zarovnávání elementů.

V levé části se standardně nachází přehled otevřených reportů a pod ním strom aktuálně otevřeného reportu, tedy přehled všech jeho elementů.

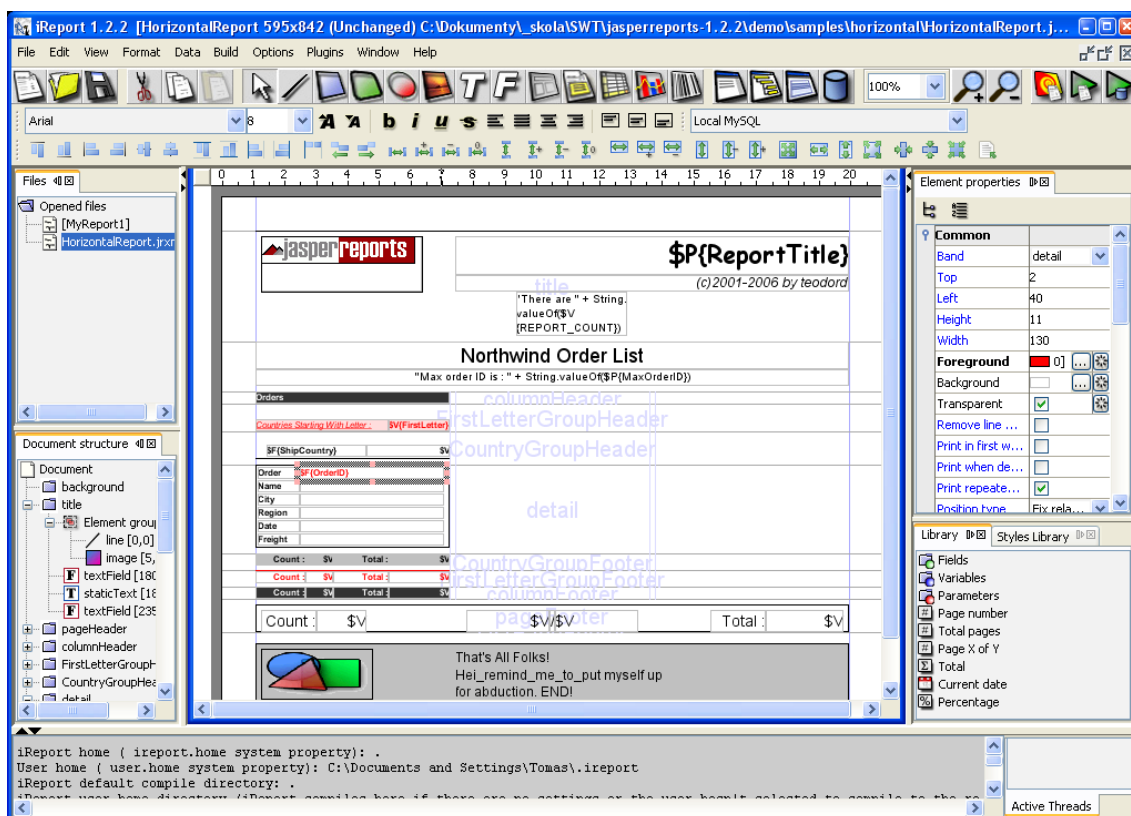
Napravo se pak nachází panel vlastností právě vybraného prvku nebo skupiny prvků a pod ním je k dispozici knihovna stylů, parametrů, proměnných a datových polí, které je možné přímo přetahovat do právě editovaného dokumentu.

¹ <http://jasperforge.org/sf/wiki/do/viewPage/projects.ireport/wiki/HomePage>

Ve spodní části se pak nachází okno compileru, kde se vypisuje průběh kompilace a případné chyby.



Obrázek 3.1. Hlavní obrazovka



Obrázek 3.2. Hlavní obrazovka s načteným reportem

3.3. Popis menu

Význam většiny položek menu je na první pohled zřejmý, proto uvedu jen několik poznámek.

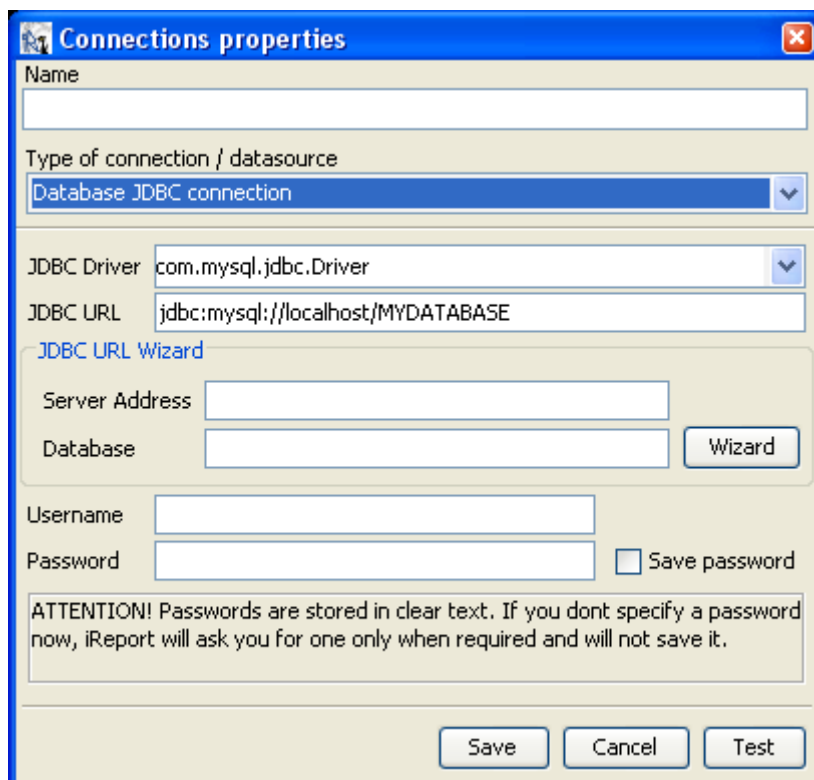
Důležité je menu *Edit*, tam se nachází veškerá nastavení týkající se aktuálně editovaného reportu, jeho chování, vzhledu. Lze zde definovat import tříd, které pak půjde používat ve výrazech atd.

Co se nastavení reportu týče, je důležité ještě menu *View*, kde lze definovat parametry reportu, proměnné, datové pole, seskupení dat a také měnit rozvržení a parametry jednotlivých sekcí reportu. Některé položky jsou k dispozici také jako ikony v tool menu.

Obecná nastavení jsou pak v menu *Options*. Důležitá je především položka *Classpath*, která umožňuje definovat cesty, kde se budou hledat knihovny, které pak můžete používat jako zdroje dat, scriptlety nebo se na ně prostě odkazovat z výrazů. Menu dále obsahuje obecná nastavení prostředí a chování iReportu, konfiguraci kompatibility s JR knihovnou, konfiguraci exportů a pluginů.

3.4. Zdroje dat (Data Source)

iReport umožňuje předdefinovat libovolné množství zdrojů dat, které pak lze použít během návrhu šablony. Seznam datových zdrojů lze zobrazit výběrem *Data* → *Connection / datasources* z hlavního menu.



Obrázek 3.3. Dialog pro nastavení typu a parametrů zdroje dat

K dispozici jsou následující typy spojení: Database JDBC connection, XML file datasource, JavaBeans set datasource, Custom JRDataSource, File CSV datasource, JRDataSourceProvider a Hibernate connection.

V menu *Data* je také možné dále definovat výchozí spojení, které se pak použije při spuštění reportu.

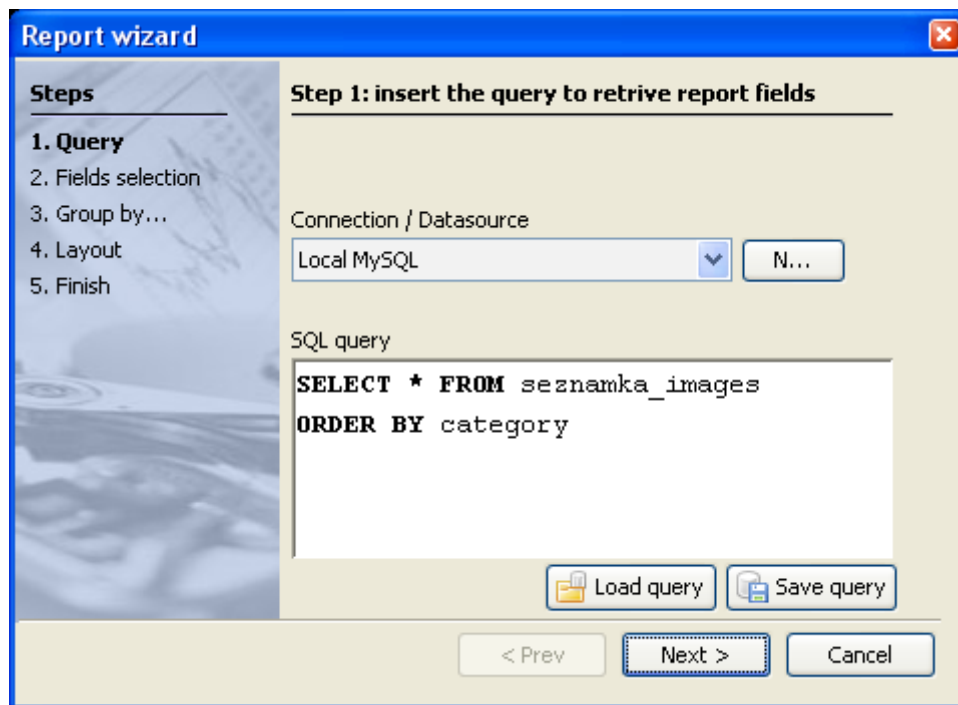
Další dvě položky se vztahují k právě editovanému reportu a umožňují nastavit hlavní SQL dotaz reportu a popřípadě také poddotazy (viz 2.16 – „Poddotazy (SubDatasets)“).

3.5. Průvodce tvorbou reportu (Report Wizard)

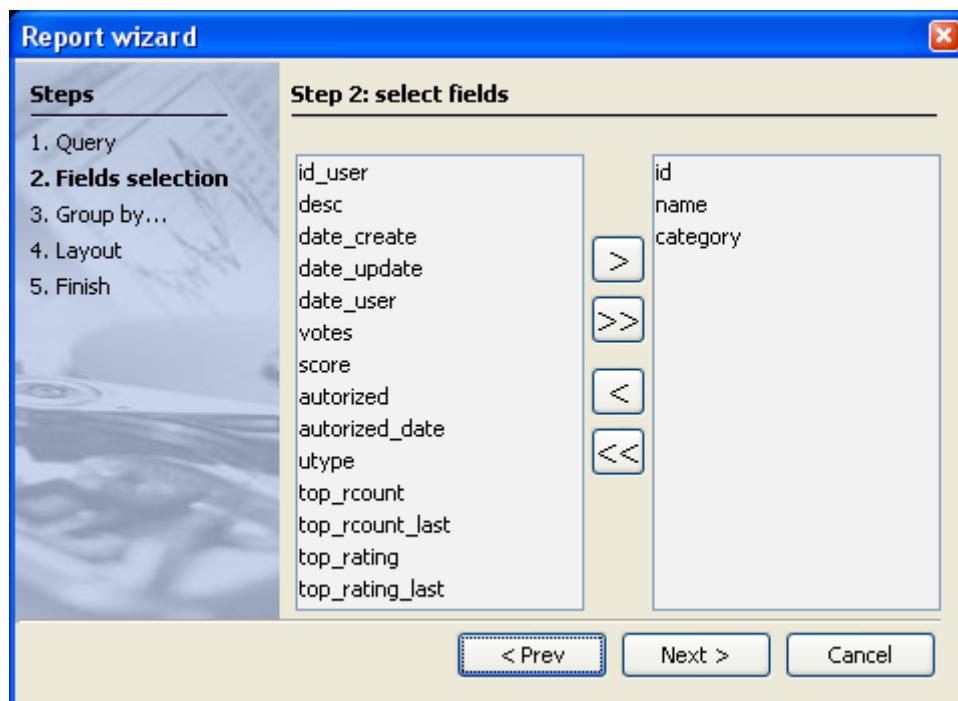
Pro tvorbu většiny reportů se sloupcově nebo tabulkově orientovaným designem poskytuje aplikace iReport velice užitečného průvodce. Ten umožňuje v několika

3.5. Průvodce tvorbou reportu (Report Wizard)

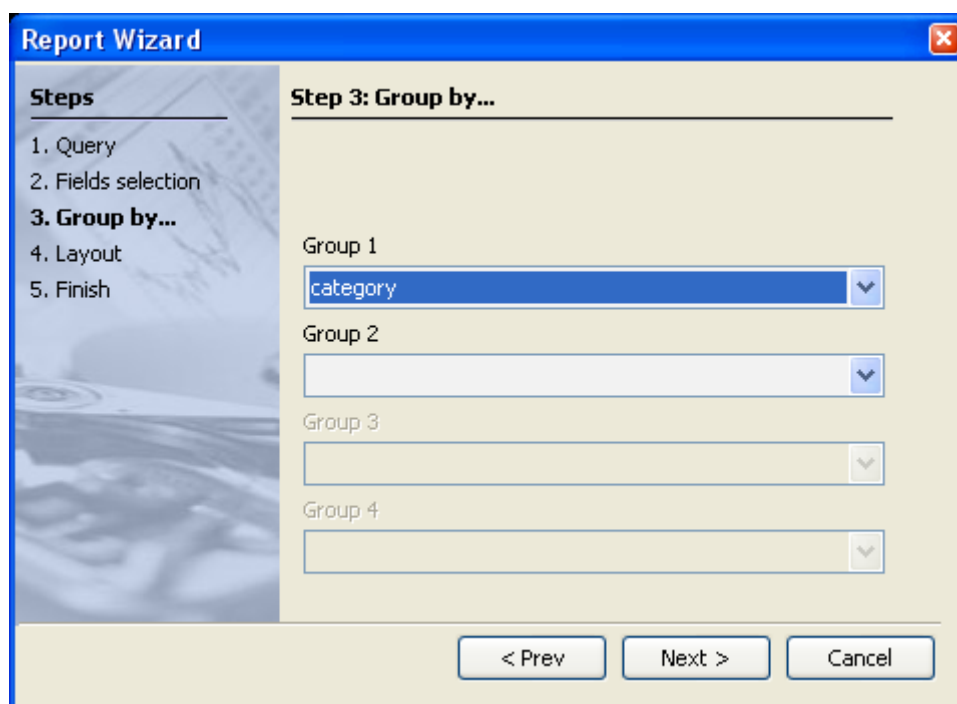
krátkých krocích nastavit vše potřebné a o zbytek se postará sám. Průvodce je k dispozici v hlavním menu v *File* → *Report Wizard*.



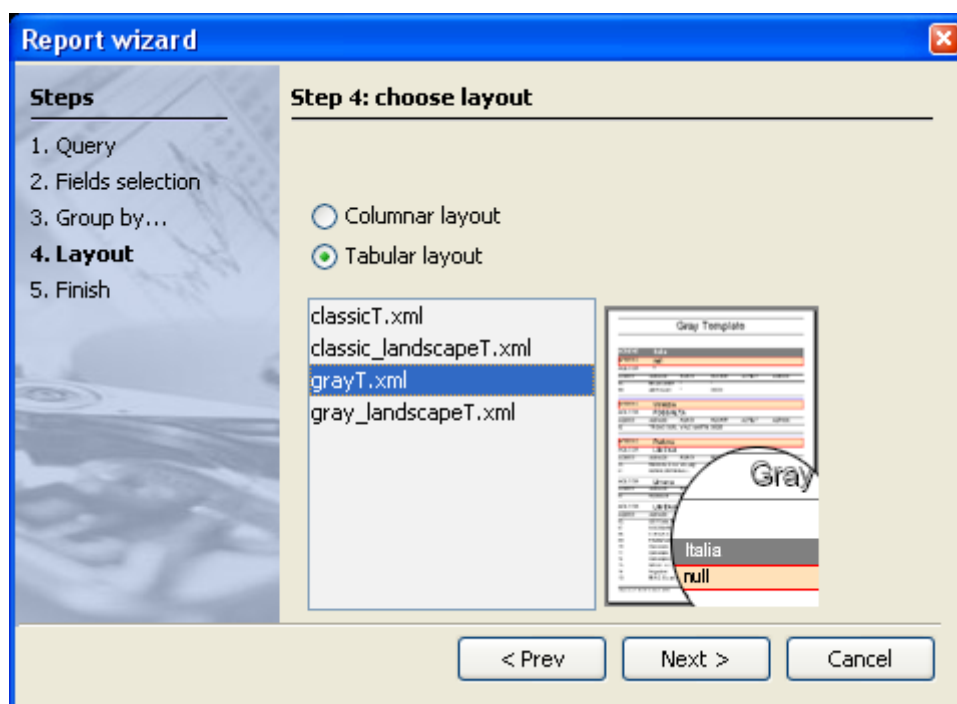
Obrázek 3.4. Krok 1: Výběr databázového spojení a zadání SQL dotazu



Obrázek 3.5. Krok 2: Výběr sloupců, které se v reportu zobrazí



Obrázek 3.6. Krok 3: Výběr sloupce, podle kterého se provede seskupení



Obrázek 3.7. Krok 4: Výběr vzhledu reportu

3.6. Scriptlety

Scriptlety se definují v menu *Edit* → *Report properties* na kartě *Scriptlet*.

iReport obsahuje také integrovaný jednoduchý editor scriptletů v menu *Edit* → *Scriptlet editor*.

Pokud se vám stane, že během kompilace reportu, který obsahuje scriptlet dojde k chybě `java.lang.ClassNotFoundException: com.sun.tools.javac.Main`, musíte přes *Options* → *Classpath* přidat `tools.jar` ze složky `lib` v adresáři, kde máte nainstalované SDK Javy. Jde totiž o to, že iReport byl spuštěn přes JRE verzi Javy, která neobsahuje podporu pro kompilaci Javy

Kapitola 4

API (použití šablon)

4.1. API JasperReportu

Cílem této kapitoly je poskytnout základní znalosti v používání JR knihovny s vlastní Java aplikací, tedy znalosti API.

Pro práci s JR knihovnou je především nutná znalost následujících tříd:

- `net.sf.jasperreports.engine.JasperCompileManager`
- `net.sf.jasperreports.engine.JasperFillManager`
- `net.sf.jasperreports.engine.JasperPrintManager`
- `net.sf.jasperreports.engine.JasperExportManager`

Tyto třídy představují fasádu (návrhový vzor Façade¹) pro jednoduchý přístup ke všem základním operacím s reporty. Statické metody jednotlivých tříd umožňují velice snadno realizovat kompilaci reportu, vyplnění reportu daty, tisk a export do libovolného formátu (PDF, HTML a XML). Pokud někdy narazíte na situaci, kdy vám nabízené metody nebudou stačit, stačí se podívat do jejich implementace, kde zjistíte, co přesně provádí a na základě toho si můžete vytvořit vlastní variantu.

Další důležitou třídou je:

- `net.sf.jasperreports.view.JasperViewer`

Ta slouží pro zobrazení náhledu výsledného reportu v prostředí Java aplikace (Swing). Například za účelem předtiskového náhledu.

A pomocí třídy `net.sf.jasperreports.view.JasperDesignViewer` je možné zobrazit prázdný report jako ukázkou designu.

Důležitým a zároveň vždy aktuálním zdrojem informací je JavaDoc dokumentace: <http://jasperreports.sourceforge.net/api/index.html>.

¹ <http://home.earthlink.net/~huston2/dp/facade.html>

4.2. Kompilace reportu

Kompilace XML souboru reportu, tedy souboru `.jrxml` se provádí pomocí metod `compileReport...()`, které se nachází ve třídě `net.sf.jasperreports.engine.JasperCompileManager`. Ve starších verzích se nacházely v `net.sf.jasperreports.engine.JasperManager`, zde jsou již ale označeny jako *deprecated*.

Během procesu kompilace se z XML souboru vygeneruje objekt (`net.sf.jasperreports.engine.JasperReport`), který je možné následně v serializované podobě uložit na disk jako `.jasper` soubor. `JasperCompileManager` také poskytuje metodu `compileReportToFile()`, která místo vrácení instance reportu provede přímé uložení do souboru.

Metody na kompilaci reportu umožňují jako vstupní parametr použít kromě cesty k souboru a `InputStreamu` také instanci třídy `net.sf.jasperreports.engine.design.JasperDesign`. Tato třída představuje v podstatě objektový model zdrojového XML souboru a umožňuje s ním manipulovat. V případě potřeby je tedy možné načíst pomocí metody `JRXmlLoader.load()` `.jrxml` soubor, provést na něm nějaké úpravy a pak teprve ho nechat zkompilevat.

Během kompilace se také kompilují výrazy a provádí se celková kontrola reportu.

Do verze 1.2.0 nebyly `.jasper` soubory, tedy serializované verze kompilovaného reportu, kompatibilní mezi jednotlivými verzemi. S výměnou knihovny bylo nutné provést znovu kompilaci. S každou verzí se totiž měnil `serialVersionUID` atribut třídy, který se nyní od verze 1.2.0 již nebude měnit.

4.3. Vložení dat

Vložení dat do zkompilevaného reportu se provádí pomocí metod `fillReport...()`, které se nachází ve třídě `net.sf.jasperreports.engine.JasperFillManager`. Ve starších verzích se nacházely v `net.sf.jasperreports.engine.JasperManager`, zde jsou již ale označeny jako *deprecated*.

Metody pro vyplnění reportu mohou pracovat přímo s objektem `JasperReport`, který vznikl po kompilaci nebo se souborem `.jasper` načteným ze souboru nebo `InputStreamu`. Dále se při vyplňování předává mapa parametrů a zdroj dat v podobě instance třídy `java.sql.Connection` nebo třídy implementující rozhraní `net.sf.jasperreports.engine.JRDataSource`.

Výsledkem je instance objektu `net.sf.jasperreports.engine.JasperPrint`, kterou je možné dále použít k tisku nebo k exportu, nebo ji lze v serializovaném tvaru uložit jako `.jrprint` soubor. K oběma variantám jsou k dispozici příslušné metody.

Ukázku práce se speciálními zdroji dat, jako je XML soubor, CVS soubor, JavaBean a další najdete v oficiálních distribuci JR ve složce `demos/samples/datasource`, `demos/samples/csvdatasource` a `demos/samples/xmldatasource`.

4.4. Export a zobrazení reportu

Vyplněný report (soubor `.jrprint` nebo instance třídy `JasperPrint`) je možné přímo snadno zkonvertovat do formátu PDF, HTML nebo XML pomocí statických metod `exportReport...()`, které jsou k dispozici ve třídě `net.sf.jasperreports.engine.JasperExportManager`.

Konverze do formátu XLS nebo RTF vyžaduje nastavení speciálních parametrů a musí se použít speciální třídy určené pro tento účel. Princip je patrný z následujících dvou ukázek, kde se exportuje do RTF a XLS.

```
JRRtfExporter exporter = new JRRtfExporter();
exporter.setParameter(JRExporterParameter.JASPER_PRINT, jasperPrint);
exporter.setParameter(JRExporterParameter.OUTPUT_FILE_NAME, "report.rtf");
exporter.exportReport();
```

```
JRXlsExporter exporter = new JRXlsExporter();
exporter.setParameter(JRExporterParameter.JASPER_PRINT, jasperPrint);
exporter.setParameter(JRExporterParameter.OUTPUT_FILE_NAME, "report.xls");
exporter.setParameter(JRXlsExporterParameter.IS_ONE_PAGE_PER_SHEET, Boolean.TRUE);
exporter.exportReport();
```

V obou ukázkách proměnná `jasperPrint` uchovává referenci na instanci třídy `JasperPrint`, která se získá buďto jako výsledek vyplnění zkompilevaného reportu daty nebo ji lze načíst ze souboru `.jrprint` pomocí:

```
JasperPrint jasperPrint = (JasperPrint)JRLoader.loadObject(new File("report.jrprint"));
```

4.5. Vlastní zdroje dat (User Datasources)

Pokud potřebuje report naplnit daty z nějakého speciálního zdroje, pro který neexistuje výchozí implementace, musíte si napsat vlastní třídu, která musí implementovat rozhraní `net.sf.jasperreports.engine.JRDataSource`. Toto rozhraní vyžaduje pouze dvě metody, a to `getFieldValue(JRField jrField)` a `next()`. Pokud váš zdroj umožňuje přesun zpět na počátek je výhodnější použít rozhraní `net.sf.jasperreports.engine.JRRewindableDataSource`, které rozšiřuje předchozí rozhraní o metodu `moveFirst()`.

Ukázku použití najdete v oficiální distribuci JR ve složce `demos/samples/datasource`.

4.6. Scriptlety (Scriptlets)

Scriptlet je Java třída, která dovoluje během generování reportu realizovat složitější operace s daty, které by se pomocí standardních nástrojů JR, jako jsou proměnné a výrazy, realizovaly velmi špatně.

Programují se jako třídy rozšiřující abstraktní třídu `net.sf.jasperreports.engine.JRAbstractScriptlet` nebo `net.sf.jasperreports.engine.JRDefaultScriptlet`. První třída vyžaduje implementaci všech abstraktních metod, zatímco druhá obsahuje výchozí implementaci všech abstraktních metod. Tyto metody se pak automaticky volají během tvorby reportu a mohou tak reagovat a ovlivnit jeho výsledný tvar. Jedná se o metody `beforeReportInit()`, `afterReportInit()`, `beforePageInit()`, `afterPageInit()`, `beforeGroupInit()` atd.

Scriptlet může obsahovat řadu metod, které se volají během tvorby jednotlivých částí reportu, jako je začátek stránky, skupiny a detailu. Navíc má každá metoda dvě varianty: jedna se volá před a druhá po vybrané události.

Kapitola 5

Dodatky

5.1. Opakovací test

TODO - bude doplněno

5.2. Licenční ujednání

Tento dokument je k dispozici zcela zdarma a smí být v nezměněné podobě (tedy jako původní PDF soubor) šířen libovolným způsobem a prostředky.

Autor nenese žádnou zodpovědnost za správnost a aktuálnost obsahu tohoto dokumentu.

5.3. Další informace

Aktuální verze tohoto dokumentu je k dispozici na:

<http://atom.mamto.cz/projekty/skola/SWT/>

Nalezené chyby nebo návrhy na zlepšení můžete posílat na <tom.p@sf.cz>.

Literatura

- [1] *Jasper Tutorial [online]*. JasperSoft Corporation. 2005 [cit. 2006-07-27].
URL: <http://jasperforge.org/sf/wiki/do/viewPage/projects.jasperreports/wiki/Tutorial>.
- [2] *iReport Documentation [online]*. JasperSoft Corporation. 2005 [cit. 2006-07-27].
URL: <http://jasperforge.org/sf/wiki/do/viewPage/projects.ireport/wiki/IReportManualV1.2.1>.
- [3] WALSH, Norman a MUELLNER, Leonard. *DocBook: The Definitive Guide [online]*. O'Reilly & Associates, Inc.. 2005 [cit. 2006-05-30].
URL: <http://docbook.org/tdg/en/html/docbook.html>. ISBN 156592-580-7.
- [4] KOSEK, Jiří. *DocBook [online]*. 2003 [cit. 2006-05-30].
URL: <http://www.kosek.cz/xml/db/>.
- [5] KOSEK, Jiří. *XSLT v příkladech [online]*. 2004 [cit. 2006-05-30].
URL: <http://www.kosek.cz/xml/xslt/>.