Orientation in Objects

High Performance BIRT Reports

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

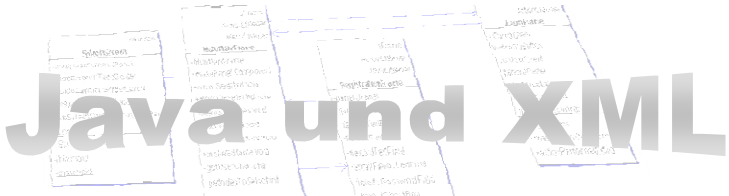
www.oio.de
info@oio.de

Processing, please wait ...

Cancel

Java, XML und Open Source seit 1998

Orientation in Objects



) Software Factory)

- Schlüsselfertige Realisierung von Java Software
- Individualsoftware
- Pilot- und Migrationsprojekte
- Sanierung von Software
- Software Wartung

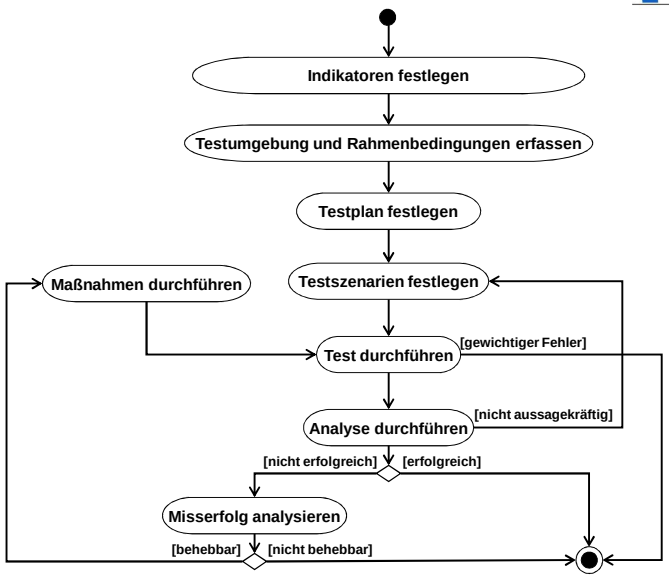
) Object Rangers)

- Unterstützung laufender Java Projekte
- Perfect Match
- Rent-a-team
- Coaching on the project
- Inhouse Outsourcing

) Competence Center)

- Schulungen, Coaching, Weiterbildungsberatung, Train & Solve-Programme
- Methoden, Standards und Tools für die Entwicklung von offenen, unternehmens-weiten Systemen

OIO Performance-Management Prozess



Motivation

- Optimierung der Werkzeuge auf Reportdesign
- Laufzeitaspekte sind oft weniger im Fokus
- Aktuelle Trends:
  - Real Time Business
  - Massenhafter Anfall von Bewegungsdaten
- Lösungsansätze sind Herausforderung für die Performance von Reporting-Tools
  - Onlinebetrieb
  - ESP/CEP
  - NoSQL-Datenbanken
  - ..

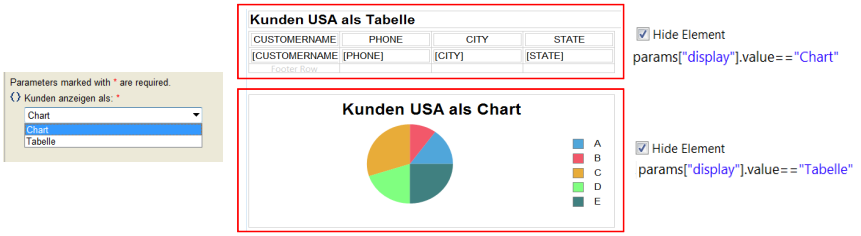
Gliederung

Orientation in Objects

- Berichtsdesign
  - Performance Anti-Patterns
  - Patterns & Tipps
- Runtime
  - Allgemeine Tipps
  - Szenario 1
  - Szenario 2
  - Performance-Management

Anti-Pattern 1: Visibility zur Auswahl zwischen mehreren Alternativen

Orientation in Objects



Empfehlung

Orientation in Objects

- Es werden immer beide Elemente ausgewertet
  - Visibility bestimmt lediglich Präsentation
  - Nur einfache Elemente ausblenden
    - **Andere Elemente zur Laufzeit löschen**

Script: beforeFactory | visibility.rptdesign

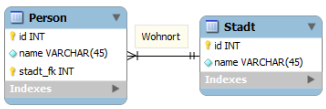
```
var tabelle = reportContext.getDesignHandle().findElement("kundentabelle");
var chart = reportContext.getDesignHandle().findElement("kundenchart");
if(params["display"].value=="Tabelle") {
    chart.drop();
}
else {
    tabelle.drop();
}
```

→ Laufzeit ~halbiert da nur benötigte Elemente generiert

©Thommy Weiss / pixelio.de

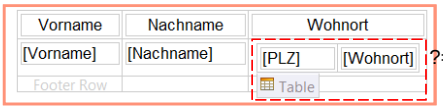
Anti-Pattern 2 – Subreport als Ersatz für Table Join

Orientation in Objects



dsPerson
select from person

dsWohnortForId
select from Stadt where id=?



?=stadt_fk

Subreport als Ersatz für Table Join

Orientation in Objects

- Keine weiteren Optimierungen durch BIRT
  - Wohnort wird für jede Person nachgeladen
    - **Laufzeit essentiell abhängig von Anzahl der Personen**
    - **n+1 selects**
- Zusätzlicher Overhead
  - Verschachtelte Tabellen
  - Viele Datasets in rptdocument statt 1
- Aber: Subreports nicht generell schlecht!
  - Voraggregieren von Daten in DB bei kleinem n

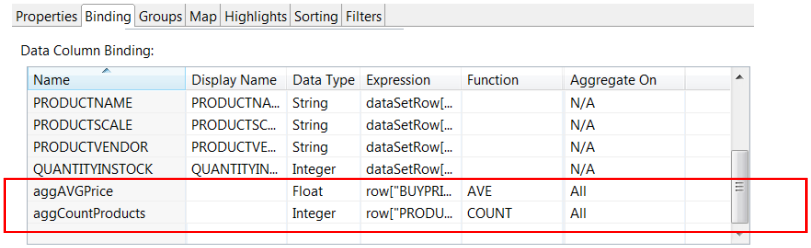
Anti-Pattern 3: Getting caught in a (Data) Bind

Orientation in Objects

- Data Sets in BIRT an Elemente gebunden
  - Daten werden geladen bei Ausgabe eines Elements das an DataSet gebunden
    - **Einsatz eines DataSet Cache (ab BIRT 2.1.3)**
      - Kein signifikanter Nachteil wenn mehr als 1 Element an DataSet gebunden
  - Element verfügt über eigenen Zwischenspeicher
    - **Visualisieren über Property-Tab Data Binding im Designer**
    - **Hinzufügen von Spalten durch Aggregationen und Data Elemente**
    - **Zwischenspeicher bei Run in rptdocument geschrieben**

Data Binding und Aggregationen

Orientation in Objects



- Elemente von Designer zu Zwischenspeicher hinzugefügt wenn benötigt
  - Achtung: Keine automatische Löschfunktion

Laufzeit Aggregationen

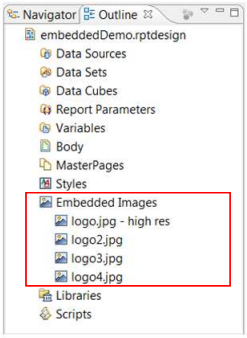
Orientation in Objects

- Unterscheidung in
  - 1 pass aggregates: Aggregation muss ResultSet 1x durchlaufen
    - **AVG, SUM, COUNT,…**
  - 2 pass aggregates: Aggregation muss ResultSet 2x durchlaufen
    - **RANK, IS-TOP-N,…**
- Laufzeit Tabelle mit/ohne Aggregation
  - +2-3% pro one pass
  - +6-8% pro two pass

Anti-Pattern 4: Falsche Verwendung von Embedded Images

Orientation in Objects

- 2 Möglichkeiten um Bilder in BIRT zu hinterlegen
  - Embedded: Bilder werden in rptdesign serialisiert
  - Shared: Bilder werden in externem Ordner hinterlegt



Bilder – Shared oder Embedded?

Orientation in Objects

- Mehrere Tests mit unterschiedlichen Bildgrößen
- Bildgröße 10 MB (1x ausgeben)
  - Shared: 0,05 s / Embedded: 1,24 s
  - Unterschied von Faktor 24!
- Bildgröße 3 KB (1x ausgeben)
  - Kein signifikanter Unterschied
- Bildgröße 10 MB (1.000x)
  - Shared um Faktor 24 schneller
- Bildgröße 3 KB (1.000x)
  - Embedded um Faktor 2,4 schneller

Empfehlung

Orientation in Objects

- Große Bilddateien immer als Shared Resources hinterlegen
- Embedded Images vorziehen wenn
  - Bilddatei sehr klein ist
  - Bilddatei nur für diesen Report gebraucht wird

©Thommy Weiss / pixelio.de

Anti-Pattern 5: Komplexe Programmierlemente in Berichten

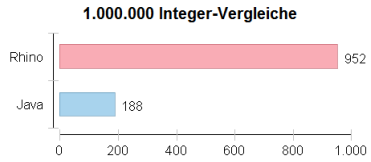
Orientation in Objects

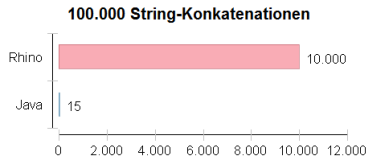
- Mozilla Rhino Script
  - Untypisiert
  - Direkt in Bericht oder externen Dateien hinterlegt
  - Zur Berichtslaufzeit interpretiert
  - Mit Hilfe der JVM ausgeführt
  - Integration von Java Code möglich

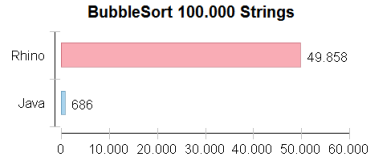
- Java
  - Java Bytecode wird von VM zur Berichtslaufzeit ausgeführt
    - **Hinzufügen der .class Dateien zum Classpath der Anwendung oder über separaten Classloader**

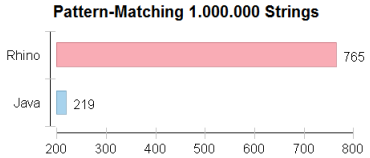
Messergebnisse in ms

Orientation in Objects









Empfehlung

Orientation in Objects

- Komplexe Algorithmen in Java implementieren und über LiveConnect in Rhino verwenden
  - IDE Support
  - Nutzung von externen Libraries möglich
  - → Robuster, komfortabler und schneller
- Hinzufügen durch
  - In Web-Viewer: Hinterlegen im scriptlib Verzeichnis
  - In eigener Applikation: Hinzufügen zum Classpath
- Java Objekte sind Rhino Entsprechungen vorzuziehen
  - z.B. Nutzung von java.lang.StringBuilder statt String.concat
  - Gilt nicht für native Types

©Thommy Weiss / pixelio.de

Patterns & Tipps

Orientation in Objects

- Viele Performance Probleme sind auf DB Ebene lösbar
  - Filtern und Aggregieren in DB
  - Indizes optimal
  - Evtl. auslagern von Logik in Stored Procedure
- Charts
  - Punkte voraggregieren
  - 3D Option nicht verwenden!
- Nur Daten laden die auch benötigt werden
  - Kontrollieren der Data Bindings auf nicht verwendete Daten

  → Spart RAM

  → Reduziert Größe des rptdocuments

©Thommy Weiss / pixelio.de

Patterns & Tipps

Orientation in Objects

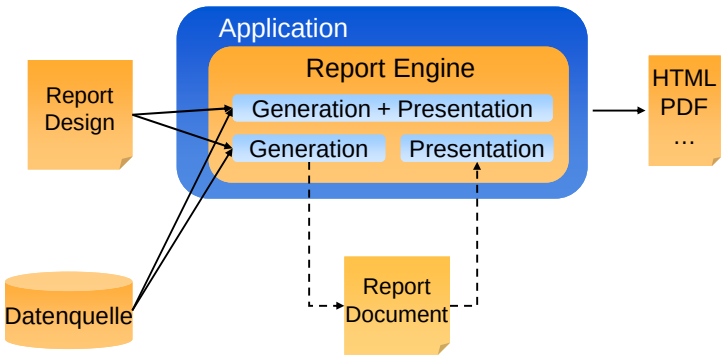
- Häufige Formatierungen über Styles lösen
- Kein Information Overload
  - Stattdessen mit Drill-Through Links arbeiten
- Verschachtelte Gruppen in Tabellen vermeiden
  - Gruppieren in BIRT teuer
  - Laufzeit steigt exponentiell mit Anzahl Verschachtelungen

©Thommy Weiss / pixelio.de

Gliederung

- Berichtsdesign
  - Performance Anti-Patterns
  - Patterns & Tipps
- **Runtime**
  - Allgemeine Tipps
  - Szenario 1
  - Szenario 2
  - Performance-Management

Einführung: BIRT-Workflow



Allgemeine Tipps

Orientation in Objects

- Erzeugung eines Report Engine Objekts teuer
  - Erzeugen und cachen bei Startup der Anwendung
- Öffnen einer Datenbankverbindung teuer
  - Verbindung nicht von BIRT für jeden Reportlauf erzeugen lassen
    - **JDNI**
    - **Datenbankverbindung über Report Engine API injizieren**
- Libraries werden für jeden Report erneut geladen
  - Evtl. eigenen Cache implementieren über Interface IResourceLocator

Einstellungen Report Engine

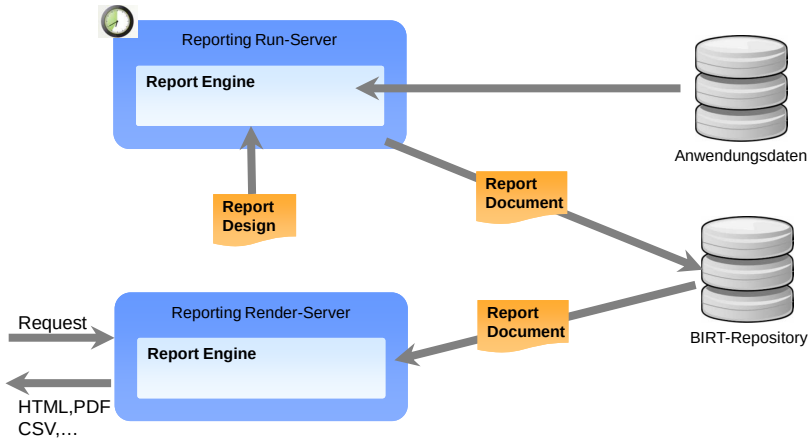
Orientation in Objects

- Entsprechendes Log-Level setzen (default ist INFO)
- Data Engine kann über AppContext pro Task konfiguriert werden
  - task.getAppContext().put(„key“, „value“)

DataEngine.MEMORY_BUFFER_SIZE*	MB-RAM pro DataSet bevor ausgelagert wird
DataEngine.IN_MEMORY_CUBE_SIZE*	MB-RAM pro Cube bevor ausgelagert wird
DE.MEMORY_DATA_SET_CACHE	Anzahl Zeilen die im RAM gecached werden (Achtung: Schränkt ResultSet ein!)
DE.CUBECURSOR_FETCH_LIMIT_ON_COLUMN_EDGE DE.CUBECUSROR_FETCH_LIMIT_ON_ROW_EDGE	Limitiert Zeilen/Spalten eines Cubes auf übergebene Anzahl (default: unbeschränkt)

*Default: 10 MB bis BIRT 2.5.2, unlimitiert ab BIRT 2.6

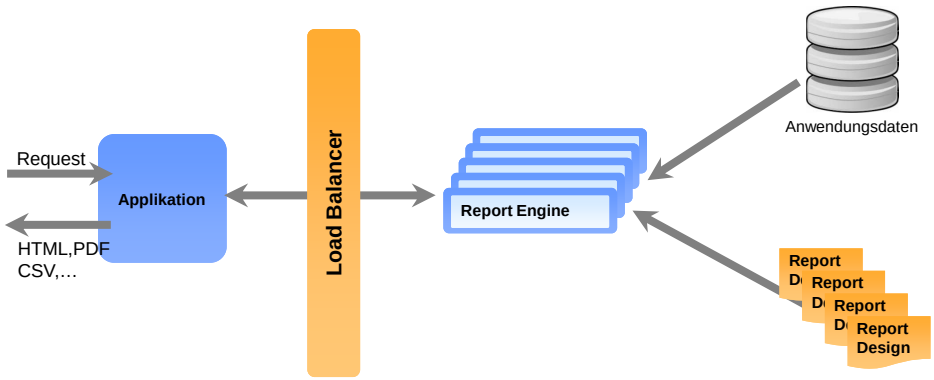
Szenario 1: Datenverarbeitung komplex – Daten in Reports nur tagesaktuell



Szenario 1: Fazit und Konsequenz

- Einfaches Setup
  - Wenig Infrastruktur notwendig
- DB wird geschont
  - Daten in Reports nur so aktuell wie Report Document
- Was passiert wenn Server ausfällt?
  - Failover notwendig?
- Ein Report Document pro Parameterwert
  - Kombinatorisches Problem

Szenario 2: Komplexe Reports – aktuelle Daten



Szenario 2: Fazit und Konsequenz

- Größe des Clusters bestimmt Durchsatz
  - Laufzeit einzelner Reports abhängig von Hardware der Komponenten
- DB wird extrem belastet
  - Kann zu limitierendem Faktor werden
  - Dedizierte Reporting-DB als Alternative
- Reports werden für jede Anfrage generiert
  - Speichern zusätzlicher Dateien entfällt

Performance-Management

Orientation in Objects

- Performanceoptimierung komplex
  - Hängt von vielen Faktoren ab
    - **Hardware, Netzwerk, JVM, …**
- Voraussetzungen
  - Vergleichbare Menge an Testdaten
  - Reproduzierbare Testläufe
    - **Hardware bleibt gleich**
  - Oftmals Einsatz spezialisierter Werkzeuge

→ Kleine Performance Iterationen
  - Einzelne Änderung an Report, danach Test

Orientation in Objects

Fragen ?

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

BIRT User Group Mannheim

- Interessiert an regelmäßigem Austausch zum Thema BIRT?
- 4 Treffen pro Jahr mit je 2 Vorträgen zu Schwerpunktthemen und Diskussionsrunden
- http://www.oio.de/Birt-User-Group/birt-user-group-mannheim/business-intelligence-reporting-tools-vortrag-kostenlos.htm

Mehr von OIO zum Thema

- Schulung Reporting mit Eclipse BIRT
  - http://www.oio.de/seminar/open-source/eclipse-birt-reporting-schulung.htm
- Vortrag Actuate Kundentag: BIRT Best Practices
  - http://www.oio.de/m/konf/actuate-day2010/BIRT-Best-Practices-Actuats-Day.pdf
- Vortrag Jax2009: Eclipse BIRT Day: BIRT within Java Enterprise
  - http://www.oio.de/m/konf/jax2009/BIRT-Enterprise-Architektur.pdf
- Vortrag W-Jax2006: Reporting mit Eclipse BIRT
  - http://www.oio.de/m/konf/wjax2006/ReportingMitEclipseBIRT-final.pdf

Orientation in Objects

Vielen Dank für ihre Aufmerksamkeit !

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de