



Creating logical SAP BusinessObjects Web Intelligence reports the Vulcan way

Alan Mayer
Solid Ground Technologies

In This Session

- Learn how to create reports that are easier to understand and maintain
- Understand the need for more advanced techniques
- Discover how to program using multi-layered logic
- Fine-tune the control of your calculations using contexts
- Interface your reports to other reports and applications
- Standardize the development of future reports

What We'll Cover

- Introduction
- Annotated Report Standards
- Calculation Contexts
- Layered Variables
- Interface Points
- Wrap up

Life as a Vulcan

- The one species where logic means everything
- Emotions are tightly reigned in
- Allows data to drive their decisions rather than belief
- Couldn't choose a better role model as an architect or technician



Life as We Practice It

- Pretending to be logical
- Reliving the same problems by trying the same solution



A Vulcanized Approach to Web Intelligence

- Let's apply the data-driven logical Vulcan viewpoint to our efforts
- We'll visit some of the most common trouble spots
 - ♦ Fully annotating reports
 - ♦ Future-proofing calculations by fully defining their contexts
 - ♦ Creating flawless variables that are maintainable
 - ♦ Launching your reports from within and outside SAP BusinessObjects



What We'll Cover

- Introduction
- **Annotated Report Standards**
- Calculation Contexts
- Layered Variables
- Interface Points
- Wrap up

From the Book of Vulcan

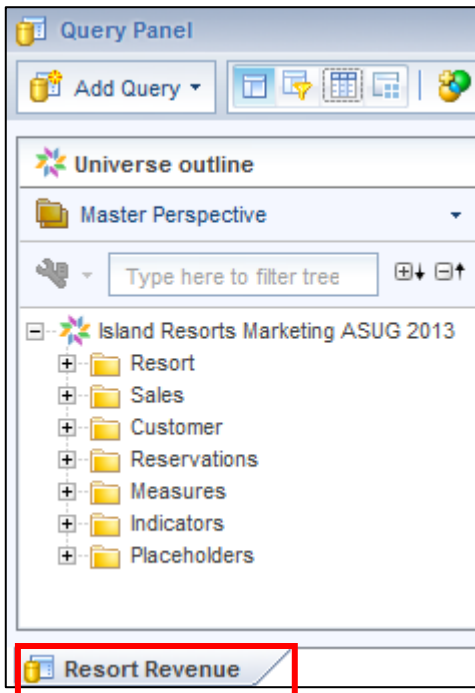
- Discipline goes beyond action.
- Structure your thoughts by giving each a name and purpose for being.



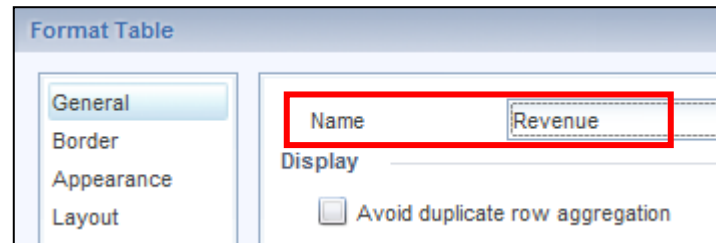
Naming Report Elements

- Everything you create should have a name

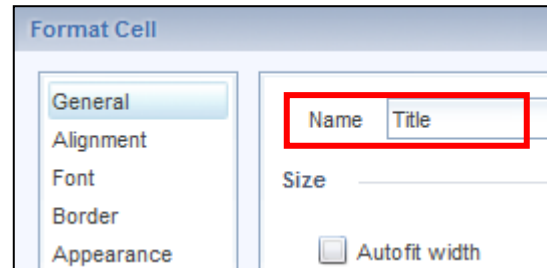
Queries



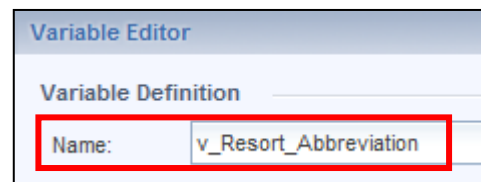
Blocks



Cells



Variables



Best Practice

The Reason for Names

• Placement

- ♦ Helps position blocks and cells relative to one another

• Sorting

- ♦ Allows variables sorted by query rather than alphabetically

Position _____

Within the **Report**



The left of this table is inches to



The top of this table is inches to

☒

Available Objects ▾

Type here to filter tree

- [-] Vulcan 01 - Annotations
 - [-] Invoiced Stays
 - Country
 - Invoice Date
 - Resort
 - Number of guests
 - Revenue
 - [-] Reservations
 - Country
 - Reservation Date
 - Resort
 - Future guests
 - Variables

Naming Variables

- Consider the following rules for variable names
 - ♦ ALWAYS add a prefix to a report variable's name
- Different prefixes give you more control
 - ♦ v_ generic prefix for any variable name
 - ♦ c_ add to constants
 - ♦ p_ add to prompts



Best Practice

Reasons for Variable Names

- Easier maintenance
- Can manipulate the report without refreshing
- Examples:
 - ♦ v_PaymentDue
 - ♦ p_Year
 - ♦ c_BucketSize



Heads-Up

Names are ESSENTIAL
for developing layered
variables (coming up!)

Document Descriptions

- Add descriptions and keywords wherever possible

Document Summary

 **Vulcan 01 - Annotations**

General

Type: Web Intelligence document
Author: Administrator
Creation date: January 18, 2014 10:58:12 AM GMT-08:00

Description

This is a sample document created for the presentation "Web Intelligence reports the Vulcan way". It demonstrates the importance of annotating report elements when building a report. The same concepts can be used to retrofit an existing report.

Keywords

annotation description best practice vulcan|



Good descriptions will allow BI Platform Search to locate this document quicker

Title Pages

- Add a report page as an introduction
 - A cell formatted to wrap text contains your document description
 - An empty two-column table contains your prompts
 - Position the prompt table relative to the description cell

This is a sample document created for the presentation "Web Intelligence reports the Vulcan way". It demonstrates the importance of annotating report elements when building a report. The same concepts can be used to retrofit an existing report.

Prompt	Value
Year	FY2006
Month	Aug

p_Year =UserResponse("Year")
p_Month =UserResponse("Month")



Best Practice

Change Logs

- Document who modified the report and when
- This could be added to your title page
 - ♦ Include another 3 to 4 column table
 - ▶ Include document version, date, author, description

Change Log			
Version	Date	Author	Description
1.0	12-1-2013	Alan Mayer	Created initial document
1.1	12-3-2013	Alan Mayer	Revised format based on Vulcan guidelines



Best Practice

Some report writers keep this information in a separate system. Making it part of the report is very convenient.

Prompts

- Embed prompts as repeated headers
 - ♦ Create a variable that will contain prompt values
 - ♦ Repeat that cell on every page
 - ▶ **Format Cell > Layout > Repeat on every page**

Resort Revenue for FY2006 Aug

Country	Resort	Revenue
France	French Riviera	23,240
US	Bahamas Beach	31,500
US	Hawaiian Club	30,200

= "Resort Revenue for " + [p_Year] + " " + [p_Month]



Best Practice

Demonstration - Annotations



What We'll Cover

- Introduction
- Annotated Report Standards
- Calculation Contexts
- Layered Variables
- Interface Points
- Wrap up

From the Book of Vulcan

- Precise logical control allows us to rise above chaos



What is a Calculation Context?

- Represents the environment in which a formula is calculated
- Similar to the English definition
 - ♦ Circumstances around an event, saying or idea
- Example:
 - ♦ Spock:
 - ▶ “I would advise no action at this time”
 - ♦ Spock in context:
 - ▶ “I would advise no action at this time with the 10 Klingon ships that surround us”



Why are Calculation Contexts Important?

- If you do not define the context of your formulas, BusinessObjects will

Year	Quarter	Month	Sales revenue
2004	Q1	1	\$1,003,541
2004	Q1	2	\$630,073
2004	Q1	3	\$1,027,085
2004	Q2	4	\$895,260
2004	Q2	5	\$865,615
2004	Q2	6	\$517,819
2004	Q3	7	\$525,904
2004	Q3	8	\$173,756
2004	Q3	9	\$668,181
2004	Q4	10	\$655,206
2004	Q4	11	\$484,024
2004	Q4	12	\$649,350
		Sum:	\$8,095,814

\$36,387,203



Caution

Calculation engine differences between versions make this painfully clear!

Calculation Context Syntax

- Examine the following formula:

sum([Sales revenue] In Body) In Report

Input Context

Output Context

- Every formula has an Input and Output Context
- Each of these contexts will be fully explored in the slides that follow



Calculation Context Keywords

1 Report 1

2

Year	Quarter	Month	Sales revenue
2004	Q1	1	\$1,003,541
2004	Q1	2	\$630,073
2004	Q1	3	\$1,027,085
2004	Q2	4	\$895,260
2004	Q2	5	\$865,615
2004	Q2	6	\$517,819
2004	Q3	7	\$525,904
2004	Q3	8	\$173,756
2004	Q3	9	\$668,181
2004	Q4	10	\$655,206
2004	Q4	11	\$484,024
2004	Q4	12	\$649,350
		Sum:	\$8,095,814

3

1 Report

- Everything on a report page

2 Block

- Everything in a table or chart

3 Body

- Everything on a row
- CurrentPage
- All content on the current report page



Calculation Context Keywords, cont'd

4

2004

5

Quarter	State	Sales revenue
Q1	California	\$519,220
	Colorado	\$131,797
	DC	\$208,324
	Florida	\$137,530
	Illinois	\$256,454
	Massachuse	\$92,596
	New York	\$555,983
	Texas	\$758,796
Q1	Sum:	\$2,660,700

4

Section

- Everything within a report section

5

Break

- Everything inside a report break



Building
Blocks

Input Context

sum([Sales revenue] In Body) In Report

Input Context



Building
Blocks

- **Input Context follows these rules:**
 - ♦ It's always found **INSIDE** the parens of the aggregating function
 - ♦ It tells **WHAT** to calculate the sum of
 - ♦ In other words, it determines the level of aggregation
- **The BODY keyword is used to include all dimensions on the row**
 - ♦ Individual dimensions could be listed in parens instead
 - ▶ **Example: In ([Year];[Quarter])**

Individual dimensions must be separated by semicolons

Using ForEach()

- The ForEach operator adds dimensions to the default context
 - The following are the same for the block below:
 - ▶ Sum([Sales Revenue] In Body)
 - ▶ Sum([Sales Revenue] In ([Year];[Quarter]))
 - ▶ Sum([Sales Revenue] ForEach ([Quarter]))

Year	Quarter	Sales revenue
2004	Q1	\$2,660,700
2004	Q2	\$2,278,693
2004	Q3	\$1,367,841
2004	Q4	\$1,788,580
		\$8,095,814



Tip

ForEach is often used for contexts with a long list of dimensions. IN would work as well but all dimensions would have to be specified

Using ForAll()

- The ForAll operator removes dimensions from the default context
 - ♦ The following are the same for the block below:
 - ▶ Sum([Sales Revenue] In ([Year])
 - ▶ Sum([Sales Revenue] ForAll ([Quarter]))

Year	Quarter	Sales revenue
2004	Q1	\$2,660,700
2004	Q2	\$2,278,693
2004	Q3	\$1,367,841
2004	Q4	\$1,788,580
		\$8,095,814



Tip

ForAll works but is not used as often. When in doubt, the IN operator will always work!

Visualizing Input Contexts

- Draw an imaginary table with all dimensions specified by the input context
- Add your formula as the last column to that table
- The value of that formula per row is **WHAT** you are aggregating!

sum(<Sales revenue> In ([Year]; [Quarter]))



Year	Quarter	Sales revenue
2004	Q1	\$2,660,700
2004	Q2	\$2,278,693
2004	Q3	\$1,367,841
2004	Q4	\$1,788,580
		\$8,095,814



Key Feature

Output Context

sum([Sales revenue] In Body) In Report

Output Context



- **Output Context follows these rules:**
 - ♦ It's always found **OUTSIDE** the parens of the aggregating function
 - ♦ It tells **HOW MANY** rows to consider
- **In the above example, consider all rows in the report**
 - ♦ Individual dimensions could be listed in parens instead
 - ▶ **Example: In ([Year])**

Output Contexts using Where()

- The Where() operator allows a calculated range of rows
 - ♦ Add a simple filter:
 - ▶ **Sum([Sales Revenue]) Where ([Month] <= 6)**
 - ▶ **Sum([Sales Revenue]) Where ([Month] <= 6 and [Year] = "FY2004")**

Year	Month	Monthly Revenue	Half Year Revenue
2004	1	\$1,003,541	4,939,392.9
2004	2	\$630,073	4,939,392.9
2004	3	\$1,027,085	4,939,392.9



Tip

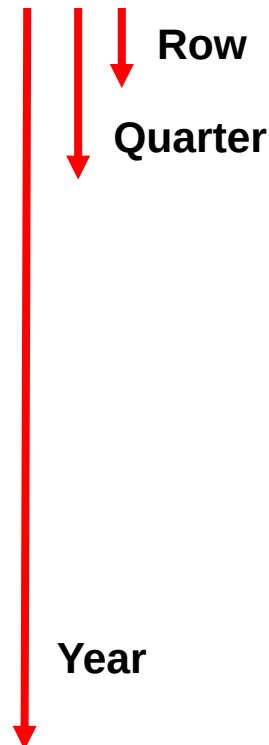
More complicated conditions can be calculated using variables within the Where() operator

Visualizing Output Contexts

- Revisit that imaginary table draw from the input context
- Output contexts determine how many rows to aggregate over

Year	Quarter	Month	Sales revenue
2004	Q1	1	\$1,003,541
2004	Q1	2	\$630,073
2004	Q1	3	\$1,027,085
2004	Q2	4	\$895,260
2004	Q2	5	\$865,615
2004	Q2	6	\$517,819
2004	Q3	7	\$525,904
2004	Q3	8	\$173,756
2004	Q3	9	\$668,181
2004	Q4	10	\$655,206
2004	Q4	11	\$484,024
2004	Q4	12	\$649,350

Sum revenue by:



Key Feature

Very common way to represent subtotals

Visualizing Input and Output Contexts

- Using both completely cover the entire report

INPUT Context

WHAT to aggregate



Year	Quarter	Month	Sales revenue
2004	Q1	1	\$1,003,541
2004	Q1	2	\$630,073
2004	Q1	3	\$1,027,085
2004	Q2	4	\$895,260
2004	Q2	5	\$865,615
2004	Q2	6	\$517,819
2004	Q3	7	\$525,904
2004	Q3	8	\$173,756
2004	Q3	9	\$668,181
2004	Q4	10	\$655,206
2004	Q4	11	\$484,024
2004	Q4	12	\$649,350

OUTPUT Context

Over HOW MANY ROWS



Key Feature

This one visualization will serve as a trusted compass whenever contexts are concerned.

Demonstration – Calculation Contexts



What We'll Cover

- Introduction
- Annotated Report Standards
- Calculation Contexts
- Layered Variables
- Interface Points
- Wrap up

From the Book of Vulcan

- Irrational fear is the hallmark of an undisciplined intellect
- Construct your understanding layer by layer in order to achieve the desired result.



Layered Variables

- Concept that works well for complex logic
- Break up the formula into simpler pieces ...
- ... then use those pieces to build the final variable
- Let's look at an example using sample data
 - ♦ Island Resorts Marketing
 - ♦ Group invoices based on date ranges
 - ♦ Very similar to aging

Creating Constants

- Programming constants makes the report more maintainable
 - ♦ Base date is set to some date
 - ▶ Usually the current date for most companies
 - ♦ Buckets defined by number of days within that range
 - ♦ Each bucket begins where last bucket ends

Variable	Formula
c_BaseDate	=ToDate("2006-01-01";"yyyy-MM-dd")
c_Bucket1	=30
c_Bucket2	=30
c_Bucket3	=30



Tip

Constants could be programmed with prompts for even more interactivity

Define Bucket Start and End Dates (Layer 1)

- Dates for Bucket 1 are shown below
- Buckets 2 – 3 and the Remainder bucket would be defined the same way
- Note how these formulas use the constants previously defined

Variable	Formula
v_Bucket1_Begin	= <code>[c_BaseDate]</code>
v_Bucket1_End	= <code>RelativeDate([v_Bucket1_Begin];[c_Bucket1] – 1)</code>



Note

The Remainder bucket picks up all invoices after Bucket3 through the end of the year

Define Indicators (Layer 2)

- Indicators determine what bucket an invoice belongs in
- The Indicator for Bucket 1 is shown below

Variable	Formula
v_Bucket1_Indicator	=If [Invoice Date] Between([v_Bucket1_Begin];[v_Bucket1_End]) Then 1 Else 0

The formula almost reads like a sentence. Easier to edit and maintain.

Create the Buckets (Layer 3)

- Create the Bucket Totals as the last step
- The formula uses Indicators to make the decision easier

Variable	Formula
v_Bucket1_Total	=Sum(If [v_Bucket1_Indicator] = 1 Then [Revenue] Else 0)

Alternatives to layered logic

- **Complicated logic can be pushed back to the universe**
 - ♦ Best practice where possible
 - ♦ Many reports can benefit from the same central definitions
 - ♦ Can push back even further to HANA models / views
- **Trading database performance for centralized definitions**
 - ♦ This may not be acceptable for certain projects



Decision Point

Demonstration – Layered Variables



What We'll Cover

- Introduction
- Annotated Report Standards
- Calculation Contexts
- Layered Variables
- Interface Points
- Wrap up

From the Book of Vulcan

- Seek multiple ways to the same end
- Reuse that which is valuable, for its value is measured by that use



Invoking Web Intelligence Reports

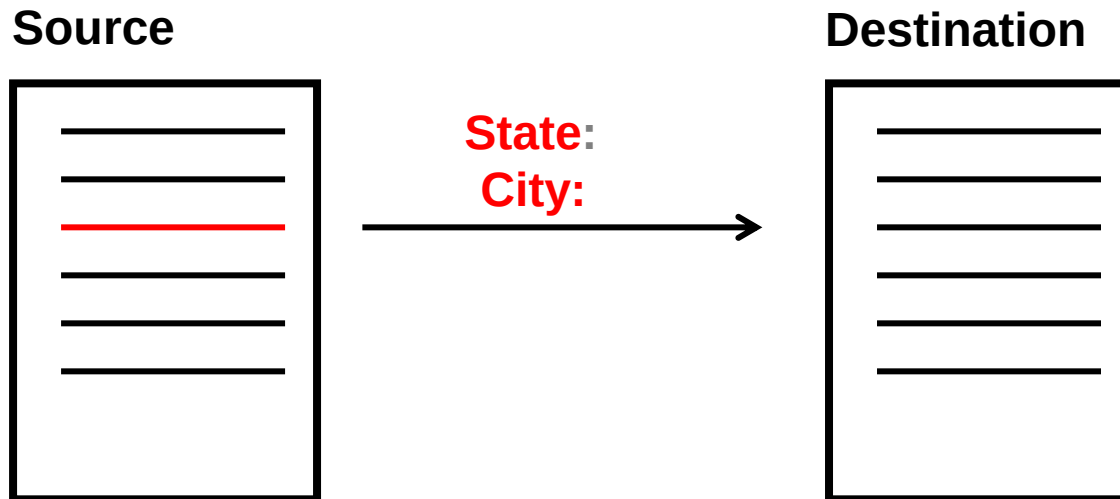
- **We'll investigate three methods**
 - ♦ From another BusinessObjects report
 - ♦ From within a BI Workspace
 - ♦ From another application



BI Workspaces can
consume portions of
Webi reports

Creating Document Links

- Documents can be linked to other via prompts
 - ♦ Links are created in the source document
 - ♦ Those links answer prompts from a second document



Examine the Source Document

- Decide which area should be dynamic
- Cells around that area become the “context”

	2006			
	Q1	Q2	Q3	Q4
e-Fashion Austin	\$314,430	\$273,608	\$294,798	\$252,644
e-Fashion Boston Newbury	\$220,301	\$220,528	\$237,464	\$208,877

Context: Year = 2006
Quarter = Q2
Store = e-Fashion Austin

Create the Destination Document

- Add prompts based on the context previously described
- Any additional detail can be added
 - ◆ Format can be graphical or tabular

The screenshot displays the SAP Query Designer interface, divided into two main sections: **Result Objects** and **Query Filters**.

Result Objects: This section contains five objects: **Year/week** (highlighted with a blue star icon), **Discount**, **Quantity sold**, **Margin**, and **Sales revenue**. Each object has a small icon to its left.

Query Filters: This section contains three filter conditions, each with a blue cube icon, a field name, a comparison operator, and a value field. The conditions are:

- Year** **Equal to** **Year**
- Quarter** **Equal to** **Quarter**
- Store name** **Equal to** **Store**

The filter conditions are connected by an **AND** operator. Each filter condition has a small icon to its right, and the entire filter section is enclosed in a dashed border.

Link Source to Destination

- Right-click on the selected area in the Source document
 - ♦ Linking > Add document link

The screenshot shows the 'Link to document' dialog box with the following fields and sections:

- Name:** Dash 6 - Link Destination. A 'Browse...' button is next to it. Callout 1 points to this button.
- Hyperlink properties:**
 - ☐ Use complete URL path to create hyperlink
 - ☒ Refresh on open. Callout 2 points to this checkbox.
 - ☐ Link to document instance
 - ☐ Target area within the document
- Document prompts:**
 - Year: =[Year]
 - Quarter: =[Quarter]
 - Store: =[Store name]Callout 3 points to these fields.
- Customize the look and behavior of the hyperlink:**
 - Document format: Default
 - Target window: New window. Callout 4 points to this dropdown.
 - Tooltip: Click for more details

Four callout boxes with arrows point to the numbered circles:

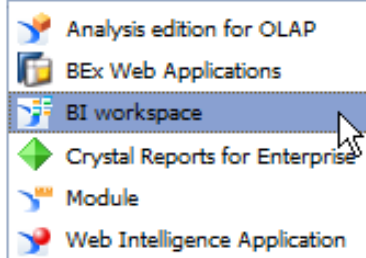
- Callout 1: Locate the document to link to
- Callout 2: Refresh that document
- Callout 3: Map prompts from that document to objects in this one
- Callout 4: Choose a new or existing window to display

Demonstration – Document Links



Working With BI Workspaces

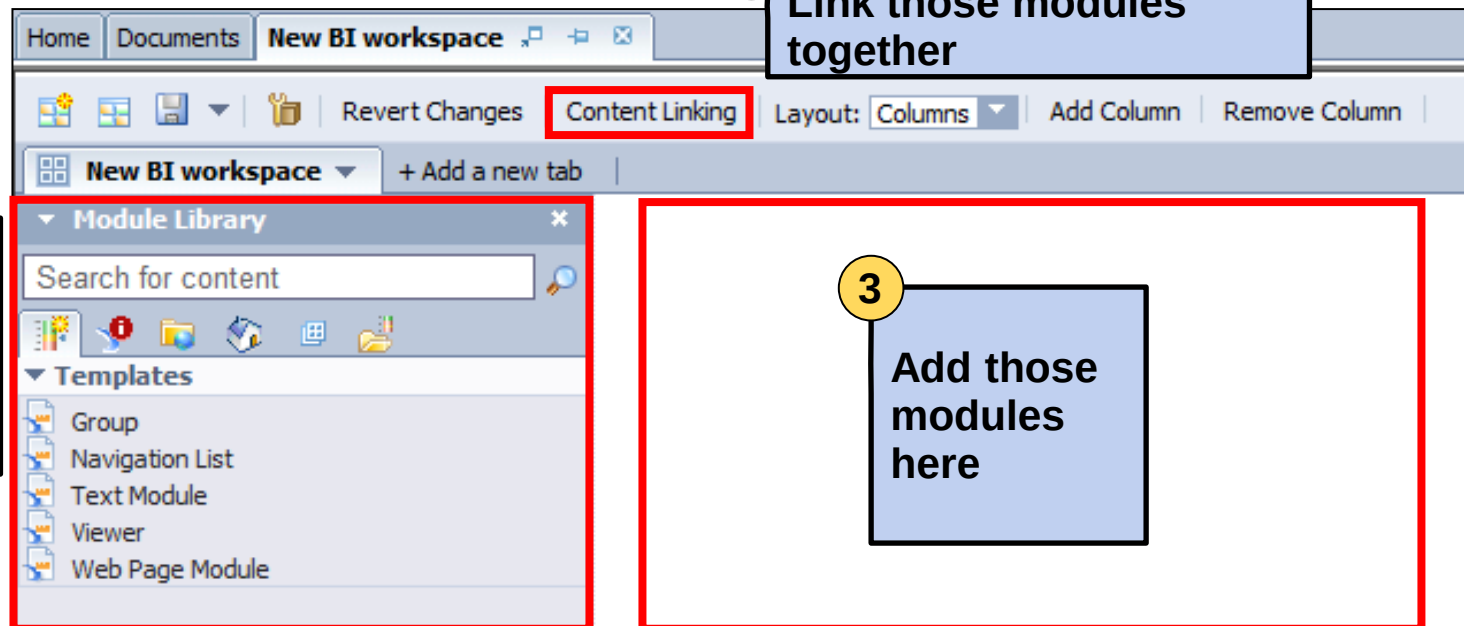
Applications ▾



1
Start the Editor

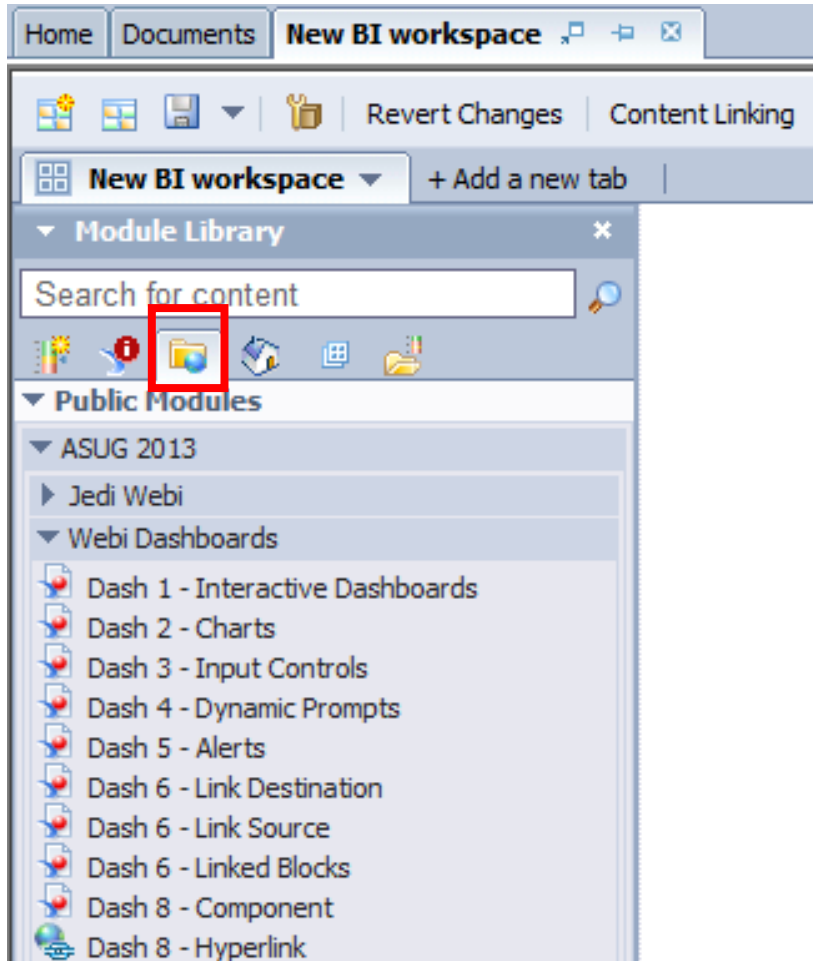
4
Link those modules together

2
Choose your content as modules



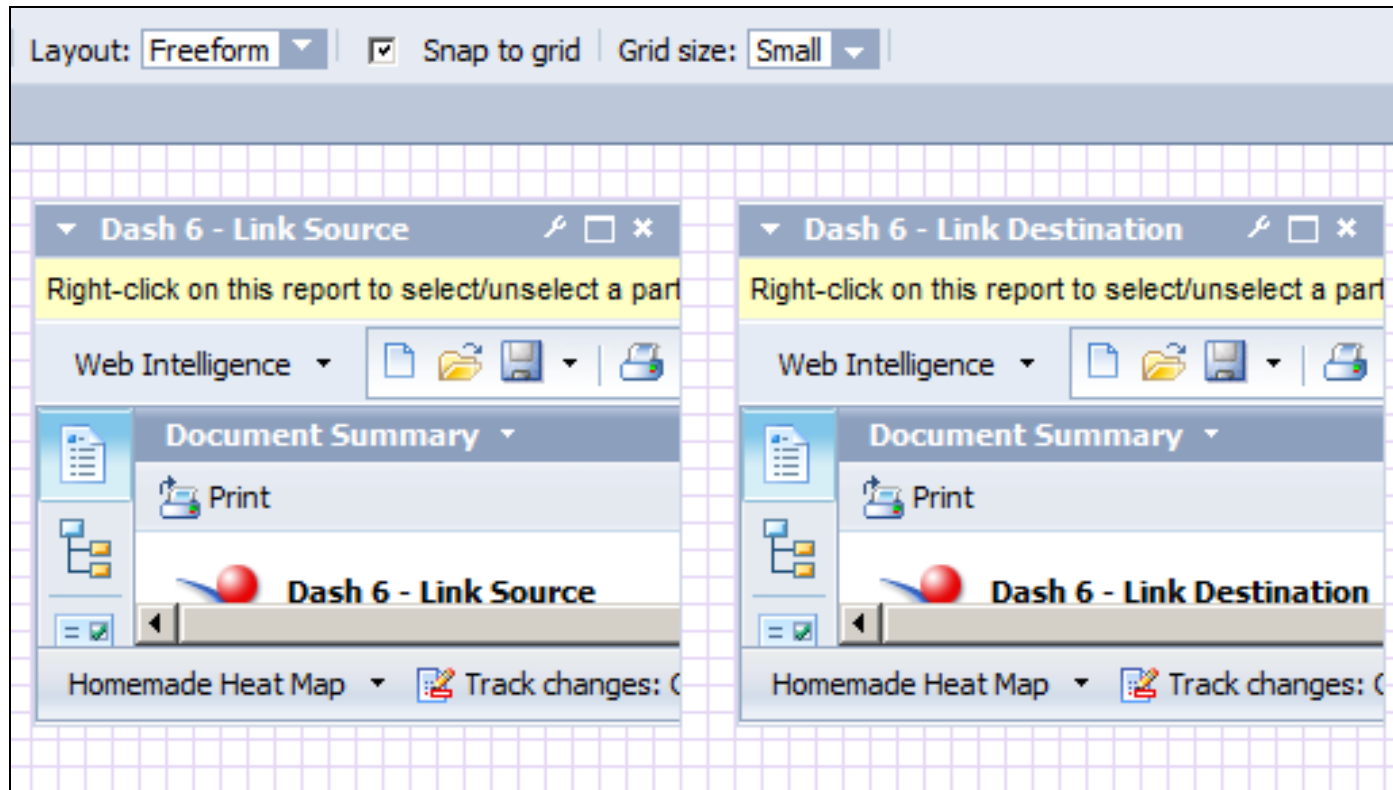
3
Add those modules here

Adding Content



- **Modules control content**
 - ♦ Many options to choose from
 - ♦ We'll focus on Public reports

Arranging Content



Documents are dragged to the workspace from the Module Library

Selecting Report Pieces

Piece 1

	2006	
Right click to select this block as linked report part		Q2
e-Fashion Austin	\$314,430	\$273,608
e-Fashion Boston Newbury	\$220,301	\$220,528

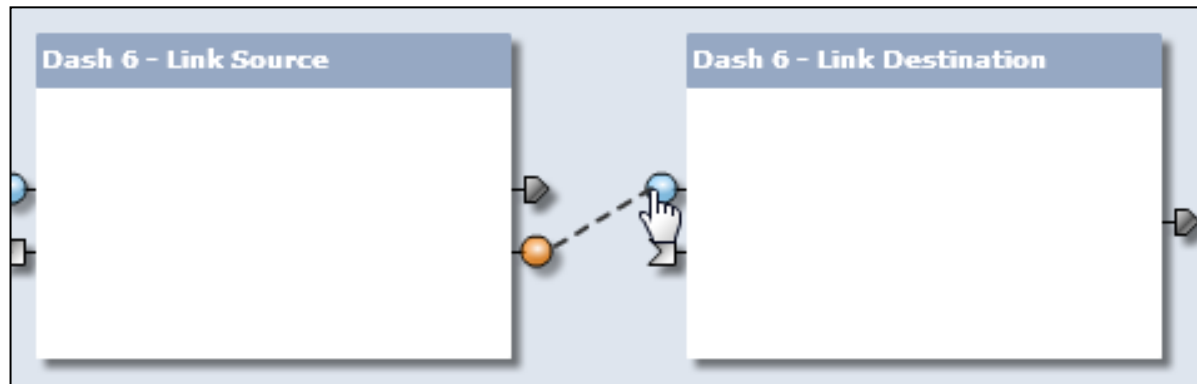
Piece 2

Year/week	Sales revenue	Quantity sold	Discount
2006/01	\$24,709	144	\$8,046
2006/02	Right click to select this block as linked report part		\$10,707
2006/03	\$22,916	148	\$10,654
2006/04	\$30,693	224	\$19,315

Resize each module and
select only the pieces
you need

Link the Pieces

- Use Content Linking to define how one module will drive the others



1

Create a connection between modules by dragging

Parameter Mapping	
Source Parameter	Target Parameter
Store name Type <i>STRING</i>	<i>Click here to select the Target parameter.</i>
Year Type <i>STRING</i>	<i>Click here to select the Target parameter.</i>
Quarter Type <i>STRING</i>	<i>Click here to select the Target parameter.</i>

2

Map parameters (prompts) between modules

Demonstration – BI Workspaces



Launching Reports From Other Programs

- **What about integrating Webi content from existing programs?**
 - ♦ Other than BI launch pad, Infoview, ...
- **Common request for custom portals**
 - ♦ An HTML application developed by another company
 - ♦ The initial user interface is handled by that application
 - ♦ All prompt values for the report are gathered
 - ♦ Webi reports are invoked through OpenDocument commands



Heads-Up

A BusinessObjects session must be previously established before attempting this!

Creating the OpenDocument Command

- The OpenDocument command looks like a hyperlink
 - ♦ What appears below is a sample
 - ♦ Argument values in red

```
http://boxi4win05:8080/BOE/OpenDocument/opendoc/openDocument.jsp?  
iDocID=AajYahfR9Z9Gh_BSbOiQNZM  
&sIDType=CUID  
&sType=wid  
&sRefresh=Y  
&lsMYear=2006
```

Document Arguments

- **iDocID** – Uniquely identifies the document
 - ♦ Example: iDocD=AajYahfR9Z9Gh_BSbOiqNZM
- **sDocName** – Document name (may not be unique)
 - ♦ Example: sDocName=2006+Sales
- **sIDType** – The type of ID supplied (CUID, InfoObjectID)
 - ♦ Example: sIDType=CUID
- **sType** – The type of document (wid = Webi)
 - ♦ Example: sType=wid
- **sInstance** – Opens the latest instance (User, Last, Param)
 - ♦ Example: sInstance=Last
- **sRefresh** – Refresh the document before viewing (Y,N)
 - ♦ Example: sRefresh=Y

Prompt Arguments

- **IsC – Selects a universe context**
 - ♦ Example: IsC=Rentals
- **IsS<prompt> - Single value for a prompt**
 - ♦ Example: IsSYear=2006
- **IsM<prompt> - One or more values for a prompt**
 - ♦ Example: IsMCountries=USA;Germany
- **IsR<prompt> - Range of values for a prompt**
 - ♦ Example: IsRDates=[Date(2013,07,01)..Date(2013,07,15)]

Last Words of Advice

- Use the plus sign to cover spaces in the URL
 - ♦ Example: sDocName=2006+Sales
- Remove any trailing spaces in a prompt
 - ♦ Example: Prompt is “Choose a year: ”
 - ♦ Change to:
 - ▶ **IsSChoose+a+year:**
- The length of the URL may be limited by the browser
 - ♦ For Internet Explorer: 2083 chars

Demonstration – Invoking from Another Application



What We'll Cover

- Introduction
- Annotated Report Standards
- Calculation Contexts
- Layered Variables
- Interface Points
- Wrap up

7 Key Points to Take Home

- Several Web Intelligence techniques can be hard to master
- Think about them logically vs. emotionally
- Annotating report elements helps start that process by providing standard “handles” for more advanced techniques
- Good naming conventions also standardize your report development
- Develop fully-formed calculation contexts to future-proof reports
- Creating variables in layers allows much more complex logic
- Reuse entire reports or portions using interface techniques

Where to Find More Information

- Dave Rathbun, “Dave’s Adventures in Business Intelligence”,
<http://www.dagira.com/category/report-techniques/web-intelligence/>
- Michael Welter, “Michael’s BI Safari”,
<http://michaelwelter.wordpress.com/category/web-intelligence/>
- Brogden, Sinkwitz, Holden, Marks, Orthous, “SAP BusinessObjects Web Intelligence: The Comprehensive Guide”, (SAP Press August 2012)
- <http://help.sap.com>
 - ♦ Follow Analytics → Business Intelligence → Web Intelligence

Your Turn!



Questions?

Alan Mayer
214-295-6250 (office)
214-755-5771 (mobile)
alan.mayer@solidgrounded.com
Twitter: @solidgrounded

Please remember to complete your session evaluation

Disclaimer

SAP, R/3, mySAP, mySAP.com, SAP NetWeaver®, Duet™®, PartnerEdge, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Wellesley Information Services is neither owned nor controlled by SAP.