XDI RDF Cell Graphs

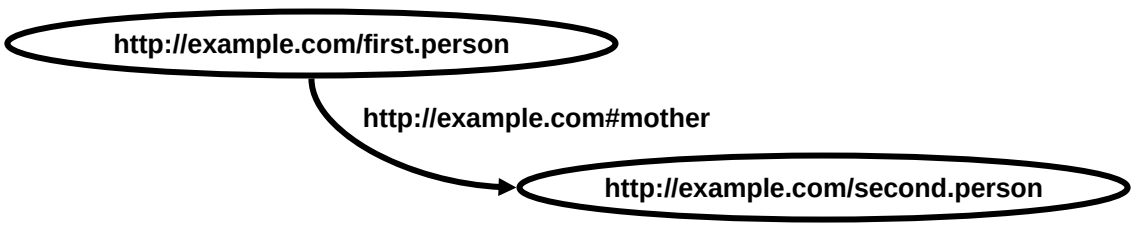
V3 2010-01-13

This document introduces a notation for graphing XDI RDF statements called *cell graphing*. The motivation is to have an easy, intuitive way to illustrate the special capability of XDI RDF to express context, i.e., the ability for RDF graphs to contain other RDF graphs to any depth.

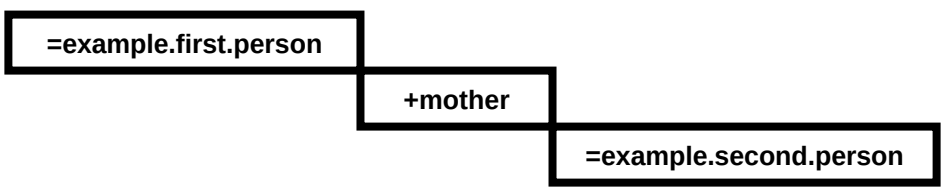
In XDI cell graphs, every RDF node and every RDF arc is represented by a cell (shown visually as a rectangle, similar to a spreadsheet cell). The reason cells are used is that with XDI addressing, every segment of an XRI structured identifier (whether an XDI subject, predicate, or object) may itself represent an RDF graph. So a complete XDI address may represent the union of dozens of component RDF graphs. In fact the only XDI addresses that do not represent RDF graphs (because they are graph primitives) are: a) single XRI delimiter characters, and b) literals. See Appendix A for more about these.

Below is an illustration of a conventional RDF graph alongside an analogous XDI RDF cell graph.

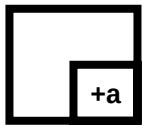
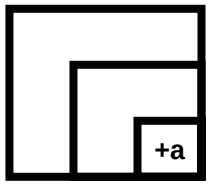
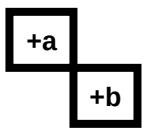
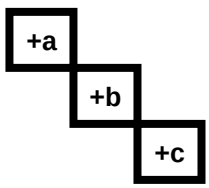
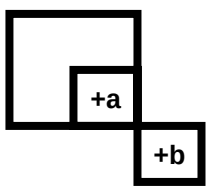
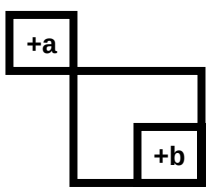
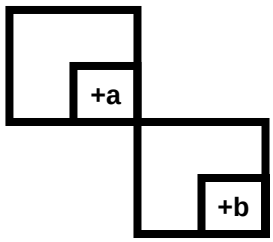
Conventional RDF Graph

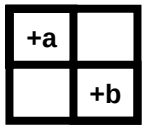
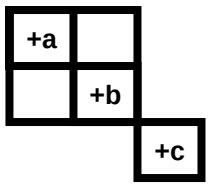
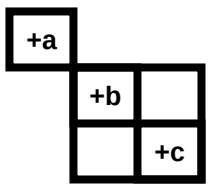
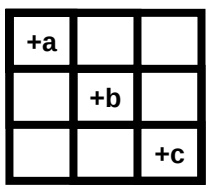


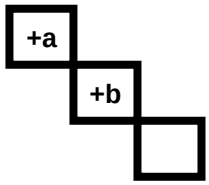
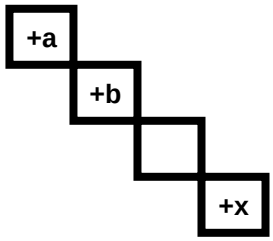
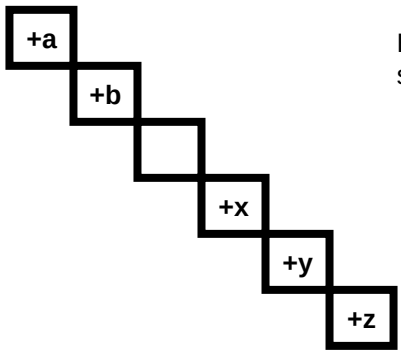
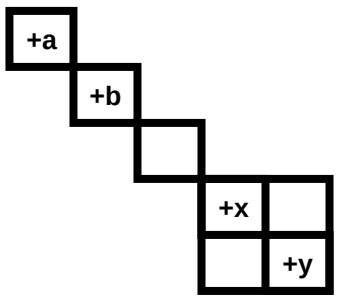
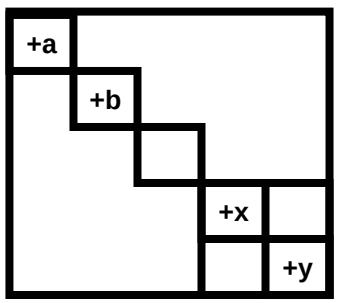
XDI RDF Cell Graph

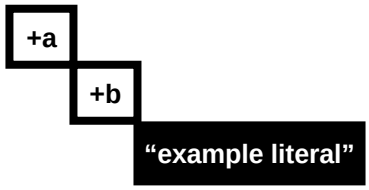
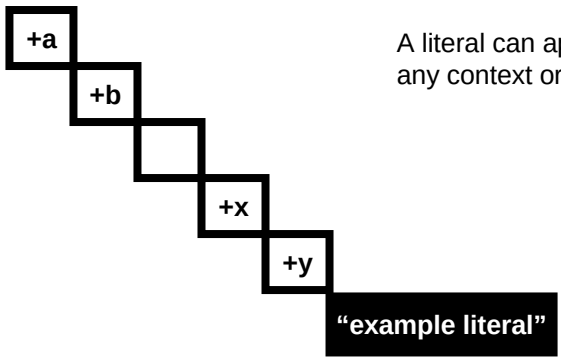
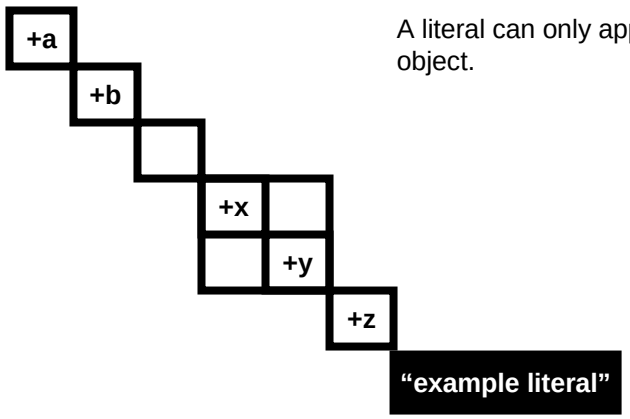


The following pages illustrate the rules of XDI cell graphing notation using examples based on simple two-character XRIs. Note that some of the examples are relatively unusual XDI statements that would rarely appear in actual practice, but are helpful for illustrating how cell graphs work.

	XDI RDF statement	Cell graph	Notes
A1	**+a**		A single cell in the left-hand column represents an XDI RDF subject node.
A2	**(+a)**		A cell around a cell is reification. The outer cell represents the XDI context containing the node as the inner cell. The outer cell is the new XDI subject node.
A3	**((+a))**		Nested reification (nested cross-references) are graphed as nested cells. Each is a context containing a subcontext. This pattern can be repeated to any length.
A4	**+a/+b**		An XDI RDF predicate is shown diagonally, in the second column, down one row, similar to the visual structure of X3 Simple syntax.
A5	**+a/+b/+c**		An XDI RDF object is shown in the third column, down a row, similar to X3 Simple syntax. You cannot put more than three cells in diagonal without starting a XDI subcontext.
A6	**(+a)/+b**		A reified XDI subject is still a first-column cell, and thus acts like any other XDI subject.
A7	**+a/(+b)**		A reified XDI predicate is still a second-column cell, and thus acts like any other XDI predicate.
A8	**(+a)/(+b)**		A reified XDI subject with a reified XDI predicate.

XDI RDF statement	Cell graph	Notes
B1 **+a/$has/+b** **(+a/+b)** **+a+b**		A reified subject/predicate relationship results in a new XDI subject. All $has statements express reified subject/predicate relationships. See Appendix B for more about $has graphs.
B2 **(+a/+b)/+c** **+a+b/+c**		A **+c** predicate added to the **+a+b** subject node above.
B3 **+a/(+b/+c)** **+a/+b+c**		A **+b+c** predicate on a **+a** subject node.
B4 **(+a/+b)/$has/+c** **+a+b/$has/+c** **((+a/+b)/+c)** **(+a+b/+c)** **+a/$has/(+b/+c)** **+a/$has/+b+c** **(+a/(+b/+c))** **(+a/+b+c)** **+a+b+c**		A reification of either #B2 or #B3 above. In both cases, the result is a single XDI subject cell (the outer cell) with the same non-blank cells, **+a**, **+b**, and **+c**, along the diagonal. This means all the different XDI statements on the far right produce the same cell graph and are therefore equivalent. In this respect, XDI **$has** statements are associative. This pattern holds no matter how deeply you nest the **$has** statements.

XDI RDF statement	Cell graph	Notes
C1 **+a/+b//**		When a blank node is specified as the object of a cell graph, it may be used to begin a subcontext (nested cell graph).
C2 **+a/+b//+x**		Just like with X3, the rules for the cell graph of a subcontext are identical for those of a parent context.
C3 **+a/+b//+x/+y/+z**		Here the context **+a/+b//** contains the subcontext statement **+x/+y/+z**.
C4 **+a/+b//(+x/+y)** **+a/+b//+x+y**		Reification works identically at all levels of contexts and subcontexts.
C5 **(+a/+b//(+x/+y))** **(+a/+b//+x+y)**		Any valid XDI statement may be reifiied– even it it crosses multiple XDI contexts. For example, this outer cell is now a new XDI subject.

XDI RDF statement	Cell graph	Notes
D1 **+a/+b**		A literal is represented by a black cell with white text. Note that the address of this data block is the XDI statement that addresses this block.
D2 **+a/+b//+x/+y**		A literal can appear as any XDI object in any context or subcontext.
D3 **+a/+b//(+x/+y)/+z** **+a/+b//+x+y/+z**		A literal can only appear as an XDI object.

XDI RDF statement	Cell graph	Notes

E1

+a/+b/+c

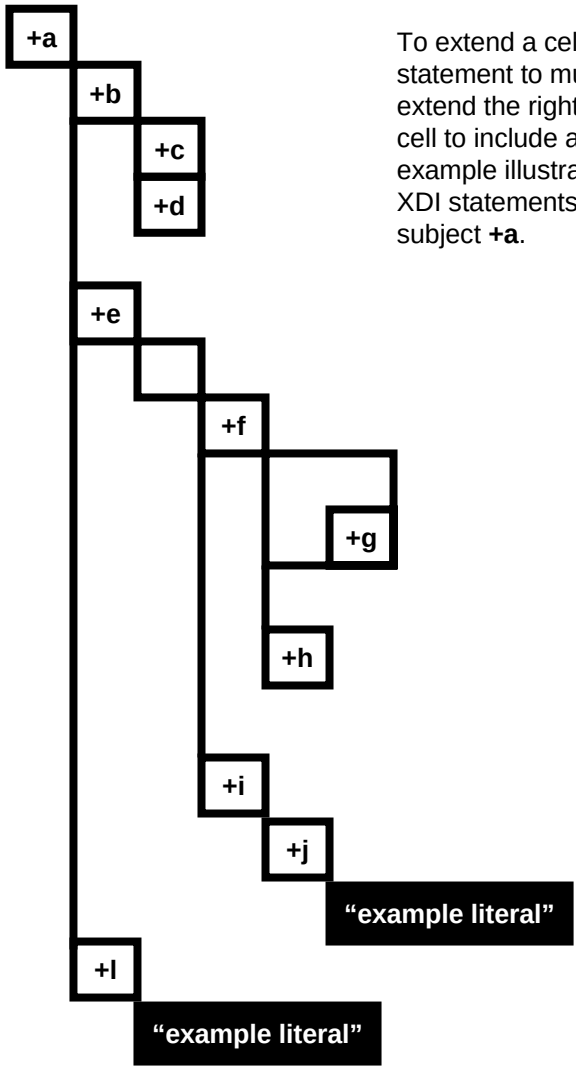
+a/+b/+d

+a/+e//+f/(+g)

+a/+e//+f/+h

+a/+e//+i/+j

+a/+l



To extend a cell graph from a single XDI statement to multiple XDI statements, extend the rightmost border of any parent cell to include all its children. This example illustrates a graph of six different XDI statements originating from the XDI subject **+a**.

Appendix A: Cell Graphs of XDI Primitives

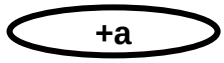
Each of the atomic XRI subsegments shown inside the cells of a cell graph actually represents an XDI $has statement between two XDI primitives: an XRI subsegment delimiter character and an XRI literal string.

To put this in RDF terms, the XDI RDF subject **+a** (shown in X1) actually represents the XDI RDF statement **+/$has/a** (shown as an RDF graph in X2 and as a XDI cell graph in X3).

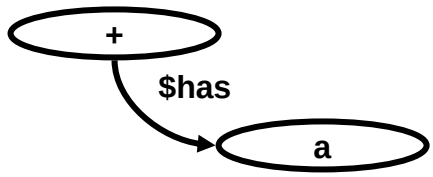
Since a **$has** statement represents a reification of an XDI subject/predicate relationship, the cell graph in X3 can be simplified to the cell graph in X4. X4 can in turn be simplified to the single cell graph in X5 representing an atomic XRI subsegment.

A different way of visualizing the atomic XDI statement **+a** shown in graphs X1 through X5 is to see it as the arc between: a) the implicit blank node representing the root of the XDI context in which **+a** is a subject, and b) the actual RDF subject node identified by the XRI **+a**. This is illustrated in X6.

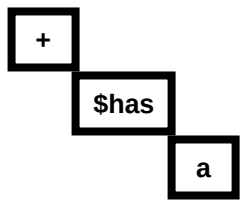
X1



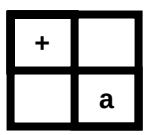
X2



X3

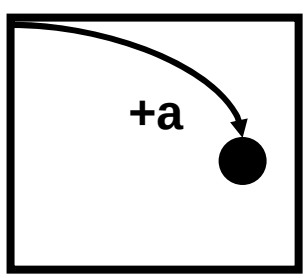


X4



X5

X6



Appendix B: $has and hasa Statements

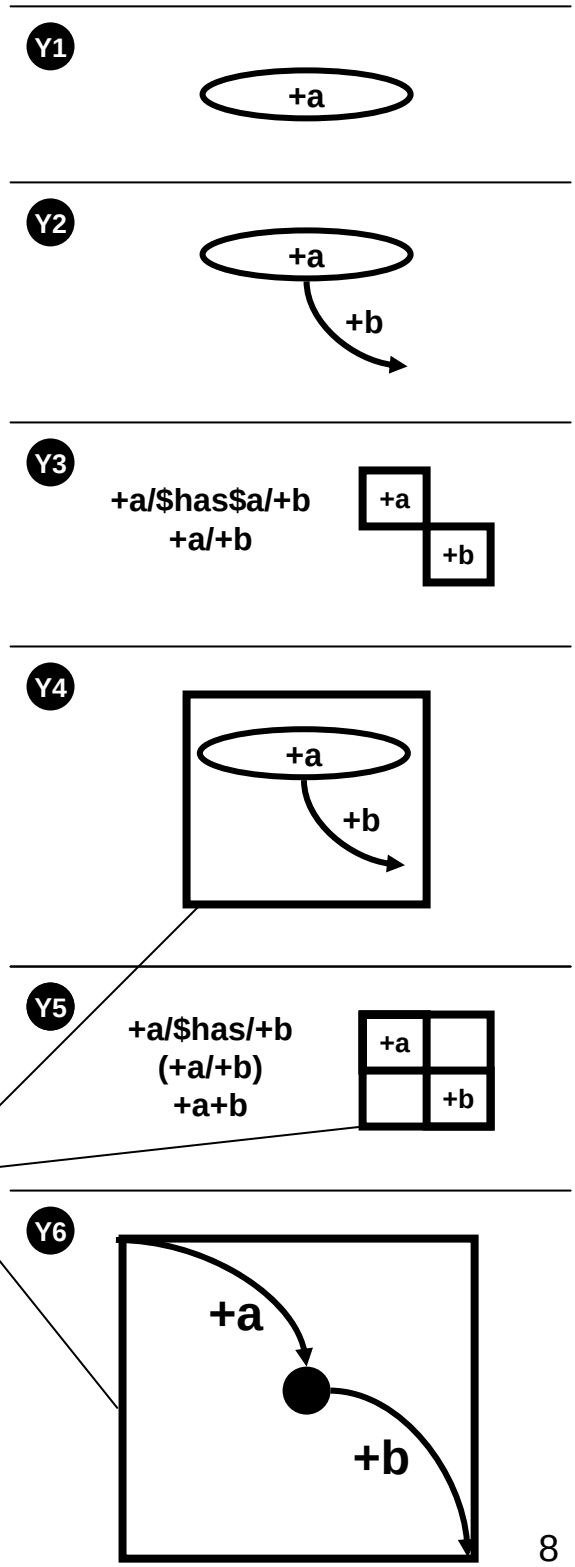
An XDI RDF **$has** statement is in many ways the basic building block of XDI RDF graphs. However from a pure RDF standpoint it is also one of the most difficult to understand because it has no corollary in RDF.

To illustrate this, Y1 is a conventional RDF graph of a single RDF subject node **+a**. Y2 is a conventional RDF graph of that same subject node plus a single outgoing arc **+b**.

Y3 is how the conventional RDF graph in Y2 is described using an XDI RDF **hasa** statement. It says, "subject **+a** has a arc **+b**". This reduces to the XDI RDF statement **+a/+b**. **hasa** statements can have cardinality, since there can be a specific number of arcs **+b** originating from **+a**.

A **$has** statement, by contrast, is a statement *identifying the relationship* between an XDI RDF subject node and an XDI RDF predicate arc. To do that, a **$has** statement identifies *the graph containing the relationship*. In Y4, this is depicted as the box drawn around the RDF graph shown in Y2. (Reminder: conventional RDF graph notation has no way to identify a graph itself, or to show graphs within graphs.)

Y5 shows the cell graph equivalent to Y4. Note that it is the same as Y3 except for the addition of a cell enclosing the entire graph. This outer cell, representing a new XDI RDF subject node, is the node identified by the **$has** statement. Y6 is another way of visualizing this using the same approach as X6 in Appendix A.



This outer cell, representing the XDI RDF context containing the graph, is the XDI RDF node is addressed by the XDI statement (+a/+b), which is equivalent to +a+b

Because a **$has** statement identifies a single XDI RDF node, it cannot have cardinality, since there is no assertion of an arc on this node (only an arc within the graph it contains).