Subject-Object[1] Variables Determination

Variable to be set:	{@rel | @rev} *"{@rel | @rev} Mode"[2]*	!{@rel | @rev}	
		{@property} & !{@content | @datatype} *"Special Property Mode"*	**Otherwise** *"Everything Else Mode"*
[new subject]	Choose first existing of: • *@about[3] • if in <root> : [base] • [parent object]	Choose first existing of: • *@about • if in <root> : [base] • [parent object]	Choose first existing of: • *@about • *@resource • *@href • *@src • if {@typeof} : new bNode[4] • if in <root> : [base] • [parent object] if !{@property} then [skip element] = true[5]
[current object resource]	Choose first existing of: • *@resource • *@href • *@src • if ({@typeof} & !{@about}) : new bNode	*Note:* *if {@typeof} : almost the same as in {@rel | @rev} Mode[6]* *if !{@typeof} : same as in "Everything Else Mode."*	*Not set. [current object resource] remains null.*
[typed resource] (ONLY if {@typeof})	if {@about} : [new subject] if !{@about} : [current object resource][7]	Choose first existing of:[8] • *@resource • *@href • *@src • new bNode[9] Set [current object resource] = [typed resource]	[new subject]

General Notes:

- There are essentially three major situations, or "modes" here:
  - {@rel | @rev} Mode[10]
  - Special Property Mode
  - Everything Else Mode
- There are three "phases" which occur in this order:
  - Determining [new subject]
  - Determining [current object resource]
  - Determining [typed resource]

- If none of the choices apply then the variable remains set to null.
- RDFa Lite subset attributes and notes are in red.
- The processor behaves exactly the same for RDFa Lite as it does for RDFa Core.
    - The only difference is that the web author chooses to use only this subset of RDFa attributes. RDFa Lite is, basically, nothing more than a teaching tool.
    - This means that all other options (such as [base], [parent object], bNode, @href, and @src) are still in play when determining the values of these three subject-object variables.
- Existing XHTML attributes @href and @src (green) are also used in both RDFa Core and RDFa Lite.
- [new subject] is never actually set to the value of [parent subject].
- Through one means or another, @typeof is prevented from "up-chaining" to the parent object. (It seems odd that these convoluted means were used rather than a more direct route.)

---

[1] I call them "subject-object" variables because these are the variables that will ultimately become subjects or objects in triples. However, for some of them, in some circumstances, we don't know yet whether they will end up being subjects or objects.

[2] This "mode" is never used by RDFa Lite because RDFa Lite does not include @rev or @rel. Therefore, I do not highlight the RDFa Lite attributes in this column.

[3] Remember: The asterisk means "the value of."

[4] This effectively skips the possibility that [typed resource] could ever be set to [base] or [parent object], because [typed resource] is only set if {@typeof}. So, at the least, [typed resource] would be set to a new bNode rather than [parent object], thus eliminating the possibility of @typeof "up-chaining" to a node from the parent context.

[5] What is the purpose of this?

[6] With the only difference being that a new bNode is created regardless of {@about}. Is this true?

[7] This effectively skips over the possibilities to use [base} or [parent object], thus eliminating the possibility of @typeof "up-chaining" to a node from the parent context.

[8] This skips over the possibilities to use *@about, [base], or [parent object], thus eliminating the possibility of @typeof "up-chaining" to a node from the parent context

[9] Apparently, regardless of {@about} Is this the intention? Note that having this in the [typed resource] phase presumes {@typeof}.

[10] Certainly, I can come up with a better name for this. What is a term that describes and includes ONLY those two attributes?

If {@property} : set [current property value] =

	{@content}	!{@content}		
{@datatype}	• *@content as a plain literal	• Concatenate text only from all descendant text nodes. • (No need to append datatype IRI)	""	*@datatype ==
	• *@content as an XML Literal ??? • Is this combination invalid? Or is *@content just ignored?	• "Serialize to text" all child nodes, including markup, and even markup that looks like RDFa. • Append XMLLiteral datatype.	"XMLLiteral"	
	• *@content as a typed literal • Append datatype IRI.	• Concatenate text only from all descendant text nodes. • Append datatype IRI.	*"anythingElse"*	

!{@datatype}	• *@content as a plain literal	❶ "Special Property Mode" (!{@content | @datatype | @rel | @rev}
		• Choose first existing of:
		  • *@resource
		  • *@href
		  • *@src
		  • Should I stick bNode in here? Or does it then skip down to choice ❸? Or what? What happens if !{@resource | @href | @src}?
			Choose the first available of ❶, ❷, or ❸.	

		❷ If {@typeof} & !{@about} : [typed resource]		
		❸ Any other situation: • Use the "child text" of the current element.		

How to use this table: Divide up the possibilities in decreasing order of number of lines dividing the sections.

- First divide along the vertical triple lines, based on {@content}.
- Then divide along the horizontal double lines, based on {@datatype}. This puts you in one of four quadrants.
- Finally divide along the horizontal single lines.
  - If {@datatype} this tells you how to apply that datatype to either the *@content or the text between the tags.
  - If !{@datatype | @content} this gives you three additional choices based on the existence of other attributes.