Formal Specification of Our Extended Vega-Lite by Hyeok Kim

Supplementary Material to a CHI 2022 paper, "Cicero: A Declarative Grammar for Responsive Visualization"

Notations	Examples	Description
":=": is defined as a tuple of	a := b, c, d	a is defined as a tuple of b, c, and d.
"=": set as value of	A=b	An argument A set as b.
"~": possible names for ... are ...	a ~ b, c	The possible names of a are b and c.
"|": alternate argument	a | b | c	Either one of a, b, or c.
"...": extensible arguments	a := b | c | ...	a can be either b, c, or something else.
"<>": datatype	A<String>	A is a string type argument.
"[]": a list of	<String>[]	A list of string elements.
	<a, b>[]	A list of tuples composed of a and b.
"[0..n]": a list of length 0 to n	[0..5]	A length-5 array.
"?": optional argument	a?	a is optional.
	...?	optional additional arguments

Notes

"<Number>" either a number or a string of a number with its unit (e.g., 350, "350px").

Formal Specification

```
ExVLSpec := Name?, Data, Layout, Layer, Transform?, Interaction?, Title?, NonData?
Name := <String> the name of a visualization
Data := <JSON> the dataset of a visualization

Layout := Width?, Height?, Composition, Row, Column, HAxis?, VAxis?, NColums?, Projection?
Width := <Number>          Height := <Number>
Composition := single | repeated | projection | ...
    the type of a visualization layout; repeated refers to small multiples

Row := <RCItem>[0..2]      Column:= <RCItem>[0..2] row and column items (as used in a Trellis plot)
RCItem:= Field<String> | FieldObject

HAxis := AxisItem          VAxis := AxisItem the design of an X and Y axis, respectively
AxisItem+ := Domain<Boolean>?, DomainColor<Color>?, DomainDash<Dash>?, DomainWidth<Number>?,
    DomainOpacity<Opacity>?, Grid<Boolean>?, GridColor<Color>?, GridDash<Dash>?,  GridWidth<Number>?,
    GridOpacity<Opacity>?, Offset<Number>? https://vega.github.io/vega-lite/docs/axis.html
NColumns := <Integer> the number of columns in a "repeated" composition
Projection+ := ProjectionType, ProjectionScale?, ProjectionTranslate?, ...? the details of a map projection
                                                    https://vega.github.io/vega-lite/docs/projection.html
FieldObject := Field<String>, DataType?, Scale?, Sort?, Aggregate?, Bin? | Aggregate=count, Scale?
    details about a data field encoded to a row/column/channel
DataType := nominal | ordinal | quantitative | temporal
Scale+ := Domain?, Range?, Scheme?, Reverse<Boolean>?, ...? https://vega.github.io/vega-lite/docs/scale.html
Domain := <Any>[]          Range := <Any>[]    Scheme := <String> (e.g., "magma")
Sort+ := ascending | descending | SortBy https://vega.github.io/vega-lite/docs/sort.html
SortBy := SortOrder?, SortField<String>? sort by a certain field
SortOrder := ascending | descending | <Any>[] an ascending, descending or custom order
Aggregate+ := count | mean | max | median | ... https://vega.github.io/vega-lite/docs/aggregate.html
Bin+ := Maxbins?, BinSteps?, Nice?, ...? https://vega.github.io/vega-lite/docs/bin.html

Layer := <LayerItem>[]
LayerItem := Mark, Text?, Tooltip?, Transform?

Mark := MarkType, $MarkProperty?
MarkType := circle | point | bar | rect | ...
$MarkProperty ~ color, shape, size, stroke, ... mark properties or such channels
$MarkProperty := Value | FieldObject if "Value" is used, then it is a static property. Otherwise, it is an encoding channel.
```

```
Text := TextType, TextField, Values?, Anchor?, Orient?, TextItems?, Tick?, TextVisibility? ...?
TextType := on-mark | on-axis | legend mark labels, axis labels, and legends, respectively
TextField := FieldName<String> the reference field of the text item (i.e., text element for each value of the "TextField")
Values := <Any>[] a subset of elements of the TextField to show the labels
Anchor, Orient := start | end | middle | right-start | right-end | ....
    Anchor: the reference position to the corresponding mark/axis element; Orient: the reference position to the text element itself
TextItems := <TextItem>[] each line of the text element
TextItems := Format?, FontColor<Color>?, Width?, ...?
Tick := <Boolean> | TickColor<Color>?, TickWidth?, ...?
    the design of the line segment between the text element and referred visual element.
TextVisibility := External<Boolean>?, Numbering<Boolean>?, Position<top|bottom|left|right>?
    internalization/externalization of mark-labels; If Numbering = true, then reference numbers are shown on corresponding marks.

Tooltip := TooltipVisibility, TooltipFields
TooltipVisibility := on-mark | fixed | hidden the position of a tooltip
TooltipFields := <FieldName, Format>[] the information shown in a tooltip

Transform+ := <TransformItem>[] global/layer-specific data transformation
TransformItem+ := Filter+ | Aggregate+, As+ | Bin+, As+ | Compute+, As+, Op+ | ...
                                                    https://vega.github.io/vega-lite/docs/transform.html
Interaction := <InteractionItem>[] global interactions
InteractionItem := ZoomPan | Context | Filter zoom+pan, brush-based context view, and interactive filter, respectively

Title := Width<Number>?, Align?, TitleItems
TitleItems := <Name<String>?, Text<String>, Align?, FontSize<Number>?, FontWeight?, ...?>[]
    the design and content of each title element

NonData := NonDataVisibility?, NonDataGlobalStyle?, NonDataItems
   annotations and emphases unbound to data
NonDataVisibility := External<Boolean>?, Numbering<Boolean>?, Position<top|bottom|left|right>?
NonDataGlobalStyle := FontFamily<String>?, FontWeight?, BoxStroke<Color>?,
    BoxStrokeWidth<Number>?, LineHeight<Number>, ...? global style of non-data items

NonDataItems := <NonDataType, Name<String>?, X<Number>, Y<Number>, DX<Number>, DY<Number>,
    Width<Number>, Height<Number>, Rotate<Number>, NonDataText, NonDataBox, NonDataMark>[]
    the appearance and contents of nondata items / X, Y: absolute position, DX, DY: relative position
NonDataType := text annotations | mark emphases
NonDataText := <Text<String>, Align?, Overflow?, FontSize<Number>?, FontWeight?, LineHeight<Number?>,
  FontColor<Color>?, Opacity<Number>?>[]
NonDataBox := Padding<Number>?, Fill<Color>?, Stroke<Color>?, StrokeWidth<Number>?, Radius<Number>?
    StrokeStyle?
StrokeStyle := solid | dashed | dotted | ...
NonDataMark := NonDataMarkDef, Fill<Color>?, Opacity<Number>?, Stroke<Color>?,
    StrokeOpacity<Number>?, StrokeWidth<Number>?, Radius<Number>?
NonDataMarkDef := Icon<String> Bootstrap icon names | Image<URI> | Shape
Shape := circle | rect | rule | ...
```

Detailed documents for the following common value types (available as of Feb. 2022).

Number	https://developer.mozilla.org/en-US/docs/Web/CSS/length
Align	https://developer.mozilla.org/en-US/docs/Web/CSS/text-align
Color	https://developer.mozilla.org/en-US/docs/Web/CSS/color
FontWeight	https://developer.mozilla.org/en-US/docs/Web/CSS/font-weight
Dash	https://developer.mozilla.org/en-US/docs/Web/SVG/Attribute/stroke-dasharray
Opacity	https://developer.mozilla.org/en-US/docs/Web/CSS/opacity
LineHeight	https://developer.mozilla.org/en-US/docs/Web/CSS/line-height
Overflow	https://developer.mozilla.org/en-US/docs/Web/CSS/overflow
Icon	https://getbootstrap.com/docs/5.1/content/typography/