SG datatype reference

Contents

[Brief explanation 2](#_Toc401241370)

[Checkbox 3](#_Toc401241371)

[Generic whole number 3](#_Toc401241372)

[Generic float number 3](#_Toc401241373)

[Label without HTML editor 4](#_Toc401241374)

[Label with HTML Editor 4](#_Toc401241375)

[Digital Join Selector 5](#_Toc401241376)

[Analog Join Selector 5](#_Toc401241377)

[Serial Join Selector 5](#_Toc401241378)

[Color Selector 6](#_Toc401241379)

[Alpha Slider 6](#_Toc401241380)

[Style selector with dropdown selection 7](#_Toc401241381)

[Template 7](#_Toc401241382)

[Template 7](#_Toc401241383)

# Brief explanation

The following is an overview on how to make your Custom smart Graphics control customizable by using the property grid in VTPro.

There are 3 files necessary to deploy your objects, a XSD file, XML file and an AS3 file(which will get compiled into a SWF file). The XSD file will determine the property grid, the XML file transports data and initializes your control with startup values, the AS3 file will use the propertyArr array to gather the data from the XML file.

You will typically start from the XSD file to choose your properties, when you start from the template provided with the SDK you will have an entry point at the bottom of the file (under xs:element name = “properties”). Let’s look at an example :

<xs:element name=**"Label"** type=**"xs:string"** title=**"Label"**/>

Adding this in the entry point will generate a new settable property in the property grid in visiontools for your control. There are many arguments that can be put in, but “name”, “type” and “title” are the main 3 that are always required.

Type= This is the type of property that will be created, there are many types which will be covered in the samples below. The xs:string will generate a text entry input that you can double click to type in a piece of text. A use case would be to place a label somewhere in your control.

title= is the text that will be displayed to the left of the type, this is a user friendly name so the visiontools user knows what property he is setting.



The name attribute is your hook to XML and eventually your Custom control, the next step is creating an XML element node in the XML file for “Label”. Again, if you started from the SDK template you will have an entry point under <Properties>

<Label>**Hello World!**</Label>

The value that you fill in for label (here “Hello World!”) is the value that you will initialize the control with, what is left now is to implement the code to read out this value in your ActionScript code, in the AS3 file add this in the initializeScript function:

**var** textLabel**:**String **=** propertyArr**[**"Label"**];**

The propertyArr will read the <Label> XML entry and store it’s value in the textLabel variable. Every time a change happens in the property grid the initializeScript function will be called reinitializing our control with the new values.

# Checkbox

## XSD

<xs:element name=**"DisplayBackground"** type=**"xs:boolean"** title=**"Show Background"**/>

## XML

<DisplayBackground>**true**</DisplayBackground>

## AS3

**var** displayBackground**:**Boolean **=** propertyArr**[**"Name"**]** **?** String**(**propertyArr**[**"Name"**]).**toLowerCase**()** **==** "true" **:** false**);**

# Generic whole number

## XSD

<xs:element name=**"Offset"** type=**"xs:int"** title=**"Offset"**/>

## XML

<Offset>**0**</Offset>

## AS3

**var** offset**:**Number **=** propertyArr**[**"Offset"**];**

# Generic float number

## XSD

<xs:element name=**"Alpha"** type=**"xs:float"** title=**"Alpha"**/>

## XML

<Alpha>**.5**</Alpha>

## AS3

**var** alpha**:**Number **=** propertyArr**[**"Alpha"**];**

# Label without HTML editor

## XSD

<xs:element name=**"Label"** type=**"xs:string"** title=**"Label"**/>

## XML

<Label>**Hello World!**</Label>

## AS3

**var** textLabel**:**String **=** propertyArr**[**"Label"**];**

# Label with HTML Editor

## XSD

<xs:element name=**"HTMLLabel"** type=**"htmlText"** title=**"Label"**/>

## XML

<HTMLLabel>**Hello World!**</HTMLLabel>

## AS3

**var** textLabel**:**String **=** propertyArr**[**"HTMLLabel"**];**

# Digital Join Selector

## XSD

<xs:element name=**"DigitalJoin"** type=**"DigitalJoin"** title=**"Digital Join"**/>

## XML

<DigitalJoin>**0**</DigitalJoin>

## AS3

**var** digitalJoin**:**int **=** propertyArr**[**"DigitalJoin"**];**

# Analog Join Selector

## XSD

<xs:element name=**"AnalogJoin"** type=**"AnalogJoin"** title=**"Analog Join"**/>

## XML

<AnalogJoin>**0**</AnalogJoin>

## AS3

**var** analogJoin**:**int **=** propertyArr**[**"AnalogJoin"**];**

# Serial Join Selector

## XSD

<xs:element name=**"SerialJoin"** type=**"SerialJoin"** title=**"Serial Join"**/>

## XML

<SerialJoin>**0**</SerialJoin>

## AS3

**var** serialJoin**:**int **=** propertyArr**[**"SerialJoin"**];**

# Color Selector

## XSD

<xs:element name=**"BackgroundColor"** type=**"Color"** title=**"Background Color"**/>

## XML

<BackgroundColor>**0x000000**</BackgroundColor>

## AS3

**var** backgroundColor**:**uint **=** propertyArr**[**"BackgroundColor"**];**

# Alpha Slider

## XSD

<xs:element name=**"Alpha"** type=**"Alpha"** title=**"Alpha"**/>

## XML

<Alpha>**255**</Alpha>

## AS3

**var** alpha**:int** **=** propertyArr**[**"Alpha"**];**

# Style selector with dropdown selection

## XSD

<xs:element categoryName=**"Buttons"** name=**"Style"** propertyType=**"Style"** styleType=**"Button"** title=**"Button Style"** type=**"Style"**/>

## XML

<Style></Style>

## AS3

# Template

## XSD

## XML

## AS3

# Template

## XSD

## XML

## AS3