Schema documentation for config.xsd

february 15, 2012

Table of Contents

Namespace: ""

Schema(s)

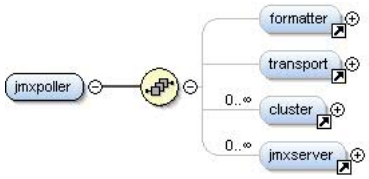
Main schema `config.xsd`

Namespace	No namespace
Properties	attribute form default: `unqualified` element form default: `qualified`

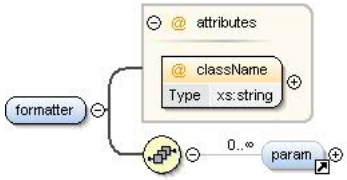
Element(s)

Element `jmxpoller`

Namespace	No namespace

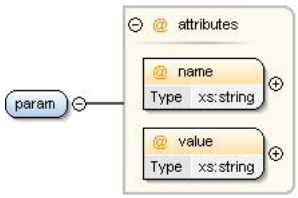
Annotations	Root element of the configuration file. This configuration file is where you specify local and remote JMX servers to connect to across your enterprise and extract whatever MBean attributes you have declared to query. The result will then be written to STDOUT for SPLUNK indexing.
Diagram	
Properties	content: complex
Model	formatter{0,1} , transport{0,1} , cluster* , jmxserver*
Children	cluster, formatter, jmxserver, transport
Instance	<jmxpoller> <formatter className="">{0,1}</formatter> <transport className="">{0,1}</transport> <cluster description="" name="">{0,unbounded}</cluster> <jmxserver encodedStub="" host="" jmxpass="" jmxport="" jmxServiceURL="" jmxuser="" jvmDescription="" jmxserver> </jmxpoller>
Source	<xs:element name="jmxpoller"> <xs:annotation> <xs:documentation>Root element of the configuration file. This configuration file is where you specify local and remote JMX servers to connect to across your enterprise and extract whatever MBean attributes you have declared to query. The result will then be written to STDOUT for SPLUNK indexing.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element minOccurs="0" maxOccurs="1" ref="formatter"/> <xs:element minOccurs="0" maxOccurs="1" ref="transport"/> <xs:element minOccurs="0" maxOccurs="unbounded" ref="cluster"/> <xs:element minOccurs="0" maxOccurs="unbounded" ref="jmxserver"/> </xs:sequence> </xs:complexType> </xs:element>

Element formatter

Namespace	No namespace
Annotations	Custom formatter declaration
Diagram	
Properties	content: complex
Used by	Element jmxpoller
Model	param*
Children	param
Instance	<formatter className=""> <param name="" value="">{0,unbounded}</param> </formatter>
Attributes	**QName** | **Type** | **Fixed** | **Default** | **Use** **className** | xs:string | | | required Fully qualified Java class name of the formatter implementation, implements the com.dtdsoftware.splunk.formatter.Formatter interface
Source	<xs:element name="formatter"> <xs:annotation> <xs:documentation>Custom formatter declaration</xs:documentation>

```
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="param"/>
    </xs:sequence>
    <xs:attribute name="className" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation>Fully qualified Java class name of the formatter implementation,
 implements the com.dtdsoftware.splunk.formatter.Formatter interface</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element `param`

Namespace	No namespace
Annotations	parameters for a ParameterizedConfig object
Diagram	
Properties	content: complex
Used by	Elements: formatter, transport

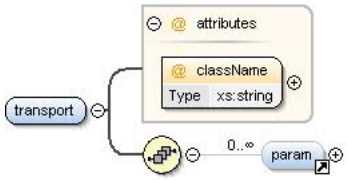
Attributes

QName	Type	Fixed	Default	Use
name	xs:string			required
	parameter name			
value	xs:string			required
	parameter value			

Source

```
<xs:element name="param">
  <xs:annotation>
    <xs:documentation xml:lang="en">parameters for a ParameterizedConfig object</
xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:attribute name="name" use="required" type="xs:string">
      <xs:annotation>
        <xs:documentation xml:lang="en">parameter name</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="value" use="required" type="xs:string">
      <xs:annotation>
        <xs:documentation xml:lang="en">parameter value</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element `transport`

Namespace	No namespace
Annotations	Custom transport declaration
Diagram	
Properties	content: complex
Used by	Element: jmxpoller

Model	param*
Children	param
Instance	`<transport className=""><param name="" value="">{0,unbounded}</param></transport>`

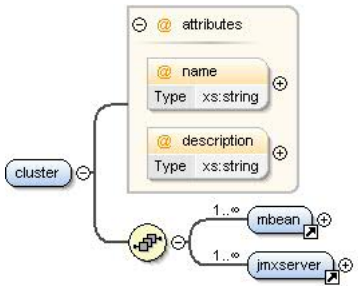
Attributes

QName	Type	Fixed	Default	Use
className	xs:string			required
	Fully qualified Java class name of the transport implementation, implements the com.dtdsoftware.splunk.transport.Transport interface			

Source

```
<xs:element name="transport">
  <xs:annotation>
    <xs:documentation>Custom transport declaration</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="param"/>
    </xs:sequence>
    <xs:attribute name="className" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation>Fully qualified Java class name of the transport implementation,
 implements the com.dtdsoftware.splunk.transport.Transport interface</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element `cluster`

Namespace	No namespace
Annotations	For JVMs with the same MBeans, you can group them under this element so you only have to declare the common beans to query once. You can still declare additional mbeans specfic to each jmxserver within the jmxserver elements.
Diagram	

Properties	content:	complex
Used by	Element	jmxpoller
Model	mbean+ , jmxserver+	
Children	jmxserver, mbean	
Instance	`<cluster description="" name=""><mbean domain="" dumpAllAttributes="" properties="">{1,unbounded}</mbean><jmxserver encodedStub="" host="" jmxpass="" jmxport="" jmxServiceURL="" jmxuser="" jvmDescription="" jmxserver></cluster>`	

Attributes

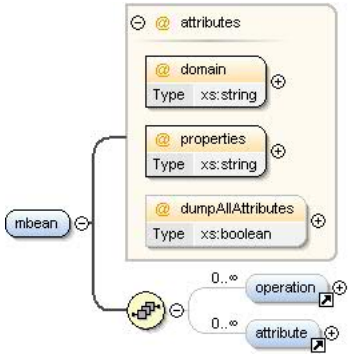
QName	Type	Fixed	Default	Use
description	xs:string			optional
	Description of this cluster			
name	xs:string			optional
	Name for this cluster			

Source

```
<xs:element name="cluster">
  <xs:annotation>
    <xs:documentation>For JVMs with the same MBeans, you can group them under this
 element so you only have to declare the common beans to query once. You can still
```

```
 declare additional mbeans specfic to each jmxserver within the jmxserver elements.</
xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element maxOccurs="unbounded" ref="mbean"/>
      <xs:element maxOccurs="unbounded" ref="jmxserver"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string">
      <xs:annotation>
        <xs:documentation>Name for this cluster</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="description" type="xs:string">
      <xs:annotation>
        <xs:documentation>Description of this cluster</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element mbean

Namespace	No namespace
Annotations	An MBean to query Standard JMX object name wildcard patterns * and ? are supported If no values are specified for the "domain" and "properties" attributes , the value will default to the * wildcard
Diagram	
Properties	content: complex
Used by	Elements: cluster, jmxserver
Model	operation* , attribute*
Children	attribute, operation

Instance	`<mbean domain="" dumpAllAttributes="" properties="">`
	`  <operation name="" outputname="">{0,unbounded}</operation>`
	`  <attribute name="" outputname="">{0,unbounded}</attribute>`
	`</mbean>`

Attributes

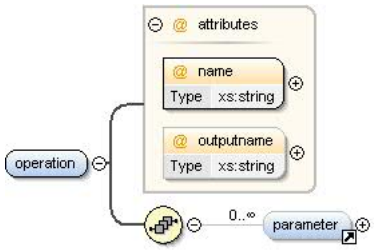
QName	Type	Fixed	Default	Use
domain	xs:string			required
	The MBean domain			
dumpAllAttributes	xs:boolean			optional
	If set to true will dump all of the attributes of the mbean. Use as an alternative to explicitly declaring each individual attribute to extract.			
properties	xs:string			required
	The MBean properties string in "key=value,key2=value2" format			

Source

```
<xs:element name="mbean">
  <xs:annotation>
    <xs:documentation>An MBean to query Standard JMX object name wildcard patterns * and ?
 are supported If no values are specified for the "domain" and "properties" attributes ,
 the value will default to the * wildcard</xs:documentation>
  </xs:annotation>
```

```
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="operation"/>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="attribute"/>
    </xs:sequence>
    <xs:attribute name="domain" use="required" type="xs:string">
      <xs:annotation>
        <xs:documentation>The MBean domain</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="properties" use="required" type="xs:string">
      <xs:annotation>
        <xs:documentation>The MBean properties string in "key=value,key2=value2" format</
xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="dumpAllAttributes" type="xs:boolean">
      <xs:annotation>
        <xs:documentation>If set to true will dump all of the attributes of the mbean.
 Use as an alternative to explicitly declaring each individual attribute to extract.</
xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element `operation`

Namespace	No namespace
Annotations	`An MBean operation`
Diagram	
Properties	content: `complex`
Used by	Element mbean
Model	parameter*
Children	parameter

`Instance`	`<operation name="" outputname="">`
	`  <parameter type="" value="">{0,unbounded}</parameter>`
	`</operation>`

Attributes

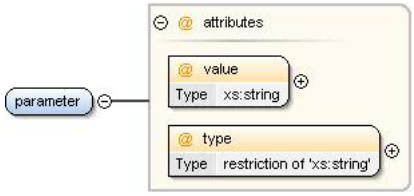
QName	Type	Fixed	Default	Use
name	xs:string			required
	The operation name. For overloaded operations, the operation signature is inferred from the paramaters list.			
outputname	xs:string			optional
	The operation result key that is output to STDOUT for SPLUNK indexing.Optional, some operations may not return values.			

`Source`

```
<xs:element name="operation">
  <xs:annotation>
    <xs:documentation>An MBean operation</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="parameter"/>
    </xs:sequence>
    <xs:attribute name="name" use="required" type="xs:string">
      <xs:annotation>
        <xs:documentation>The operation name. For overloaded operations, the operation
 signature is inferred from the paramaters list.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="outputname" type="xs:string">
      <xs:annotation>
```

```
        <xs:documentation>The operation result key that is output to STDOUT for SPLUNK
 indexing.Optional, some operations may not return values.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element `parameter`

Namespace	No namespace
Annotations	`An MBean operation parameter`
Diagram	
Properties	content: `complex`
Used by	Element operation

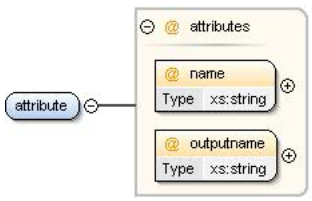
Attributes

QName	Type	Fixed	Default	Use
type	restriction of xs:string			required
	`The parameter type`			
value	xs:string			required
	`The parameter value`			

Source

```
<xs:element name="parameter">
  <xs:annotation>
    <xs:documentation>An MBean operation parameter</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:attribute name="value" use="required" type="xs:string">
      <xs:annotation>
        <xs:documentation>The parameter value</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="type" use="required">
      <xs:annotation>
        <xs:documentation>The parameter type</xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="int"/>
          <xs:enumeration value="byte"/>
          <xs:enumeration value="short"/>
          <xs:enumeration value="long"/>
          <xs:enumeration value="float"/>
          <xs:enumeration value="double"/>
          <xs:enumeration value="boolean"/>
          <xs:enumeration value="char"/>
          <xs:enumeration value="string"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element `attribute`

Namespace	No namespace
Annotations	`An MBean attribute`
Diagram	

Properties	content:	complex
Used by	Element	mbean

Attributes

QName	Type	Fixed	Default	Use
name	xs:string			required
	The attribute name For attributes that are multi level ie: composite and tabular attributes , then you can use a ":" delimited notation for specifying the attribute name. ie: foo:goo:myattribute			
outputname	xs:string			required
	The attribute key that is output to STDOUT for SPLUNK indexing			

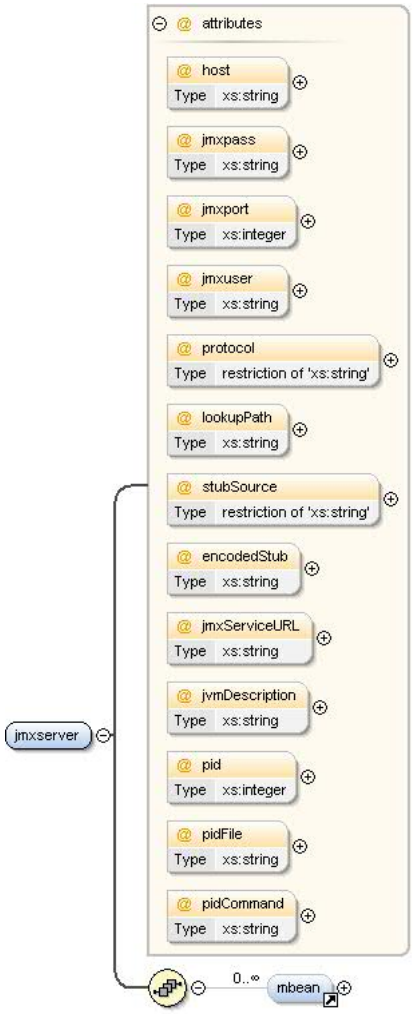
Source

```
<xs:element name="attribute">
  <xs:annotation>
    <xs:documentation>An MBean attribute</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:attribute name="name" use="required" type="xs:string">
      <xs:annotation>
        <xs:documentation>The attribute name For attributes that are multi level ie:
 composite and tabular attributes , then you can use a ":" delimited notation for
 specifying the attribute name. ie: foo:goo:myattribute</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="outputname" use="required" type="xs:string">
      <xs:annotation>
        <xs:documentation>The attribute key that is output to STDOUT for SPLUNK indexing</
xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element jmxserver

Namespace	No namespace
Annotations	A local or remote JMX Server to connect to

Diagram



Properties	content:	complex
Used by	Elements	cluster, jmxpoller
Model	mbean*	
Children	mbean	

Instance

```
<jmxserver encodedStub="" host="" jmxpass="" jmxport="" jmxServiceURL="" jmxuser="" jvmDescription="" lo
  <mbean domain="" dumpAllAttributes="" properties="">{0,unbounded}</mbean>
</jmxserver>
```

Attributes

QName	Type	Fixed	Default	Use
encodedStub	xs:string			optional
	Base64 encoded stub value for stubSource types of "ior" and "stub"			
host	xs:string			optional
	IP Address, Hostname or DNS Alias.			
jmxServiceURL	xs:string			optional
	A raw jmx service url in format "service:jmx:protocol:sap" ie: service:jmx:rmi:///jndi/rmi://myhost:9909/jmxrmi If set will take precedence over other parameters			
jmxpass	xs:string			optional
	JMX Password			
jmxport	xs:integer			optional
	JMX Port			
jmxuser	xs:string			optional
	JMX Username			
jvmDescription	xs:string			optional

QName	Type	Fixed	Default	Use
	A description of this JVM			
lookupPath	xs:string			optional
	The url path of the service.			
pid	xs:integer			optional
	Process ID for attaching directly to a locally running JVM			
pidCommand	xs:string			optional
	Command/Script that outputs to STDOUT the Process ID for attaching directly to a locally running JVM			
pidFile	xs:string			optional
	File containing the Process ID for attaching directly to a locally running JVM.The only file contents should be the PID on the first line of the file.			
protocol	restriction of xs:string			optional
	The service protocol to use.Will default to "rmi". Note : the "local", "soap", "hessian" and "burlap" protocols require MX4J to be used as the JMX implementation at both the client and server endss of the connection.			
stubSource	restriction of xs:string			optional
	The source of the remote stub.Will default to "jmxi"			

Source

```
<xs:element name="jmxserver">
  <xs:annotation>
    <xs:documentation>A local or remote JMX Server to connect to</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="mbean"/>
    </xs:sequence>
    <xs:attribute name="host" type="xs:string">
      <xs:annotation>
        <xs:documentation>IP Address, Hostname or DNS Alias.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="jmxpass" type="xs:string">
      <xs:annotation>
        <xs:documentation>JMX Password</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="jmxport" type="xs:integer">
      <xs:annotation>
        <xs:documentation>JMX Port</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="jmxuser" type="xs:string">
      <xs:annotation>
        <xs:documentation>JMX Username</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="protocol">
      <xs:annotation>
        <xs:documentation>The service protocol to use.Will default to "rmi". Note : the
"local", "soap", "hessian" and "burlap" protocols require MX4J to be used as the JMX
implementation at both the client and server endss of the connection.</xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="rmi"/>
          <xs:enumeration value="iiop"/>
          <xs:enumeration value="local"/>
          <xs:enumeration value="soap"/>
          <xs:enumeration value="hessian"/>
          <xs:enumeration value="burlap"/>
          <xs:enumeration value="soap+ssl"/>
          <xs:enumeration value="hessian+ssl"/>
          <xs:enumeration value="burlap+ssl"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="lookupPath" type="xs:string">
      <xs:annotation>
        <xs:documentation>The url path of the service.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
```

```
    <xs:attribute name="stubSource">
      <xs:annotation>
        <xs:documentation>The source of the remote stub.Will default to "jndi"</
xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="jndi"/>
          <xs:enumeration value="stub"/>
          <xs:enumeration value="ior"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="encodedStub" type="xs:string">
      <xs:annotation>
        <xs:documentation>Base64 encoded stub value for stubSource types of "ior" and
 "stub"</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="jmxServiceURL" type="xs:string">
      <xs:annotation>
        <xs:documentation>A raw jmx service url in format "service:jmx:protocol:sap" ie:
 service:jmx:rmi:///jndi/rmi://myhost:9909/jmxrmi If set will take precedence over other
 parameters</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="jvmDescription" type="xs:string">
      <xs:annotation>
        <xs:documentation>A description of this JVM</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="pid" type="xs:integer">
      <xs:annotation>
        <xs:documentation>Process ID for attaching directly to a locally running JVM</
xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="pidFile" type="xs:string">
      <xs:annotation>
        <xs:documentation>File containing the Process ID for attaching directly to a
 locally running JVM.The only file contents should be the PID on the first line of the
 file.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="pidCommand" type="xs:string">
      <xs:annotation>
        <xs:documentation>Command/Script that outputs to STDOUT the Process ID for
 attaching directly to a locally running JVM</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Attribute(s)

Attribute `param / @name`

Namespace	No namespace
Annotations	`parameter name`
Type	xs:string
Properties	use: `required`
Used by	Element param

`Source`	`<xs:attribute name="name" use="required" type="xs:string">`
	`  <xs:annotation>`
	`  <xs:documentation xml:lang="en">parameter name</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `param / @value`

Namespace	No namespace
Annotations	`parameter value`
Type	xs:string
Properties	use: `required`

Used by	Element param
Source	<xs:attribute name="value" use="required" type="xs:string"> <xs:annotation> <xs:documentation xml:lang="en">parameter value</xs:documentation> </xs:annotation> </xs:attribute>

Attribute formatter / @className

Namespace	No namespace
Annotations	Fully qualified Java class name of the formatter implementation, implements the com.dtdsoftware.splunk.formatter.Formatter interface
Type	xs:string
Properties	use: required
Used by	Element formatter
Source	<xs:attribute name="className" type="xs:string" use="required"> <xs:annotation> <xs:documentation>Fully qualified Java class name of the formatter implementation, implements the com.dtdsoftware.splunk.formatter.Formatter interface</xs:documentation> </xs:annotation> </xs:attribute>

Attribute transport / @className

Namespace	No namespace
Annotations	Fully qualified Java class name of the transport implementation, implements the com.dtdsoftware.splunk.transport.Transport interface
Type	xs:string
Properties	use: required
Used by	Element transport
Source	<xs:attribute name="className" type="xs:string" use="required"> <xs:annotation> <xs:documentation>Fully qualified Java class name of the transport implementation, implements the com.dtdsoftware.splunk.transport.Transport interface</xs:documentation> </xs:annotation> </xs:attribute>

Attribute parameter / @value

Namespace	No namespace
Annotations	The parameter value
Type	xs:string
Properties	use: required
Used by	Element parameter
Source	<xs:attribute name="value" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The parameter value</xs:documentation> </xs:annotation> </xs:attribute>

Attribute parameter / @type

Namespace	No namespace
Annotations	The parameter type
Type	restriction of xs:string
Properties	use: required
Facets	enumeration int enumeration byte

	enumeration	short
	enumeration	long
	enumeration	float
	enumeration	double
	enumeration	boolean
	enumeration	char
	enumeration	string
Used by	Element	parameter

Source

```
<xs:attribute name="type" use="required">
  <xs:annotation>
    <xs:documentation>The parameter type</xs:documentation>
  </xs:annotation>
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="int"/>
      <xs:enumeration value="byte"/>
      <xs:enumeration value="short"/>
      <xs:enumeration value="long"/>
      <xs:enumeration value="float"/>
      <xs:enumeration value="double"/>
      <xs:enumeration value="boolean"/>
      <xs:enumeration value="char"/>
      <xs:enumeration value="string"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
```

Attribute `operation / @name`

Namespace	No namespace	
Annotations	The operation name. For overloaded operations, the operation signature is inferred from the paramaters list.	
Type	xs:string	
Properties	use:	required
Used by	Element	operation

Source

```
<xs:attribute name="name" use="required" type="xs:string">
  <xs:annotation>
    <xs:documentation>The operation name. For overloaded operations, the operation
 signature is inferred from the paramaters list.</xs:documentation>
  </xs:annotation>
</xs:attribute>
```

Attribute `operation / @outputname`

Namespace	No namespace	
Annotations	The operation result key that is output to STDOUT for SPLUNK indexing.Optional, some operations may not return values.	
Type	xs:string	
Properties	content:	simple
Used by	Element	operation

Source

```
<xs:attribute name="outputname" type="xs:string">
  <xs:annotation>
    <xs:documentation>The operation result key that is output to STDOUT for SPLUNK
 indexing.Optional, some operations may not return values.</xs:documentation>
  </xs:annotation>
</xs:attribute>
```

Attribute `attribute / @name`

Namespace	No namespace
Annotations	The attribute name For attributes that are multi level ie: composite and tabular attributes , then you can use a ":"

	delimited notation for specifying the attribute name. ie: foo:goo:myattribute
Type	xs:string
Properties	use: required
Used by	Element attribute
Source	<xs:attribute name="name" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The attribute name For attributes that are multi level ie: composite and tabular attributes , then you can use a ":" delimited notation for specifying the attribute name. ie: foo:goo:myattribute</xs:documentation> </xs:annotation> </xs:attribute>

Attribute attribute / @outputname

Namespace	No namespace
Annotations	The attribute key that is output to STDOUT for SPLUNK indexing
Type	xs:string
Properties	use: required
Used by	Element attribute
Source	<xs:attribute name="outputname" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The attribute key that is output to STDOUT for SPLUNK indexing</xs:documentation> </xs:annotation> </xs:attribute>

Attribute mbean / @domain

Namespace	No namespace
Annotations	The MBean domain
Type	xs:string
Properties	use: required
Used by	Element mbean
Source	<xs:attribute name="domain" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The MBean domain</xs:documentation> </xs:annotation> </xs:attribute>

Attribute mbean / @properties

Namespace	No namespace
Annotations	The MBean properties string in "key=value,key2=value2" format
Type	xs:string
Properties	use: required
Used by	Element mbean
Source	<xs:attribute name="properties" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The MBean properties string in "key=value,key2=value2" format</xs:documentation> </xs:annotation> </xs:attribute>

Attribute mbean / @dumpAllAttributes

Namespace	No namespace
Annotations	If set to true will dump all of the attributes of the mbean.

	Use as an alternative to explicitly declaring each individual attribute to extract.
Type	xs:boolean
Properties	content: simple
Used by	Element mbean
Source	<xs:attribute name="dumpAllAttributes" type="xs:boolean"> <xs:annotation> <xs:documentation>If set to true will dump all of the attributes of the mbean. Use as an alternative to explicitly declaring each individual attribute to extract.</ xs:documentation> </xs:annotation> </xs:attribute>

Attribute jmxserver / @host

Namespace	No namespace
Annotations	IP Address, Hostname or DNS Alias.
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<xs:attribute name="host" type="xs:string"> <xs:annotation> <xs:documentation>IP Address, Hostname or DNS Alias.</xs:documentation> </xs:annotation> </xs:attribute>

Attribute jmxserver / @jmxpass

Namespace	No namespace
Annotations	JMX Password
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<xs:attribute name="jmxpass" type="xs:string"> <xs:annotation> <xs:documentation>JMX Password</xs:documentation> </xs:annotation> </xs:attribute>

Attribute jmxserver / @jmxport

Namespace	No namespace
Annotations	JMX Port
Type	xs:integer
Properties	content: simple
Used by	Element jmxserver
Source	<xs:attribute name="jmxport" type="xs:integer"> <xs:annotation> <xs:documentation>JMX Port</xs:documentation> </xs:annotation> </xs:attribute>

Attribute jmxserver / @jmxuser

Namespace	No namespace
Annotations	JMX Username
Type	xs:string
Properties	content: simple
Used by	Element jmxserver

Source	<xs:attribute name="jmxuser" type="xs:string"> <xs:annotation> <xs:documentation>JMX Username</xs:documentation> </xs:annotation> </xs:attribute>

Attribute jmxserver / @protocol

Namespace	No namespace
Annotations	The service protocol to use.Will default to "rmi". Note : the "local", "soap", "hessian" and "burlap" protocols require MX4J to be used as the JMX implementation at both the client and server endss of the connection.
Type	restriction of xs:string
Properties	content: simple
Facets	enumeration: rmi enumeration: iiop enumeration: local enumeration: soap enumeration: hessian enumeration: burlap enumeration: soap+ssl enumeration: hessian+ssl enumeration: burlap+ssl
Used by	Element: jmxserver
Source	<xs:attribute name="protocol"> <xs:annotation> <xs:documentation>The service protocol to use.Will default to "rmi". Note : the "local", "soap", "hessian" and "burlap" protocols require MX4J to be used as the JMX implementation at both the client and server endss of the connection.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:string"> <xs:enumeration value="rmi"/> <xs:enumeration value="iiop"/> <xs:enumeration value="local"/> <xs:enumeration value="soap"/> <xs:enumeration value="hessian"/> <xs:enumeration value="burlap"/> <xs:enumeration value="soap+ssl"/> <xs:enumeration value="hessian+ssl"/> <xs:enumeration value="burlap+ssl"/> </xs:restriction> </xs:simpleType> </xs:attribute>

Attribute jmxserver / @lookupPath

Namespace	No namespace
Annotations	The url path of the service.
Type	xs:string
Properties	content: simple
Used by	Element: jmxserver
Source	<xs:attribute name="lookupPath" type="xs:string"> <xs:annotation> <xs:documentation>The url path of the service.</xs:documentation> </xs:annotation> </xs:attribute>

Attribute jmxserver / @stubSource

Namespace	No namespace
Annotations	The source of the remote stub.Will default to "jndi"

Type	restriction of xs:string
Properties	content: simple
Facets	enumeration jndi enumeration stub enumeration ior
Used by	Element jmxserver
Source	<xs:attribute name="stubSource"> <xs:annotation> <xs:documentation>The source of the remote stub.Will default to "jndi"</ xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:string"> <xs:enumeration value="jndi"/> <xs:enumeration value="stub"/> <xs:enumeration value="ior"/> </xs:restriction> </xs:simpleType> </xs:attribute>

Attribute jmxserver / @encodedStub

Namespace	No namespace
Annotations	Base64 encoded stub value for stubSource types of "ior" and "stub"
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<xs:attribute name="encodedStub" type="xs:string"> <xs:annotation> <xs:documentation>Base64 encoded stub value for stubSource types of "ior" and "stub"</ xs:documentation> </xs:annotation> </xs:attribute>

Attribute jmxserver / @jmxServiceURL

Namespace	No namespace
Annotations	A raw jmx service url in format "service:jmx:protocol:sap" ie: service:jmx:rmi:///jndi/rmi://myhost:9909/jmxrmi If set will take precedence over other parameters
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<xs:attribute name="jmxServiceURL" type="xs:string"> <xs:annotation> <xs:documentation>A raw jmx service url in format "service:jmx:protocol:sap" ie: service:jmx:rmi:///jndi/rmi://myhost:9909/jmxrmi If set will take precedence over other parameters</xs:documentation> </xs:annotation> </xs:attribute>

Attribute jmxserver / @jvmDescription

Namespace	No namespace
Annotations	A description of this JVM
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<xs:attribute name="jvmDescription" type="xs:string"> <xs:annotation> <xs:documentation>A description of this JVM</xs:documentation>

```
  </xs:annotation>
</xs:attribute>
```

Attribute `jmxserver / @pid`

Namespace	No namespace
Annotations	Process ID for attaching directly to a locally running JVM
Type	xs:integer
Properties	content: simple
Used by	Element jmxserver
Source	

```
<xs:attribute name="pid" type="xs:integer">
  <xs:annotation>
    <xs:documentation>Process ID for attaching directly to a locally running JVM</
xs:documentation>
  </xs:annotation>
</xs:attribute>
```

Attribute `jmxserver / @pidFile`

Namespace	No namespace
Annotations	File containing the Process ID for attaching directly to a locally running JVM.The only file contents should be the PID on the first line of the file.
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	

```
<xs:attribute name="pidFile" type="xs:string">
  <xs:annotation>
    <xs:documentation>File containing the Process ID for attaching directly to a locally
 running JVM.The only file contents should be the PID on the first line of the file.</
xs:documentation>
  </xs:annotation>
</xs:attribute>
```

Attribute `jmxserver / @pidCommand`

Namespace	No namespace
Annotations	Command/Script that outputs to STDOUT the Process ID for attaching directly to a locally running JVM
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	

```
<xs:attribute name="pidCommand" type="xs:string">
  <xs:annotation>
    <xs:documentation>Command/Script that outputs to STDOUT the Process ID for attaching
 directly to a locally running JVM</xs:documentation>
  </xs:annotation>
</xs:attribute>
```

Attribute `cluster / @name`

Namespace	No namespace
Annotations	Name for this cluster
Type	xs:string
Properties	content: simple
Used by	Element cluster
Source	

```
<xs:attribute name="name" type="xs:string">
  <xs:annotation>
    <xs:documentation>Name for this cluster</xs:documentation>
  </xs:annotation>
</xs:attribute>
```

Attribute `cluster / @description`

Namespace	No namespace
Annotations	`Description of this cluster`
Type	xs:string
Properties	content: `simple`
Used by	Element cluster

`Source`	`<xs:attribute name="description" type="xs:string">`
	`  <xs:annotation>`
	`  <xs:documentation>Description of this cluster</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`