System.Text.Json Library Reference

September 24, 2014

Contents

Description

Provides support for JavaScript Object Notation (http://json.org/).

Copyrights

Namespaces

Namespace	Description
System.Text.Json	Contains a function for obtaining structured representation for data in JSON format.
System.Text.Json.Data	Contains classes for representing data in JSON format.
System.Text.Json.Schema	Contains classes that represent the structure of data in JSON format. Classes derived from these schema classes can be populated with JSON data.

1 Usage

1.0.1 Referencing the Json Library

Right-click a project node in IDE | Project References... | Add System Extension Library Reference... | enable *System.Text.Json* check box

or add following line to your project's .cmp file:

```
reference <ext/System.Text.Json/System.Text.Json.cml>;
```

2 Schema Generation

2.0.1 Introduction

If some JSON data have a fixed structure, that is: it can be represented by classes, arrays and sequences, this library supports the generation of schema classes that can be populated with a specific JSON data.

The schema classes can be generated with the **jsonschemagen** utility that comes with the Cmajor compiler. jsonschemagen is a source-to-source tool. It reads a file containing schema classes and generates a Cmajor source code file that contains corresponding Cmajor classes.

For example, **downloads** example program has a schema file **statistics.schema** (2.0.3) that contains definition of schema classes that can be populated with statistics data in JSON format retrieved from SourceForge Cmajor project site.

2.0.2 Schema File Syntax

2.0.2.1 Top-level Structure

The top-level of a schema file consists of sequences and structures.

⟨*schema-file*⟩ ::= ⟨*sequence*⟩ | ⟨*struct*⟩

⟨*sequence*⟩ ::= `sequence` ⟨*identifier*⟩ `{` ⟨*field*⟩ (`,` ⟨*field*⟩)* `}`

⟨*struct*⟩ ::= `struct` ⟨*identifier*⟩ `{` ⟨*field*⟩ (`,` ⟨*field*⟩)* `}`

The difference between sequence and structure is that sequence can be populated with a JSON array with fixed number of elements of possibly different types whereas structure has named fields.

For example if we have JSON data `["foo", 123]` and `["bar", 234]` , they can be represented as sequences of the form

```
sequence Foo
{
    string name;
    number value;
}
```

On the other hand, if we have JSON object `{"foo" : 123, "bar" : 234}`, it can be represented as a structure of the form

```
struct Foo
{
    number foo;
    number bar;
}
```

2.0.2.2 Fields

Structures and sequences contain fields.

⟨*field*⟩ ::= ⟨*string-field*⟩ | ⟨*number-field*⟩ | ⟨*boolean-field*⟩ | ⟨*container-field*⟩ | ⟨*array-field*⟩

2.0.2.2.1 String Field

A string field can be populated with a JSON string; for example string `"foo"`.

⟨*string-field*⟩ ::= `string` ⟨*field-name*⟩ `;`

⟨*field-name*⟩ ::= ⟨*identifier*⟩

⟨*identifier*⟩ ::= [a-zA-Z_][0-9a-zA-Z_]* - ⟨*keyword*⟩

⟨*keyword*⟩ ::= `string` | `number` | `bool` | `sequence` | `struct`

2.0.2.2.2 Number Field

A number field can be populated with a JSON number; for example 10 or -5.7.

⟨*number-field*⟩ ::= `number` ⟨*field-name*⟩ `;`

2.0.2.2.3 Boolean Field

A Boolean field can be populated with a JSON Boolean value: **true** or **false**.

⟨*boolean-field*⟩ ::= `bool` ⟨*field-name*⟩ `;`

2.0.2.2.4 Container Field

Container field is a field of whose type is a sequence or a structure defined earlier in the same schema file.

⟨*container-field*⟩ ::= ⟨*container-name*⟩ ⟨*field-name*⟩ `;`

⟨*container-name*⟩ ::= ⟨*identifier*⟩

2.0.2.2.5 Array Field

An array field can be populated with a JSON array of elements of same type. For example, JSON string array: [”foo”, ”bar”, ”baz”], or JSON number array: [1, 2, 3].

⟨*array-field*⟩ ::= ⟨*field-type*⟩ `[ ]` ⟨*field-name*⟩ `;`

⟨*field-type*⟩ ::= `string` | `number` | `bool` | ⟨*container-type*⟩

⟨*container-type*⟩ ::= ⟨*identifier*⟩

2.0.3 Example

This is a schema of statistics data that can be retrieved from the SourceForge Cmajor project site in JSON format.

```
sequence OsStat
{
    string os;
    number count;
}

sequence CountryStat
{
    string country;
    number count;
}

sequence DateStat
{
    string date;
    number count;
}

struct OsSummary
{
    string top;
    number percent;
    string modifier_text;
}

struct GeoSummary
{
    string top;
    number percent;
    string modifier_text;
}
```

```
struct TimeSummary
{
    number downloads;
}

struct Summaries
{
    OsSummary os;
    GeoSummary geo;
    TimeSummary time;
}

struct Statistics
{
    OsStat[] oses;
    string start_date;
    string end_date;
    CountryStat[] countries;
    string[] oses_with_downloads;
    DateStat[] downloads;
    string[] messages;
    string period;
    number total;
    string stats_updated;
    Summaries summaries;
}
```

3 System.Text.Json Namespace

Contains a function for obtaining structured representation for data in JSON format.

3.1 Functions

Function	Description
ParseJson(const String&)	Parses a string that contains data in JSON format and returns an object representing it.

3.1.1 ParseJson(const String&) Function

Parses a string that contains data in JSON format and returns an object representing it.

Syntax

```
public UniquePtr<JsonValue> ParseJson(const String& jsonText);
```

Parameters

Name	Type	Description
jsonText	const String&	Data in JSON format.

Returns

UniquePtr<JsonValue>

Returns an object that represents the data.

4 System.Text.Json.Data Namespace

Contains classes for representing data in JSON format.

4.2 Classes

Class	Description
JsonArray	Represents an array of JSON values.
JsonBool	Represents a JSON Boolean value.
JsonNull	Represents a JSON null.
JsonNumber	Represents a JSON number.
JsonObject	Represents a JSON object.
JsonString	Represents a JSON string.
JsonValue	An abstract base class for JSON values.

4.2.1 JsonArray Class

Represents an array of JSON values.

Syntax

```
public class JsonArray;
```

Base Class

JsonValue

4.2.1.1 Member Functions

Member Function	Description
JsonArray()	Default constructor. Initializes an empty array of JSON values.
~JsonArray()	Destructor.
operator[](int) const	Returns a JSON value of given index from the array.
AddItem(JsonValue*)	Adds a JSON value to the array.
Count() const	Returns the number of items in the array.
GetItem(int) const	Returns a JSON value of given index from the array.
IsArray() const	Returns true.
ToString() const	Returns the string representation of the array in JSON format.

4.2.1.1.1 JsonArray() Member Function

Default constructor. Initializes an empty array of JSON values.

Syntax

```
public JsonArray();
```

4.2.1.1.2 ~JsonArray() Member Function

Destructor.

Syntax

```
public override nothrow ~JsonArray();
```

4.2.1.1.3 operator[](int) const Member Function

Returns a JSON value of given index from the array.

Syntax

```
public nothrow JsonValue* operator[](int index) const;
```

Parameters

Name	Type	Description
index	int	Index of the JSON value to retrieve.

Returns

JsonValue*

Returns a JSON value of given index from the array.

4.2.1.1.4 AddItem(JsonValue*) Member Function

Adds a JSON value to the array.

Syntax

```
public void AddItem(JsonValue* item);
```

Parameters

Name	Type	Description
item	JsonValue*	A JSON value.

4.2.1.1.5 Count() const Member Function

Returns the number of items in the array.

Syntax

```
public inline nothrow int Count() const;
```

Returns

int

Returns the number of items in the array.

4.2.1.1.6 GetItem(int) const Member Function

Returns a JSON value of given index from the array.

Syntax

```
public nothrow JsonValue* GetItem(int index) const;
```

Parameters

Name	Type	Description
index	int	Index of the JSON value to retrieve.

Returns

JsonValue*

Returns a JSON value of given index from the array.

4.2.1.1.7 IsArray() const Member Function

Returns true.

Syntax

```
public override bool IsArray() const;
```

Returns

bool

Return true;

4.2.1.1.8 ToString() const Member Function

Returns the string representation of the array in JSON format.

Syntax

```
public override String ToString() const;
```

Returns

String

Returns the string representation of the array in JSON format.

4.2.2 JsonBool Class

Represents a JSON Boolean value.

Syntax

```
public class JsonBool;
```

Base Class

JsonValue

4.2.2.1 Member Functions

Member Function	Description
JsonBool()	Default constructor. Initializes the value to false.
JsonBool(const JsonBool&)	Copy constructor.
JsonBool(JsonBool&&)	Move constructor.
JsonBool(bool)	Constructor. Initializes the JSON Boolean value with the specified value.
operator=(const JsonBool&)	Copy assignment.
operator=(JsonBool&&)	Move assignment.
~JsonBool()	Destructor.
IsBool() const	Returns true.
SetValue(bool)	Sets the contained Boolean value.
ToString() const	Returns the string representation of contained value in JSON format.
Value() const	Returns the contained value.

4.2.2.1.1 JsonBool() Member Function

Default constructor. Initializes the value to false.

Syntax

```
public JsonBool();
```

4.2.2.1.2 JsonBool(const JsonBool&) Member Function

Copy constructor.

Syntax

```
public nothrow JsonBool(const JsonBool& that);
```

Parameters

Name	Type	Description
that	const JsonBool&	A JSON Boolean value to copy.

4.2.2.1.3 JsonBool(JsonBool&&) Member Function

Move constructor.

Syntax

```
public nothrow JsonBool(JsonBool&& that);
```

Parameters

Name	Type	Description
that	JsonBool&&	A JSON Boolean value to move.

4.2.2.1.4 JsonBool(bool) Member Function

Constructor. Initializes the JSON Boolean value with the specified value.

Syntax

```
public nothrow JsonBool(bool value_);
```

Parameters

Name	Type	Description
value_	bool	A Boolean value.

4.2.2.1.5 operator=(const JsonBool&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const JsonBool& that);
```

Parameters

Name	Type	Description
that	const JsonBool&	A JSON Boolean value to copy.

4.2.2.1.6 operator=(JsonBool&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(JsonBool&& that);
```

Parameters

Name	Type	Description
that	JsonBool&&	A JSON Boolean value to move.

4.2.2.1.7 ~JsonBool() Member Function

Destructor.

Syntax

```
public override nothrow ~JsonBool();
```

4.2.2.1.8 IsBool() const Member Function

Returns true.

Syntax

```
public override bool IsBool() const;
```

Returns

bool

Returns true.

4.2.2.1.9 SetValue(bool) Member Function

Sets the contained Boolean value.

Syntax

```
public void SetValue(bool value_);
```

Parameters

Name	Type	Description
value_	bool	A Boolean value.

4.2.2.1.10 ToString() const Member Function

Returns the string representation of contained value in JSON format.

Syntax

```
public override String ToString() const;
```

Returns

String

Returns the string representation of contained value in JSON format.

4.2.2.1.11 Value() const Member Function

Returns the contained value.

Syntax

```
public inline nothrow bool Value() const;
```

Returns

bool

Returns the contained value.

4.2.3 JsonNull Class

Represents a JSON null.

Syntax

```
public class JsonNull;
```

Base Class

JsonValue

4.2.3.1 Member Functions

Member Function	Description
JsonNull()	Default constructor.
JsonNull(JsonNull&&)	Move constructor.
JsonNull(const JsonNull&)	Copy constructor.
operator=(JsonNull&&)	Move assignment.
operator=(const JsonNull&)	Copy assignment.
~JsonNull()	Destructor.
IsNull() const	Returns true.
ToString() const	Returns the string representation of the contained value in JSON format.

4.2.3.1.1 JsonNull() Member Function

Default constructor.

Syntax

```
public nothrow JsonNull();
```

4.2.3.1.2 JsonNull(JsonNull&&) Member Function

Move constructor.

Syntax

```
public nothrow JsonNull(JsonNull&& that);
```

Parameters

Name	Type	Description
that	JsonNull&&	A JSON null to move.

4.2.3.1.3 JsonNull(const JsonNull&) Member Function

Copy constructor.

Syntax

```
public nothrow JsonNull(const JsonNull& that);
```

Parameters

Name	Type	Description
that	const JsonNull&	A JSON null to copy.

4.2.3.1.4 operator=(JsonNull&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(JsonNull&& that);
```

Parameters

Name	Type	Description
that	JsonNull&&	A JSON null to move.

4.2.3.1.5 operator=(const JsonNull&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const JsonNull& that);
```

Parameters

Name	Type	Description
that	const JsonNull&	A JSON null to copy.

4.2.3.1.6 ~JsonNull() Member Function

Destructor.

Syntax

```
public override nothrow ~JsonNull();
```

4.2.3.1.7 IsNull() const Member Function

Returns true.

Syntax

```
public override bool IsNull() const;
```

Returns

bool

Returns true.

4.2.3.1.8 ToString() const Member Function

Returns the string representation of the contained value in JSON format.

Syntax

```
public override String ToString() const;
```

Returns

String

Returns the string representation of the contained value in JSON format.

4.2.4 JsonNumber Class

Represents a JSON number.

Syntax

```
public class JsonNumber;
```

Base Class

JsonValue

4.2.4.1 Member Functions

Member Function	Description
JsonNumber()	Default constructor. Initializes the JSON number to zero.
JsonNumber(JsonNumber&&)	Move constructor.
JsonNumber(const JsonNumber&)	Copy constructor.
JsonNumber(double)	Constructor. Initializes the JSON number with the specified value.
operator=(JsonNumber&&)	Move assignment.
operator=(const JsonNumber&)	Copy assignment.
~JsonNumber()	Destructor.
IsNumber() const	Returns true.
SetValue(double)	Sets the contained value.
ToString() const	Returns the string representation of the contained number in JSON format.
Value() const	Returns the contained value.

4.2.4.1.1 JsonNumber() Member Function

Default constructor. Initializes the JSON number to zero.

Syntax

```
public JsonNumber();
```

4.2.4.1.2 JsonNumber(JsonNumber&&) Member Function

Move constructor.

Syntax

```
public nothrow JsonNumber(JsonNumber&& that);
```

Parameters

Name	Type	Description
that	JsonNumber&&	A JSON number to move.

4.2.4.1.3 JsonNumber(const JsonNumber&) Member Function

Copy constructor.

Syntax

```
public nothrow JsonNumber(const JsonNumber& that);
```

Parameters

Name	Type	Description
that	const JsonNumber&	A JSON number to copy.

4.2.4.1.4 JsonNumber(double) Member Function

Constructor. Initializes the JSON number with the specified value.

Syntax

```
public nothrow JsonNumber(double value_);
```

Parameters

Name	Type	Description
value_	double	A number.

4.2.4.1.5 operator=(JsonNumber&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(JsonNumber&& that);
```

Parameters

Name	Type	Description
that	JsonNumber&&	A JSON number to move.

4.2.4.1.6 operator=(const JsonNumber&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const JsonNumber& that);
```

Parameters

Name	Type	Description
that	const JsonNumber&	A JSON number to copy.

4.2.4.1.7 ~JsonNumber() Member Function

Destructor.

Syntax

```
public override nothrow ~JsonNumber();
```

4.2.4.1.8 IsNumber() const Member Function

Returns true.

Syntax

```
public override bool IsNumber() const;
```

Returns

bool

Returns true.

4.2.4.1.9 SetValue(double) Member Function

Sets the contained value.

Syntax

```
public void SetValue(double value_);
```

Parameters

Name	Type	Description

value_	double	A number.

4.2.4.1.10 ToString() const Member Function

Returns the string representation of the contained number in JSON format.

Syntax

```
public override String ToString() const;
```

Returns

String

Returns the string representation of the contained number in JSON format.

4.2.4.1.11 Value() const Member Function

Returns the contained value.

Syntax

```
public inline nothrow double Value() const;
```

Returns

double

Returns the contained value.

4.2.5 JsonObject Class

Represents a JSON object.

Syntax

```
public class JsonObject;
```

Base Class

JsonValue

4.2.5.1 Member Functions

Member Function	Description
JsonObject()	Default constructor. Initializes an empty JSON object.
~JsonObject()	Destructor.
AddField(JsonString&&, JsonValue*)	Adds a named field with a JSON value to the JSON object.
GetField(const JsonString&) const	Returns the value of the field with the specified name, or null if the field with the given name does not exist.
IsObject() const	Returns true.
ToString() const	Returns the string representation of the contained fields in JSON format.

4.2.5.1.1 JsonObject() Member Function

Default constructor. Initializes an empty JSON object.

Syntax

```
public JsonObject();
```

4.2.5.1.2 ~JsonObject() Member Function

Destructor.

Syntax

```
public override nothrow ~JsonObject();
```

4.2.5.1.3 AddField(JsonString&&, JsonValue*) Member Function

Adds a named field with a JSON value to the JSON object.

Syntax

```
public void AddField(JsonString&& name, JsonValue* value);
```

Parameters

Name	Type	Description
name	JsonString&&	The name of the field to add.
value	JsonValue*	The value of the field to add.

4.2.5.1.4 GetField(const JsonString&) const Member Function

Returns the value of the field with the specified name, or null if the field with the given name does not exist.

Syntax

```
public JsonValue* GetField(const JsonString& name) const;
```

Parameters

Name	Type	Description
name	const JsonString&	The name of the field to retrieve.

Returns

JsonValue*

Returns the value of the field with the specified name, or null if the field with the given name does not exist.

4.2.5.1.5 IsObject() const Member Function

Returns true.

Syntax

```
public override bool IsObject() const;
```

Returns

bool

Returns true.

4.2.5.1.6 ToString() const Member Function

Returns the string representation of the contained fields in JSON format.

Syntax

```
public override String ToString() const;
```

Returns

String

Returns the string representation of the contained fields in JSON format.

4.2.6 JsonString Class

Represents a JSON string.

Syntax

```
public class JsonString;
```

Base Class

JsonValue

4.2.6.1 Member Functions

Member Function	Description
JsonString()	Default constructor. Initializes an empty JSON string.
JsonString(const String&)	Constructor. Initializes the JSON string with the specified value.
JsonString(JsonString&&)	Move constructor.
JsonString(const JsonString&)	Copy constructor.
operator=(JsonString&&)	Move assignment.
operator=(const JsonString&)	Copy assignment.
~JsonString()	Destructor.
Append(const String&)	Appends the specified string to the end of the contained string.
Append(char)	Appends the specified character to the end of the contained string.
IsString() const	Returns true.
SetValue(const String&)	Sets the contained value.
ToString() const	Returns the string representation of the contained value in JSON format.

Value() const	Returns the contained value.

4.2.6.1.1 JsonString() Member Function

Default constructor. Initializes an empty JSON string.

Syntax

```
public JsonString();
```

4.2.6.1.2 JsonString(const String&) Member Function

Constructor. Initializes the JSON string with the specified value.

Syntax

```
public JsonString(const String& value_);
```

Parameters

Name	Type	Description
value_	const String&	A value.

4.2.6.1.3 JsonString(JsonString&&) Member Function

Move constructor.

Syntax

```
public nothrow JsonString(JsonString&& that);
```

Parameters

Name	Type	Description
that	JsonString&&	A JSON string to move.

4.2.6.1.4 JsonString(const JsonString&) Member Function

Copy constructor.

Syntax

```
public nothrow JsonString(const JsonString& that);
```

Parameters

Name	Type	Description
that	const JsonString&	A JSON string to copy.

4.2.6.1.5 operator=(JsonString&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(JsonString&& that);
```

Parameters

Name	Type	Description
that	JsonString&&	A JSON string to move.

4.2.6.1.6 operator=(const JsonString&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const JsonString& that);
```

Parameters

Name	Type	Description
that	const JsonString&	A JSON string to copy.

4.2.6.1.7 ~JsonString() Member Function

Destructor.

Syntax

```
public override nothrow ~JsonString();
```

4.2.6.1.8 Append(const String&) Member Function

Appends the specified string to the end of the contained string.

Syntax

```
public void Append(const String& s);
```

Parameters

Name	Type	Description
s	const String&	A string to append.

4.2.6.1.9 Append(char) Member Function

Appends the specified character to the end of the contained string.

Syntax

```
public void Append(char c);
```

Parameters

Name	Type	Description
c	char	A character to append.

4.2.6.1.10 IsString() const Member Function

Returns true.

Syntax

```
public override bool IsString() const;
```

Returns

bool

Returns true.

4.2.6.1.11 SetValue(const String&) Member Function

Sets the contained value.

Syntax

```
public void SetValue(const String& value_);
```

Parameters

Name	Type	Description
value_	const String&	A value.

4.2.6.1.12 ToString() const Member Function

Returns the string representation of the contained value in JSON format.

Syntax

```
public override String ToString() const;
```

Returns

String

Returns the string representation of the contained value in JSON format.

4.2.6.1.13 Value() const Member Function

Returns the contained value.

Syntax

```
public nothrow const String& Value() const;
```

Returns

const String&

Returns the contained value.

4.2.6.2 Nonmember Functions

Function	Description
operator==(const JsonString&, const JsonString&)	Compares to JSON string for equality.
operator<(const JsonString&, const JsonString&)	Compares two JSON strings for less than relationship.

4.2.6.2.1 operator==(const JsonString&, const JsonString&) Function

Compares to JSON string for equality.

Syntax

```
public nothrow bool operator==(const JsonString& left, const JsonString& right);
```

Parameters

Name	Type	Description
left	const JsonString&	The first JSON string to compare.
right	const JsonString&	The second JSON string to compare.

Returns

bool

Returns true, if the first string contains equal number of equal characters with the second string, false otherwise.

4.2.6.2.2 operator<(const JsonString&, const JsonString&) Function

Compares two JSON strings for less than relationship.

Syntax

```
public nothrow bool operator<(const JsonString& left, const JsonString& right);
```

Parameters

Name	Type	Description
left	const JsonString&	The first JSON string to compare.
right	const JsonString&	The second JSON string to compare.

Returns

bool

Returns true if the first JSON string comes lexicographically before the second JSON string, false otherwise.

4.2.7 JsonValue Class

An abstract base class for JSON values.

Syntax

```
public abstract class JsonValue;
```

4.2.7.1 Member Functions

Member Function	Description
JsonValue()	Default constructor. Initializes an empty value.
~JsonValue()	Destructor.
IsArray() const	Returns true if this JSON value is a JsonArray, false otherwise.
IsBool() const	Returns true if this JSON value is a JsonBool, false otherwise.
IsNull() const	Returns true if this JSON value is a JsonNull, false otherwise.
IsNumber() const	Returns true if this JSON value is a JsonNumber, false otherwise.
IsObject() const	Returns true if this JSON value is a JsonObject, false otherwise.
IsString() const	Returns true if this JSON value is a JsonString, false otherwise.
ToString() const	Returns a string representation of the value in JSON format.

4.2.7.1.1 JsonValue() Member Function

Default constructor. Initializes an empty value.

Syntax

```
public nothrow JsonValue();
```

4.2.7.1.2 ~JsonValue() Member Function

Destructor.

Syntax

```
public virtual nothrow ~JsonValue();
```

4.2.7.1.3 IsArray() const Member Function

Returns true if this JSON value is a JsonArray, false otherwise.

Syntax

```
public virtual bool IsArray() const;
```

Returns

bool

Returns true if this JSON value is a JsonArray, false otherwise.

4.2.7.1.4 IsBool() const Member Function

Returns true if this JSON value is a JsonBool, false otherwise.

Syntax

```
public virtual bool IsBool() const;
```

Returns

bool

Returns true if this JSON value is a JsonBool, false otherwise.

4.2.7.1.5 IsNull() const Member Function

Returns true if this JSON value is a JsonNull, false otherwise.

Syntax

```
public virtual bool IsNull() const;
```

Returns

bool

Returns true if this JSON value is a JsonNull, false otherwise.

4.2.7.1.6 IsNumber() const Member Function

Returns true if this JSON value is a JsonNumber, false otherwise.

Syntax

```
public virtual bool IsNumber() const;
```

Returns

bool

Returns true if this JSON value is a JsonNumber, false otherwise.

4.2.7.1.7 IsObject() const Member Function

Returns true if this JSON value is a JsonObject, false otherwise.

Syntax

```
public virtual bool IsObject() const;
```

Returns

bool

Returns true if this JSON value is a JsonObject, false otherwise.

4.2.7.1.8 IsString() const Member Function

Returns true if this JSON value is a JsonString, false otherwise.

Syntax

```
public virtual bool IsString() const;
```

Returns

bool

Returns true if this JSON value is a JsonString, false otherwise.

4.2.7.1.9 ToString() const Member Function

Returns a string representation of the value in JSON format.

Syntax

```
public abstract String ToString() const;
```

Returns

String

Returns a string representation of the value in JSON format.

5 System.Text.Json.Schema Namespace

Contains classes that represent the structure of data in JSON format. Classes derived from these schema classes can be populated with JSON data.

5.3 Classes

Class	Description
ArrayField<ItemType>	Represents a sequence of items of same type.
ArrayFieldBase	A base class for array fields.
BooleanField	Represents a Boolean valued field.
Container	Base class of Sequence and Struct.
ContainerField	Helper class for schema class generation.
Field	Base class for all schema field types.
NumberField	Represents numeric field.
SchemaFileContent	Represents contents of a schema file.
Sequence	Represents a sequence of fields. Differs from ArrayField<ItemType> in that number of fields must be fixed and fields can be polymorphic.
StringField	Represents a string field.
Struct	Represents a structured field.

5.3.1 ArrayField<ItemType> Class

Represents a sequence of items of same type.

Syntax

```
public class ArrayField<ItemType>;
```

Base Class

ArrayFieldBase

5.3.1.1 Remarks

ItemType can be StringField, NumberField, BooleanField, a container derived from Sequence or Struct.

5.3.1.2 Member Functions

Member Function	Description
ArrayField(const String&)	Constructor. Initializes the array field with specified field name.
operator[](int) const	Gets item with the specified index.
Begin()	Returns an iterator pointing to the first array item.
Count() const	Returns the number of items in this array.
End()	Returns an iterator pointing one past the last array item.
FetchData(JsonValue*)	Populates the array field with data retrieved from the specified JSON array.

5.3.1.2.1 ArrayField(const String&) Member Function

Constructor. Initializes the array field with specified field name.

Syntax

```
public ArrayField(const String& fieldName_);
```

Parameters

Name	Type	Description
fieldName_	const String&	The name of the array field.

5.3.1.2.2 operator[](int) const Member Function

Gets item with the specified index.

Syntax

```
public nothrow const ItemType& operator[](int index) const;
```

Parameters

Name	Type	Description
index	int	Index of item to retrieve.

Returns

const ItemType&

Returns a reference to the requested item.

5.3.1.2.3 Begin() Member Function

Returns an iterator pointing to the first array item.

Syntax

```
public nothrow RandomAccessIter<T, T&, T*> Begin();
```

Returns

RandomAccessIter<T, T&, T*>

Returns an iterator pointing to the first array item.

5.3.1.2.4 Count() const Member Function

Returns the number of items in this array.

Syntax

```
public nothrow int Count() const;
```

Returns

int

Returns the number of items in this array.

5.3.1.2.5 End() Member Function

Returns an iterator pointing one past the last array item.

Syntax

```
public nothrow RandomAccessIter<T, T&, T*> End();
```

Returns

RandomAccessIter<T, T&, T*>

Returns an iterator pointing one past the last array item.

5.3.1.2.6 FetchData(JsonValue*) Member Function

Populates the array field with data retrieved from the specified JSON array.

Syntax

```
public override void FetchData(JsonValue* value);
```

Parameters

Name	Type	Description
value	JsonValue*	A JSON array.

5.3.2 ArrayFieldBase Class

A base class for array fields.

Syntax

```
public class ArrayFieldBase;
```

Base Class

Field

5.3.2.1 Member Functions

Member Function	Description
ArrayFieldBase(const ArrayFieldBase&)	Copy constructor.
ArrayFieldBase(ArrayFieldBase&&)	Move constructor.
ArrayFieldBase(const String&)	Constructor. Initializes the array field with the specified field name.
ArrayFieldBase(const String&, const String&)	Constructor. Initializes the array field with the specified item type name and field name.
operator=(const ArrayFieldBase&)	Copy assignment.
operator=(ArrayFieldBase&&)	Move assignment.
~ArrayFieldBase()	Destructor.
GenerateDefinitions(CodeFormatter&)	Generates textual field definition.

5.3.2.1.1 ArrayFieldBase(const ArrayFieldBase&) Member Function

Copy constructor.

Syntax

```
public nothrow ArrayFieldBase(const ArrayFieldBase& that);
```

Parameters

Name	Type	Description
that	const ArrayFieldBase&	An array field to copy.

5.3.2.1.2 ArrayFieldBase(ArrayFieldBase&&) Member Function

Move constructor.

Syntax

```
public nothrow ArrayFieldBase(ArrayFieldBase&& that);
```

Parameters

Name	Type	Description
that	ArrayFieldBase&&	An array field to move.

5.3.2.1.3 ArrayFieldBase(const String&) Member Function

Constructor. Initializes the array field with the specified field name.

Syntax

```
public ArrayFieldBase(const String& fieldName_);
```

Parameters

Name	Type	Description
fieldName_	const String&	The name of the array field.

5.3.2.1.4 ArrayFieldBase(const String&, const String&) Member Function

Constructor. Initializes the array field with the specified item type name and field name.

Syntax

```
public ArrayFieldBase(const String& itemTypeName_, const String& fieldName_);
```

Parameters

Name	Type	Description
itemTypeName_	const String&	The name of the item type.
fieldName_	const String&	The name of the array field.

5.3.2.1.5 operator=(const ArrayFieldBase&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const ArrayFieldBase& that);
```

Parameters

Name	Type	Description
that	const ArrayFieldBase&	An array field to copy.

5.3.2.1.6 operator=(ArrayFieldBase&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(ArrayFieldBase&& that);
```

Parameters

Name	Type	Description
that	ArrayFieldBase&&	An array field to move.

5.3.2.1.7 ~ArrayFieldBase() Member Function

Destructor.

Syntax

```
public override nothrow ~ArrayFieldBase();
```

5.3.2.1.8 GenerateDefinitions(CodeFormatter&) Member Function

Generates textual field definition.

Syntax

```
public override void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description
formatter	CodeFormatter&	A code formatter.

5.3.3 BooleanField Class

Represents a Boolean valued field.

Syntax

```
public class BooleanField;
```

Base Class

Field

5.3.3.1 Member Functions

Member Function	Description
BooleanField(const BooleanField&)	Copy constructor.
BooleanField(const String&)	Constructor. Initializes the Boolean field with the specified field name.
BooleanField(BooleanField&&)	Move constructor.
operator=(const BooleanField&)	Copy assignment.
operator=(BooleanField&&)	Move assignment.
~BooleanField()	Destructor.
@operator_conv() const	Conversion function that returns the value of the field.
FetchData(JsonValue*)	Populates the Boolean field with data retrieved from the specified JSON Boolean value.
GenerateDefinitions(CodeFormatter&)	Generates textual field definition.

5.3.3.1.1 BooleanField(const BooleanField&) Member Function

Copy constructor.

Syntax

```
public nothrow BooleanField(const BooleanField& that);
```

Parameters

Name	Type	Description
that	const BooleanField&	A Boolean field to copy.

5.3.3.1.2 BooleanField(const String&) Member Function

Constructor. Initializes the Boolean field with the specified field name.

Syntax

```
public BooleanField(const String& fieldName_);
```

Parameters

Name	Type	Description
fieldName_	const String&	The name of the Boolean field.

5.3.3.1.3 BooleanField(BooleanField&&) Member Function

Move constructor.

Syntax

```
public nothrow BooleanField(BooleanField&& that);
```

Parameters

Name	Type	Description
that	BooleanField&&	A Boolean field to move.

5.3.3.1.4 operator=(const BooleanField&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const BooleanField& that);
```

Parameters

Name	Type	Description
that	const BooleanField&	A Boolean field to copy.

5.3.3.1.5 operator=(BooleanField&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(BooleanField&& that);
```

Parameters

Name	Type	Description
that	BooleanField&&	A Boolean field to move.

5.3.3.1.6 ~BooleanField() Member Function

Destructor.

Syntax

```
public override nothrow ~BooleanField();
```

5.3.3.1.7 @operator_conv() const Member Function

Conversion function that returns the value of the field.

Syntax

```
public nothrow bool @operator_conv() const;
```

Returns

bool

Returns the value of the field.

5.3.3.1.8 FetchData(JsonValue*) Member Function

Populates the Boolean field with data retrieved from the specified JSON Boolean value.

Syntax

```
public override void FetchData(JsonValue* jsonValue);
```

Parameters

Name	Type	Description
jsonValue	JsonValue*	A JSON Boolean value.

5.3.3.1.9 GenerateDefinitions(CodeFormatter&) Member Function

Generates textual field definition.

Syntax

```
public override void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description

formatter	CodeFormatter&	A code formatter.

5.3.4 Container Class

Base class of Sequence and Struct.

Syntax

```
public class Container;
```

Base Class

Field

5.3.4.1 Member Functions

Member Function	Description
Container(const String&)	Constructor. Initializes the container with the specified field name.
Container(Container&&)	Move constructor.
Container(const Container&)	Copy constructor.
operator=(const Container&)	Copy assignment.
operator=(Container&&)	Move assignment.
~Container()	Destructor.
AddField(Field*)	Adds a field to the contained fields.
Fields() const	Returns a list of contained fields.
GenAddFields(CodeFormatter&)	Generates textual add field command.
GenerateDefinitions(CodeFormatter&)	Generates textual definition for contained fields.
SetTypeName(const String&)	Sets the type name of the container.
TypeName() const	Returns the type name of the container.

5.3.4.1.1 Container(const String&) Member Function

Constructor. Initializes the container with the specified field name.

Syntax

```
public Container(const String& fieldName_);
```

Parameters

Name	Type	Description
fieldName_	const String&	The name of the container field.

5.3.4.1.2 Container(Container&&) Member Function

Move constructor.

Syntax

```
public nothrow Container(Container&& that);
```

Parameters

Name	Type	Description
that	Container&&	A container to move.

5.3.4.1.3 Container(const Container&) Member Function

Copy constructor.

Syntax

```
public Container(const Container& that);
```

Parameters

Name	Type	Description
that	const Container&	A container to copy.

5.3.4.1.4 operator=(const Container&) Member Function

Copy assignment.

Syntax

```
public void operator=(const Container& that);
```

Parameters

Name	Type	Description
that	const Container&	A container to copy.

5.3.4.1.5 operator=(Container&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(Container&& that);
```

Parameters

Name	Type	Description
that	Container&&	A container to move.

5.3.4.1.6 ~Container() Member Function

Destructor.

Syntax

```
public override nothrow ~Container();
```

5.3.4.1.7 AddField(Field*) Member Function

Adds a field to the contained fields.

Syntax

```
public void AddField(Field* field);
```

Parameters

Name	Type	Description
field	Field*	A field to add.

5.3.4.1.8 Fields() const Member Function

Returns a list of contained fields.

Syntax

```
public nothrow const List<Field*>& Fields() const;
```

Returns

const List<Field*>&

Returns a list of contained fields.

5.3.4.1.9 GenAddFields(CodeFormatter&) Member Function

Generates textual add field command.

Syntax

```
public void GenAddFields(CodeFormatter& formatter);
```

Parameters

Name	Type	Description
formatter	CodeFormatter&	A code formatter.

5.3.4.1.10 GenerateDefinitions(CodeFormatter&) Member Function

Generates textual definition for contained fields.

Syntax

```
public override void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description
formatter	CodeFormatter&	A code formatter.

5.3.4.1.11 SetTypeName(const String&) Member Function

Sets the type name of the container.

Syntax

```
public void SetTypeName(const String& typeName_);
```

Parameters

Name	Type	Description
typeName_	const String&	The type name of the container.

5.3.4.1.12 TypeName() const Member Function

Returns the type name of the container.

Syntax

```
public nothrow const String& TypeName() const;
```

Returns

const String&

Returns the type name of the container.

5.3.5 ContainerField Class

Helper class for schema class generation.

Syntax

```
public class ContainerField;
```

Base Class

Field

5.3.5.1 Member Functions

Member Function	Description
ContainerField(ContainerField&&)	Move constructor.
ContainerField(const ContainerField&)	Copy constructor.
ContainerField(Container*, const String&)	Constructor. Initializes the container field with given container and field name.
operator=(ContainerField&&)	Move assignment.
operator=(const ContainerField&)	Copy assignment.
~ContainerField()	Destructor.
GenerateDefinitions(CodeFormatter&)	Generates textual definition of the container field.

5.3.5.1.1 ContainerField(ContainerField&&) Member Function

Move constructor.

Syntax

```
public nothrow ContainerField(ContainerField&& that);
```

Parameters

Name	Type	Description
that	ContainerField&&	Container field to move.

5.3.5.1.2 ContainerField(const ContainerField&) Member Function

Copy constructor.

Syntax

```
public nothrow ContainerField(const ContainerField& that);
```

Parameters

Name	Type	Description
that	const ContainerField&	Container field to copy.

5.3.5.1.3 ContainerField(Container*, const String&) Member Function

Constructor. Initializes the container field with given container and field name.

Syntax

```
public ContainerField(Container* container_, const String& fieldName_);
```

Parameters

Name	Type	Description
container_	Container*	Container.
fieldName_	const String&	Field name.

5.3.5.1.4 operator=(ContainerField&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(ContainerField&& that);
```

Parameters

Name	Type	Description
that	ContainerField&&	A container field to move.

5.3.5.1.5 operator=(const ContainerField&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const ContainerField& that);
```

Parameters

Name	Type	Description
that	const ContainerField&	A container field to assign.

5.3.5.1.6 ~ContainerField() Member Function

Destructor.

Syntax

```
public override nothrow ~ContainerField();
```

5.3.5.1.7 GenerateDefinitions(CodeFormatter&) Member Function

Generates textual definition of the container field.

Syntax

```
public override void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description
formatter	CodeFormatter&	A code formatter.

5.3.6 Field Class

Base class for all schema field types.

Syntax

```
public abstract class Field;
```

5.3.6.1 Member Functions

Member Function	Description
Field(const String&)	Constructor. Initializes the field with given name.
Field(Field&&)	Move constructor.
Field(const Field&)	Copy constructor.
operator=(Field&&)	Move assignment.
operator=(const Field&)	Copy assignment.
~Field()	Destructor.
FetchData(JsonValue*)	Virtual member function for populating schema field with JSON data.
GenerateDefinitions(CodeFormatter&)	Abstract member function for generating textual representation of the field.
Name() const	Returns the name of the field.

5.3.6.1.1 Field(const String&) Member Function

Constructor. Initializes the field with given name.

Syntax

```
public Field(const String& name_);
```

Parameters

Name	Type	Description
name_	const String&	The name of the field.

5.3.6.1.2 Field(Field&&) Member Function

Move constructor.

Syntax

```
public nothrow Field(Field&& that);
```

Parameters

Name	Type	Description
that	Field&&	A field to move.

5.3.6.1.3 Field(const Field&) Member Function

Copy constructor.

Syntax

```
public nothrow Field(const Field& that);
```

Parameters

Name	Type	Description
that	const Field&	A field to copy.

5.3.6.1.4 operator=(Field&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(Field&& that);
```

Parameters

Name	Type	Description
that	Field&&	A field to move.

5.3.6.1.5 operator=(const Field&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const Field& that);
```

Parameters

Name	Type	Description
that	const Field&	A field to assign.

5.3.6.1.6 ~Field() Member Function

Destructor.

Syntax

```
public virtual nothrow ~Field();
```

5.3.6.1.7 FetchData(JsonValue*) Member Function

Virtual member function for populating schema field with JSON data.

Syntax

```
public virtual void FetchData(JsonValue* jsonValue);
```

Parameters

Name	Type	Description
jsonValue	JsonValue*	Data in JSON format.

5.3.6.1.8 GenerateDefinitions(CodeFormatter&) Member Function

Abstract member function for generating textual representation of the field.

Syntax

```
public abstract void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description
formatter	CodeFormatter&	A code formatter.

5.3.6.1.9 Name() const Member Function

Returns the name of the field.

Syntax

```
public nothrow const String& Name() const;
```

Returns

const String&

Returns the name of the field.

5.3.7 NumberField Class

Represents numeric field.

Syntax

```
public class NumberField;
```

Base Class

Field

5.3.7.1 Member Functions

Member Function	Description
NumberField(NumberField&&)	Move constructor.
NumberField(const NumberField&)	Copy constructor.
NumberField(const String&)	Constructor. Initializes the number field with the given name.
operator=(const NumberField&)	Copy assignment.
operator=(NumberField&&)	Move assignment.
~NumberField()	Destructor.
@operator_conv() const	Conversion function that returns that value of the number field.
FetchData(JsonValue*)	Populates the number field with data retrieved from the specified JSON number value.
GenerateDefinitions(CodeFormatter&)	Generates textual definition of the number field.

5.3.7.1.1 NumberField(NumberField&&) Member Function

Move constructor.

Syntax

```
public nothrow NumberField(NumberField&& that);
```

Parameters

Name	Type	Description
that	NumberField&&	A number field to move.

5.3.7.1.2 NumberField(const NumberField&) Member Function

Copy constructor.

Syntax

```
public nothrow NumberField(const NumberField& that);
```

Parameters

Name	Type	Description
that	const NumberField&	A number field to copy.

5.3.7.1.3 NumberField(const String&) Member Function

Constructor. Initializes the number field with the given name.

Syntax

```
public NumberField(const String& fieldName_);
```

Parameters

Name	Type	Description
fieldName_	const String&	The name of the field.

5.3.7.1.4 operator=(const NumberField&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const NumberField& that);
```

Parameters

Name	Type	Description
that	const NumberField&	A number field to assign.

5.3.7.1.5 operator=(NumberField&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(NumberField&& that);
```

Parameters

Name	Type	Description
that	NumberField&&	A number field to move.

5.3.7.1.6 ~NumberField() Member Function

Destructor.

Syntax

```
public override nothrow ~NumberField();
```

5.3.7.1.7 @operator_conv() const Member Function

Conversion function that returns that value of the number field.

Syntax

```
public nothrow double @operator_conv() const;
```

Returns

double

Returns that value of the number field.

5.3.7.1.8 FetchData(JsonValue*) Member Function

Populates the number field with data retrieved from the specified JSON number value.

Syntax

```
public override void FetchData(JsonValue* jsonValue);
```

Parameters

Name	Type	Description
jsonValue	JsonValue*	A JSON number value.

5.3.7.1.9 GenerateDefinitions(CodeFormatter&) Member Function

Generates textual definition of the number field.

Syntax

```
public override void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description

formatter	CodeFormatter&	A code formatter.

5.3.8 SchemaFileContent Class

Represents contents of a schema file.

Syntax

```
public class SchemaFileContent;
```

5.3.8.1 Member Functions

Member Function	Description
SchemaFileContent()	Default constructor. Initializes empty schema file content.
~SchemaFileContent()	Destructor.
AddContainer(Container*)	Adds a container to schema file content.
AddField(Field*)	Adds a field to schema file content.
GenerateDefinitions(CodeFormatter&)	Generates textual Cmajor definitions for the contents of a schema file.
NewContainerField(const String&, const String&)	Creates a new container field when container name and field name are given.

5.3.8.1.1 SchemaFileContent() Member Function

Default constructor. Initializes empty schema file content.

Syntax

```
public SchemaFileContent();
```

5.3.8.1.2 ~SchemaFileContent() Member Function

Destructor.

Syntax

```
public nothrow ~SchemaFileContent();
```

5.3.8.1.3 AddContainer(Container*) Member Function

Adds a container to schema file content.

Syntax

```
public void AddContainer(Container* container);
```

Parameters

Name	Type	Description
container	Container*	A container to add.

5.3.8.1.4 AddField(Field*) Member Function

Adds a field to schema file content.

Syntax

```
public void AddField(Field* field);
```

Parameters

Name	Type	Description
field	Field*	A field to add.

5.3.8.1.5 GenerateDefinitions(CodeFormatter&) Member Function

Generates textual Cmajor definitions for the contents of a schema file.

Syntax

```
public void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description
formatter	CodeFormatter&	A code formatter.

5.3.8.1.6 NewContainerField(const String&, const String&) Member Function

Creates a new container field when container name and field name are given.

Syntax

```
public Field* NewContainerField(const String& containerName, const String& fieldName);
```

Parameters

Name	Type	Description
containerName	const String&	Name of existing container in a schema file.

fieldName	const String&	Name of container field.

Returns

Field*

Returns a new container field.

5.3.9 Sequence Class

Represents a sequence of fields. Differs from ArrayField<ItemType> in that number of fields must be fixed and fields can be polymorphic.

Syntax

```
public class Sequence;
```

Base Class

Container

5.3.9.1 Member Functions

Member Function	Description
Sequence(Sequence&&)	Move constructor.
Sequence(const String&)	Constructor. Initializes a sequence with the specified field name.
Sequence(const Sequence&)	Copy constructor.
operator=(const Sequence&)	Copy assignment.
operator=(Sequence&&)	Move assignment.
~Sequence()	Destructor.
FetchData(JsonValue*)	Populates the sequence field with data retrieved from the specified JSON array.
GenerateDefinitions(CodeFormatter&)	Generates textual definition of the sequence field.

5.3.9.1.1 Sequence(Sequence&&) Member Function

Move constructor.

Syntax

```
public nothrow Sequence(Sequence&& that);
```

Parameters

Name	Type	Description

that	Sequence&&	A sequence to move.

5.3.9.1.2 Sequence(const String&) Member Function

Constructor. Initializes a sequence with the specified field name.

Syntax

```
public Sequence(const String& fieldName_);
```

Parameters

Name	Type	Description
fieldName_	const String&	The name of the sequence field.

5.3.9.1.3 Sequence(const Sequence&) Member Function

Copy constructor.

Syntax

```
public Sequence(const Sequence& that);
```

Parameters

Name	Type	Description
that	const Sequence&	A sequence to copy.

5.3.9.1.4 operator=(const Sequence&) Member Function

Copy assignment.

Syntax

```
public void operator=(const Sequence& that);
```

Parameters

Name	Type	Description
that	const Sequence&	A sequence to assign.

5.3.9.1.5 operator=(Sequence&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(Sequence&& that);
```

Parameters

Name	Type	Description
that	Sequence&&	A sequence to move.

5.3.9.1.6 ~Sequence() Member Function

Destructor.

Syntax

```
public override nothrow ~Sequence();
```

5.3.9.1.7 FetchData(JsonValue*) Member Function

Populates the sequence field with data retrieved from the specified JSON array.

Syntax

```
public override void FetchData(JsonValue* value);
```

Parameters

Name	Type	Description
value	JsonValue*	A JSON array.

5.3.9.1.8 GenerateDefinitions(CodeFormatter&) Member Function

Generates textual definition of the sequence field.

Syntax

```
public override void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description
formatter	CodeFormatter&	A code formatter.

5.3.10 StringField Class

Represents a string field.

Syntax

```
public class StringField;
```

Base Class

Field

5.3.10.1 Member Functions

Member Function	Description
StringField(const StringField&)	Copy constructor.
StringField(const String&)	Constructor. Initializes the string field with the specified field name.
StringField(StringField&&)	Move constructor.
operator=(const StringField&)	Copy assignment.
operator=(StringField&&)	Move assignment.
~StringField()	Destructor.
@operator_conv() const	Conversion operator that returns the value of the field.
FetchData(JsonValue*)	Populates the string field with data retrieved from the specified JSON string value.
GenerateDefinitions(CodeFormatter&)	Generates textual definition of the string field.

5.3.10.1.1 StringField(const StringField&) Member Function

Copy constructor.

Syntax

```
public nothrow StringField(const StringField& that);
```

Parameters

Name	Type	Description
that	const StringField&	A string field to copy.

5.3.10.1.2 StringField(const String&) Member Function

Constructor. Initializes the string field with the specified field name.

Syntax

```
public StringField(const String& fieldName_);
```

Parameters

Name	Type	Description
fieldName_	const String&	The name of the string field.

5.3.10.1.3 StringField(StringField&&) Member Function

Move constructor.

Syntax

```
public nothrow StringField(StringField&& that);
```

Parameters

Name	Type	Description
that	StringField&&	A string field to move.

5.3.10.1.4 operator=(const StringField&) Member Function

Copy assignment.

Syntax

```
public nothrow void operator=(const StringField& that);
```

Parameters

Name	Type	Description
that	const StringField&	A string field to assign.

5.3.10.1.5 operator=(StringField&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(StringField&& that);
```

Parameters

Name	Type	Description
that	StringField&&	A string field to move.

5.3.10.1.6 ~StringField() Member Function

Destructor.

Syntax

```
public override nothrow ~StringField();
```

5.3.10.1.7 @operator_conv() const Member Function

Conversion operator that returns the value of the field.

Syntax

```
public nothrow const String& @operator_conv() const;
```

Returns

const String&

Returns the value of the field.

5.3.10.1.8 FetchData(JsonValue*) Member Function

Populates the string field with data retrieved from the specified JSON string value.

Syntax

```
public override void FetchData(JsonValue* jsonValue);
```

Parameters

Name	Type	Description
jsonValue	JsonValue*	A JSON string.

5.3.10.1.9 GenerateDefinitions(CodeFormatter&) Member Function

Generates textual definition of the string field.

Syntax

```
public override void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description
formatter	CodeFormatter&	A code formatter.

5.3.11 Struct Class

Represents a structured field.

Syntax

```
public class Struct;
```

Base Class

Container

5.3.11.1 Member Functions

Member Function	Description
Struct()	Default constructor. Initializes the structured field with dummy field name.
Struct(const String&)	Constructor. Initializes the structured field with the specicied field name.
Struct(const Struct&)	Copy constructor.
Struct(Struct&&)	Move constructor.
operator=(const Struct&)	Copy assignment.
operator=(Struct&&)	Move assignment.
~Struct()	Destructor.
FetchData(JsonValue*)	Populates the structued field with data retrieved from the specified JSON object.
GenerateDefinitions(CodeFormatter&)	Generates textual definition of the structured field.

5.3.11.1.1 Struct() Member Function

Default constructor. Initializes the structured field with dummy field name.

Syntax

```
public Struct();
```

5.3.11.1.2 Struct(const String&) Member Function

Constructor. Initializes the structured field with the specicied field name.

Syntax

```
public Struct(const String& fieldName_);
```

Parameters

Name	Type	Description
fieldName_	const String&	The name of the structured field.

5.3.11.1.3 Struct(const Struct&) Member Function

Copy constructor.

Syntax

```
public Struct(const Struct& that);
```

Parameters

Name	Type	Description
that	const Struct&	A structured field to copy.

5.3.11.1.4 Struct(Struct&&) Member Function

Move constructor.

Syntax

```
public nothrow Struct(Struct&& that);
```

Parameters

Name	Type	Description
that	Struct&&	A structured field to move.

5.3.11.1.5 operator=(const Struct&) Member Function

Copy assignment.

Syntax

```
public void operator=(const Struct& that);
```

Parameters

Name	Type	Description
that	const Struct&	A structured field to assign.

5.3.11.1.6 operator=(Struct&&) Member Function

Move assignment.

Syntax

```
public nothrow void operator=(Struct&& that);
```

Parameters

Name	Type	Description
that	Struct&&	A structured field to move.

5.3.11.1.7 ~Struct() Member Function

Destructor.

Syntax

```
public override nothrow ~Struct();
```

5.3.11.1.8 FetchData(JsonValue*) Member Function

Populates the structued field with data retrieved from the specified JSON object.

Syntax

```
public override void FetchData(JsonValue* value);
```

Parameters

Name	Type	Description
value	JsonValue*	A JSON object.

5.3.11.1.9 GenerateDefinitions(CodeFormatter&) Member Function

Generates textual definition of the structured field.

Syntax

```
public override void GenerateDefinitions(CodeFormatter& formatter);
```

Parameters

Name	Type	Description
formatter	CodeFormatter&	A code formatter.