JSONParser Class

Represents a parser for JSON-encoded content.

Namespace

System

Usage

Use the `System.JSONParser` methods to parse a response that's returned from a call to an external service that is in JSON format, such as a JSON-encoded response of a Web service callout.

See Also

- *Apex Developer Guide*: JSON Parsing

JSONParser Methods

The following are methods for `JSONParser`. All are instance methods.

- **clearCurrentToken()**
  Removes the current token.
- **getBlobValue()**
  Returns the current token as a BLOB value.
- **getBooleanValue()**
  Returns the current token as a Boolean value.
- **getCurrentName()**
  Returns the name associated with the current token.
- **getCurrentToken()**
  Returns the token that the parser currently points to or `null` if there's no current token.
- **getDatetimeValue()**
  Returns the current token as a date and time value.
- **getDateValue()**
  Returns the current token as a date value.
- **getDecimalValue()**
  Returns the current token as a decimal value.
- **getDoubleValue()**
  Returns the current token as a double value.
- **getIdValue()**
  Returns the current token as an ID value.
- **getIntegerValue()**
  Returns the current token as an integer value.
- **getLastClearedToken()**
  Returns the last token that was cleared by the `clearCurrentToken` method.
- **getLongValue()**
  Returns the current token as a long value.

- **hasCurrentToken()**
  Returns `true` if the parser currently points to a token; otherwise, returns `false`.
- **nextToken()**
  Returns the next token or `null` if the parser has reached the end of the input stream.
- **nextValue()**
  Returns the next token that is a value type or `null` if the parser has reached the end of the input stream.
- **readValueAs(apexType)**
  Deserializes JSON content into an object of the specified Apex type and returns the deserialized object.
- **readValueAsStrict(apexType)**
  Deserializes JSON content into an object of the specified Apex type and returns the deserialized object. All attributes in the JSON content must be present in the specified type.
- **skipChildren()**
  Skips all child tokens of type `JSONToken.START_ARRAY` and `JSONToken.START_OBJECT` that the parser currently points to.

clearCurrentToken()

Removes the current token.

Signature

```
public Void clearCurrentToken()
```

Return Value

Type: Void

Usage

After this method is called, a call to `hasCurrentToken` returns `false` and a call to `getCurrentToken` returns `null`. You can retrieve the cleared token by calling `getLastClearedToken`.

getBlobValue()

Returns the current token as a BLOB value.

Signature

```
public Blob getBlobValue()
```

Return Value

Type: Blob

Usage

The current token must be of type `JSONToken.VALUE_STRING` and must be Base64-encoded.

getBooleanValue()

Returns the current token as a Boolean value.

Signature

```
public Boolean getBooleanValue()
```

Return Value

The following example parses a sample JSON string and retrieves a Boolean value.

```
String JSONContent =
    '{"isActive":true}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the start object marker.
parser.nextToken();
// Advance to the next value.
parser.nextValue();
// Get the Boolean value.
Boolean isActive = parser.getBooleanValue();
```

getCurrentName()

Returns the name associated with the current token.

Signature

```
public String getCurrentName()
```

Return Value

Type: String

Usage

If the current token is of type `JSONToken.FIELD_NAME`, this method returns the same value as `getText`. If the current token is a value, this method returns the field name that precedes this token. For other values such as array values or root-level values, this method returns `null`.

The following example parses a sample JSON string. It advances to the field value and retrieves its corresponding field name.

Example

```
String JSONContent = '{"firstName":"John"}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the start object marker.
parser.nextToken();
// Advance to the next value.
parser.nextValue();
// Get the field name for the current value.
String fieldName = parser.getCurrentName();
// Get the textual representation
// of the value.
String fieldValue = parser.getText();
```

getCurrentToken()

Returns the token that the parser currently points to or `null` if there's no current token.

Signature

```
public System.JSONToken getCurrentToken()
```

Return Value

Type: System.JSONToken

```
String JSONContent = '{"firstName":"John"}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the next token.
while (parser.nextToken() != null) {
    System.debug('Current token: ' +
        parser.getCurrentToken());
}
```

getDatetimeValue()

Returns the current token as a date and time value.

Signature

```
public Datetime getDatetimeValue()
```

Return Value

Type: Datetime

Usage

The current token must be of type `JSONToken.VALUE_STRING` and must represent a `Datetime` value in the ISO-8601 format.

The following example parses a sample JSON string and retrieves a Datetime value.

```
String JSONContent =
'{"transactionDate":"2011-03-22T13:01:23"}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the start object marker.
parser.nextToken();
// Advance to the next value.
parser.nextValue();
// Get the transaction date.
Datetime transactionDate =
   parser.getDatetimeValue();
```

getDateValue()

Returns the current token as a date value.

Signature

```
public Date getDateValue()
```

Return Value

Type: Date

Usage

The current token must be of type `JSONToken.VALUE_STRING` and must represent a `Date` value in the ISO-8601 format.

The following example parses a sample JSON string and retrieves a Date value.

```
String JSONContent =
   '{"dateOfBirth":"2011-03-22"}';
```

```
// Get the date of birth.
Date dob = parser.getDateValue();
```

getDecimalValue()

Returns the current token as a decimal value.

Signature

```
public Decimal getDecimalValue()
```

Return Value

Type: Decimal

Usage

The current token must be of type `JSONToken.VALUE_NUMBER_FLOAT` or `JSONToken.VALUE_NUMBER_INT` and is a numerical value that can be converted to a value of type `Decimal`.

The following example parses a sample JSON string and retrieves a Decimal value.

```
String JSONContent =
   '{"GPA":3.8}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the start object marker.
parser.nextToken();
// Advance to the next value.
parser.nextValue();
// Get the GPA score.
Decimal gpa = parser.getDecimalValue();
```

getDoubleValue()

Returns the current token as a double value.

Signature

```
public Double getDoubleValue()
```

Return Value

Type: Double

Usage

The current token must be of type `JSONToken.VALUE_NUMBER_FLOAT` and is a numerical value that can be converted to a value of type `Double`.

The following example parses a sample JSON string and retrieves a Double value.

```
String JSONContent =
   '{"GPA":3.8}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the start object marker.
parser.nextToken();
// Advance to the next value.
parser.nextValue();
```

Returns the current token as an ID value.

Signature

```
public ID getIdValue()
```

Return Value

Type: ID

Usage

The current token must be of type `JSONToken.VALUE_STRING` and must be a valid `ID`.

The following example parses a sample JSON string and retrieves an ID value.

```
String JSONContent =
   '{"recordId":"001R0000002nO6H"}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the start object marker.
parser.nextToken();
// Advance to the next value.
parser.nextValue();
// Get the record ID.
ID recordID = parser.getIdValue();
```

getIntegerValue()

Returns the current token as an integer value.

Signature

```
public Integer getIntegerValue()
```

Return Value

Type: Integer

Usage

The current token must be of type `JSONToken.VALUE_NUMBER_INT` and must represent an `Integer`.

The following example parses a sample JSON string and retrieves an Integer value.

```
String JSONContent =
   '{"recordCount":10}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the start object marker.
parser.nextToken();
// Advance to the next value.
parser.nextValue();
// Get the record count.
Integer count = parser.getIntegerValue();
```

getLastClearedToken()

Returns the last token that was cleared by the `clearCurrentToken` method.

Type: System.JSONToken

getLongValue()

Returns the current token as a long value.

Signature

```
public Long getLongValue()
```

Return Value

Type: Long

Usage

The current token must be of type `JSONToken.VALUE_NUMBER_INT` and is a numerical value that can be converted to a value of type `Long` .

The following example parses a sample JSON string and retrieves a Long value.

```
String JSONContent =
   '{"recordCount":2097531021}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the start object marker.
parser.nextToken();
// Advance to the next value.
parser.nextValue();
// Get the record count.
Long count = parser.getLongValue();
```

getText()

Returns the textual representation of the current token or `null` if there's no current token.

Signature

```
public String getText()
```

Return Value

Type: String

Usage

No current token exists, and therefore this method returns `null`, if `nextToken` has not been called yet for the first time or if the parser has reached the end of the input stream.

getTimeValue()

Returns the current token as a time value.

Signature

```
public Time getTimeValue()
```

Return Value

Type: Time

The following example parses a sample JSON string and retrieves a Datetime value.

```
String JSONContent =
   '{"arrivalTime":"18:05"}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Advance to the start object marker.
parser.nextToken();
// Advance to the next value.
parser.nextValue();
// Get the arrival time.
Time arrivalTime = parser.getTimeValue();
```

hasCurrentToken()

Returns `true` if the parser currently points to a token; otherwise, returns `false`.

Signature

```
public Boolean hasCurrentToken()
```

Return Value

Type: Boolean

nextToken()

Returns the next token or `null` if the parser has reached the end of the input stream.

Signature

```
public System.JSONToken nextToken()
```

Return Value

Type: System.JSONToken

Usage

Advances the stream enough to determine the type of the next token, if any.

nextValue()

Returns the next token that is a value type or `null` if the parser has reached the end of the input stream.

Signature

```
public System.JSONToken nextValue()
```

Return Value

Type: System.JSONToken

Usage

Advances the stream enough to determine the type of the next token that is of a value type, if any, including a JSON array and object start and end markers.

readValueAs(apexType)

```
public Object readValueAs(System.Type apexType)
```

Parameters

apexType

Type: System.Type

The *apexType* argument specifies the type of the object that this method returns after deserializing the current value.

Return Value

Type: Object

Usage

If the JSON content contains attributes not present in the `System.Type` argument, such as a missing field or object, deserialization fails in some circumstances. When deserializing JSON content into a custom object or an sObject using Salesforce API version 34.0 or earlier, this method throws a runtime exception when passed extraneous attributes. When deserializing JSON content into an Apex class in any API version, or into an object in API version 35.0 or later, no exception is thrown. When no exception is thrown, this method ignores extraneous attributes and parses the rest of the JSON content.

Example

The following example parses a sample JSON string and retrieves a Datetime value. Before being able to run this sample, you must create a new Apex class as follows:

```
public class Person {
    public String name;
    public String phone;
}
```

Next, insert the following sample in a class method:

```
// JSON string that contains a Person object.
String JSONContent =
    '{"person":{' +
        '"name":"John Smith",' +
        '"phone":"555-1212"}}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Make calls to nextToken()
// to point to the second
// start object marker.
parser.nextToken();
parser.nextToken();
parser.nextToken();
// Retrieve the Person object
// from the JSON string.
Person obj =
   (Person)parser.readValueAs(
      Person.class);
System.assertEquals(
   obj.name, 'John Smith');
System.assertEquals(
   obj.phone, '555-1212');
```

readValueAsStrict(apexType)

```
public Object readValueAsStrict(System.Type apexType)
```

Parameters

apexType

Type: System.Type

The *apexType* argument specifies the type of the object that this method returns after deserializing the current value.

Return Value

Type: Object

Usage

If the JSON content contains attributes not present in the `System.Type` argument, such as a missing field or object, deserialization fails in some circumstances. When deserializing JSON content with extraneous attributes into an Apex class, this method throws an exception in all API versions. However, no exception is thrown when you use this method to deserialize JSON content into a custom object or an sObject.

The following example parses a sample JSON string and retrieves a Datetime value. Before being able to run this sample, you must create a new Apex class as follows:

```
public class Person {
    public String name;
    public String phone;
}
```

Next, insert the following sample in a class method:

```
// JSON string that contains a Person object.
String JSONContent =
    '{"person":{' +
        '"name":"John Smith",' +
        '"phone":"555-1212"}}';
JSONParser parser =
   JSON.createParser(JSONContent);
// Make calls to nextToken()
// to point to the second
// start object marker.
parser.nextToken();
parser.nextToken();
parser.nextToken();
// Retrieve the Person object
// from the JSON string.
Person obj =
   (Person)parser.readValueAsStrict(
      Person.class);
System.assertEquals(
   obj.name, 'John Smith');
System.assertEquals(
   obj.phone, '555-1212');
```

skipChildren()

Skips all child tokens of type `JSONToken.START_ARRAY` and `JSONToken.START_OBJECT` that the parser currently points to.

Signature

DID THIS ARTICLE SOLVE YOUR ISSUE?
Let us know so we can improve!

Share your feedback

DEVELOPER CENTERS

- Heroku
- MuleSoft
- Tableau
- Commerce Cloud
- Lightning Design System
- Einstein
- Quip

POPULAR RESOURCES

- Documentation
- Component Library
- APIs
- Trailhead
- Sample Apps
- Podcasts
- AppExchange

COMMUNITY

- Trailblazer Community
- Events and Calendar
- Partner Community
- Blog
- Salesforce Admins
- Salesforce Architects