TypeScript: JSX

Appendix: JSX Specification

Version 1.8

January, 2016

Microsoft is making this Specification available under the Open Web Foundation Final Specification Agreement Version 1.0 ("OWF 1.0") as of October 1, 2012. The OWF 1.0 is available at <http://www.openwebfoundation.org/legal/the-owf-1-0-agreements/owfa-1-0>.

TypeScript is a trademark of Microsoft Corporation.

Table of Contents

[1 Changes in JSX 1](#_Toc439860476)

[1.1 Changes in Grammar 1](#_Toc439860477)

[1.1.1 Type Assertions 1](#_Toc439860478)

[1.1.2 Generic Arrow Functions 1](#_Toc439860479)

[1.1.3 JSX 1](#_Toc439860480)

[2 JSX Expressions 1](#_Toc439860481)

[2.1 JSX Elements 1](#_Toc439860482)

[2.1.1 Intrinsic Elements 1](#_Toc439860483)

[2.1.2 Non-intrinsic Elements 1](#_Toc439860484)

[2.2 JSX Attributes 1](#_Toc439860485)

[2.2.1 Augmented Attributes Type 1](#_Toc439860486)

[2.2.2 Required Attributes 1](#_Toc439860487)

[2.2.3 Surplus and Duplicate Attributes 1](#_Toc439860488)

[2.2.4 Type Checking of Attributes 1](#_Toc439860489)

[2.2.5 Type Checking of Spread Attributes 1](#_Toc439860490)

# Changes in JSX

This appendix specifies the behavior of the JSX variant of the TypeScript language. This variant adds support for JSX, an XML-like extension to the JavaScript language. Its grammar can be found at the [Draft JSX Specification website](https://facebook.github.io/jsx/).

Source files ending in .tsx are ***tsx files*** and use this modified language.

## Changes in Grammar

The TypeScript grammar is modified in tsx files

### Type Assertions

The [Type Assertion](#_Type_Assertions) production is removed. Instead, the as operator (*TODO: Spec and link*) must instead be used to assert the type of an expression.

### Generic Arrow Functions

In tsx files, a generic arrow function with a single type parameter must have a constraint to be parsed as a generic arrow function (rather than as a JSX opening element).

### JSX

All productions specified by the Draft JSX Specification are added to the language with the exception of JSXNamespacedName.

# JSX Expressions

Expressions from the JSXElement production are ***JSX Elements***. The type of a JSX Element expression is the global type JSX.Element, if that type exists, otherwise they are of type any.

The attributes of a JSX Element must conform to its ***element attributes type***. This type is determined differently depending on tag name of the JSX Element. The next section outlines each kind of element and how the element attributes type is determined.

## JSX Elements

JSX Elements may either be ***intrinsic*** or ***non-intrinsic***. An element is intrinsic if it is an identifier which begins with a lower-case character. Otherwise, it is non-intrinsic.

### 2.1.1 Intrinsic Elements

The element attributes type of an intrinsic element is determined as follows:

* If the global type JSX.IntrinsicElements exists, then:
  + If a property of JSX.IntrinsicElements with the same name as the tag name exists, the element attributes type is the type of that property
  + Otherwise, if JSX.IntrinsicElements contains a string index signature, the element attributes type is the type of the string indexer
* Otherwise, an error occurs

### 2.1.2 Non-intrinsic Elements

Non-intrinsic elements have an ***element type***. The element type of an element is the type of the expression that the element’s tag name resolves to. An error occurs if this expression cannot be resolved.

Non-intrinsic elements may be ***functional*** or ***class-based***. A non-intrinsic element is functional if all of the following are true, otherwise is it class-based:

* The element type has at least one call signature
* The first call signature of the element type is not generic
* The return type of the element’s first call signature is assignable to the global type JSX.Element, or the global type JSX.Element does not exist

The element attributes type of a functional element is the type of the first parameter of the first call signature of the element type, or the empty object type {} if the signature has no parameters.

Class-based elements have an ***element instance type***. The element instance type is determined as follows:

* If the element type is any, the element instance type is any
* If the element type has at least one construct signature, the element instance type is the union type whose constituents are the return types of those construct signatures
* If the element type has at least one call signature, the element instance type is the union type whose constituents are the return types of those call signatures
* Otherwise, an error occurs

An error occurs if the global type JSX.ElementClass exists and the element instance type is not assignable to it.

Given an element instance type for a class-based element, the element attributes type is determined as follows:

* If the element instance type is any, the element attributes type is any
* If the global type JSX.ElementAttributesProperty exists and has zero properties, the element attributes type is the element instance type
* If the global type JSX.ElementAttributesProperty exists and has one property *propertiesName*:
  + If the element instance type has a property *propertiesName*, the element attributes type is the type of that property
  + Otherwise, the element attributes type is the empty object type
* If the global type JSX.ElementAttributesProperty exists and has more than one property, an error occurs
* Otherwise, the element attributes type is any

## JSX Attributes

A JSX Element must specify all required attributes and not specify attributes which do not exist. This section describes how valid, required, and invalid attribute names are determined.

### Augmented Attributes Type

The ***augmented attributes type*** of a JSX element is determined as follows:

* If the element attributes type is any, the augmented attributes type is any.
* For a class-based element, the augmented attributes type is the intersection of the element instance type, the global type JSX.IntrinsicAttributes if it exists, and the global type JSX.IntrinsicClassAttributes if it exists and has zero or one type parameters. The type JSX.IntrinsicClassAttributes is instantiated with the element instance type as the type parameter if it is generic.
* For a functional element, the augmented attributes type is the intersection of the element instance type and the global type JSX.IntrinsicAttributes, if it exists.
* For an intrinsic element, the augmented attributes type is the element attributes type.

### Required Attributes

For each non-optional property in the augmented attributes type of a JSX element, one of the following must be true or an error occurs:

* The element contains an attribute with the same name as the property.
* The element contains a spread attribute and the type of the spreaded expression is any.
* The element contains a spread attribute and the type of the spreaded expression has a property with the same name as the attribute.

### Surplus and Duplicate Attributes

For each attribute specified in the element, one of the following must be true or an error occurs:

* The element attributes type is any
* The augmented element attributes type has a string indexer
* The attribute name contains a hyphen
* The augmented element attributes type contains an apparent member with the same name as the attribute

It is an error to specify two attributes with the same name in a given JSX Element.

### Type Checking of Attributes

For a given attribute, its ***corresponding property type*** is determined as follows:

* If the element attributes type is any, the corresponding property type is any
* If the augmented element attributes type contains a property with the same name as the attribute, the corresponding property type is the type of that property
* If the augmented element attributes type contains a string indexer, the corresponding property type is the type of the indexer
* Otherwise, there is no corresponding property type

For each attribute in an element, one the following must be true:

* There is no corresponding property type
* The attribute does not provide an expression, and the primitive type boolean is assignable to the corresponding property type
* The attribute provides a literal string expression, and the primitive type string is assignable to the corresponding property type
* The attribute provides an expression, and the result of processing this expression with the contextual type of the corresponding property type is assignable to the corresponding property type

### Type Checking of Spread Attributes

For a spread attribute whose spreaded expression is not of type any, one of the following must be true for each property of the type of the expression:

* The augmented element attributes type is any
* The containing element contains a spread attribute to the right of the given attribute where the type of the other spreaded expression is any
* The containing element contains a spread attribute to the right of the given attribute where the type of the other spreaded expression is has a property with the same name as the given property
* The containing element contains an attribute to the right of the given attribute with the same name as the given property
* There is no property with the same name in the augmented element attributes type
* There is a property with the same name in the augmented element attributes type, and the type of that property is assignable from the type of the property in the spreaded expression

## JSX Children

There are no type-based constraints on which JSX Elements, JSX Expressions, or JSX Text may appear as the child of a JSX Element.