创心

第十四届

D2前端技术论坛

2019-12-14 杭州和达希尔顿逸林酒店

Let's work together on the future of JavaScript through TC39

Daniel Ehrenberg
Igalia, in partnership with Bloomberg
D2 2019

D2-语言框架专场-(9:30-10:1...

7人

扫一扫群二维码，立刻加入该群。

我

- Daniel Ehrenberg
- @littledan
- Delegate in TC39

TC39

家
Les Roquetes del Garraf, Europe

工作

igalia

Embedded WebKit and Chromium development

Mesa and GStreamer drivers

CSS, ARIA, WebAssembly, MathML, JavaScript
standards+implementation in web browsers

學而時習之、不亦說乎。

Isn't it a pleasure to study and practice what you have learned?

有朋自遠方來、不亦樂乎。

Isn't it also great when friends visit from distant places?

人不知而不慍、不亦君子乎。

If one remains not annoyed when he is not understood by people around him, isn't he a sage?

TC39

Who is TC39?

- A committee of Ecma, with…
- JS developers
- JavaScript engines
- Transpilers
- Frameworks, libraries
- Academics
- Major websites/app platforms
- Now, some Chinese companies!
- etc

TC 39

Search...

TABLE OF CONTENTS

Introduction
1 Scope
2 Conformance
3 Normative References
4 Overview
5 Notational Conventions
6 ECMAScript Data Types and Values
7 Abstract Operations
8 Executable Code and Execution Contexts
9 Ordinary and Exotic Objects Behaviours
10 ECMAScript Language: Source Code
11 ECMAScript Language: Lexical Grammar
12 ECMAScript Language: Expressions
13 ECMAScript Language: Statements and Declarati...
14 ECMAScript Language: Functions and Classes
15 ECMAScript Language: Scripts and Modules
16 Error Handling and Language Extensions
17 ECMAScript Standard Built-in Objects
18 The Global Object
19 Fundamental Objects
20 Numbers and Dates
21 Text Processing
22 Indexed Collections
23 Keyed Collections
24 Structured Data

Draft ECMA-262 / April 23, 2018

ECMAScript® 2019 Language Specification

ecma INTERNATIONAL

Contributing to this Specification

This specification is developed on GitHub with the help of the ECMAScript community. There are a number of ways to contribute to the development of this specification:

GitHub Repository: https://github.com/tc39/ecma262
Issues: All Issues, File a New Issue
Pull Requests: All Pull Requests, Create a New Pull Request
Test Suite: Test262
Editor: Brian Terlson (E-mail, Twitter, GitHub)

Community:

- Mailing list: es-discuss
- IRC: #tc39 on freenode

This repository Search Pull requests Issues Marketplace Explore

tc39 / ecma262

Unwatch 836 Star 6,408 Fork 446

Code Issues 157 Pull requests 52 Projects 0 Wiki Insights Settings

Status, process, and documents for ECMA262 https://tc39.github.io/ecma262/

Edit

ecmascript javascript Manage topics

1,191 commits 10 branches 10 releases 85 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

jmdyck and bterlson Editorial: Misc Editorial (#1182) ... Latest commit cbe4679 20 hours ago

img	Editorial: Store and rethrow module instantiation/evaluation errors	9 months ago
scripts	Meta: Build ES2016 in a subfolder.	2 years ago
workingdocs	Editorial: Use HTTPS for ecma-international.org URLs (#1017)	6 months ago
.editorconfig	Add .editorconfig	3 years ago
.gitignore	Editorial: reword InstanceofOperator copy and rename vars (closes #894)	11 months ago
.travis.yml	meta: require node 8 to build the spec	4 months ago
CODE_OF_CONDUCT.md	Add CoC.md reference file (#1168)	13 days ago
CONTRIBUTING.md	Meta: Add paragraph about resolving conflicts with master. (#1099)	2 months ago
FAQ.md	Fix example file name in FAQ.	3 years ago
README.md	meta: Add Community section to readme.md	a year ago
figure-2.uxf	Editorial: fix size of figure 2.	2 years ago

Meetings

- Every two months
- For three days
- Discuss language changes
- Seek consensus on proposal
  Stage advancement

有朋自遠方來、不亦樂乎。

Isn't it also great when friends visit from distant places?

igalia

6. Web compatibility issues / Needs Consensus PRs

✓	timebox	topic
✓	30m	Normative: Fix extending null
✓	5m	Normative: Make super() throw after evaluating args
✓	10m	Normative: make async iterators next/return/throw not p `undefined` when value is absent
✓	25m	Normative: Eliminate extra environment for eval in para initializers redux (slides)
✓	25m	Normative: TypedArray on prototypes web reality (low pr (slides)
✓	15m	Normative: make EnumerableOwnPropertyNames ordere

7. Overflow from previous meeting

✓	timebox	topic	presenter

8. Short (≤30m) Timeboxed Discussions

✓	timebox	topic
✓	20m	async-of grammar ambiguity
✓	30m	RegExp match indices performance feedback (slides)
~	30m	Policy on published code/polyfills in proposal repos
✓	30m (could be 10m)	JSExplain demo (code, demo)

9. Proposals

✓ represents an agenda item which has been presented, and does not indica

✓	stage	timebox	topic	present
✓	3	15m	Intl.RelativeTimeFormat for stage 4 (slides)	Zibi Bra
✓	3	15m	For-in order for Stage 4 (PR, tests, slides)	Kevin Gi

TC39 stages

- Stage 1: An idea under discussion
- Stage 2: We want to do this, and we have a first draft
- Stage 3: Basically final draft; ready to go
- Stage 4: 2+ implementations, tests ⇒ standard

Consensus-based decision-making

- TC39 never votes
  - (maybe on unimportant technicalities)
- TC39 is *consensus-seeking*
  - We work together to meet everyone's goals
  - "Does anyone object to stage advancement?"
  - Objections must have rationale, appropriate to stage
  - Technical concerns should be raised early
- We assume *good faith*

Stage 4 features

2+ implementations, tests ⇒ standard

BigInt: Stage 4!

```
const x = 2 ** 53;
```

⇒ x === 9007199254740992

```
const y = x + 1;
```

⇒ y === 9007199254740992

```
const x = 2n ** 53n;
```

⇒ x === 9007199254740992n

```
const y = x + 1n;
```

⇒ y === 9007199254740993n

```
const x = 2n ** 53n;
```

⇒ `x === 9007199254740992n`

```
const y = x + 1n;
```

⇒ `y === 9007199254740993n`

n stands for BigInt

67

68

Beta

10.4

Flag

Dynamic import(): Stage 4

Domenic Denicola

@littledan

Search or jump to... / Pull requests Issues Marketplace Explore

tc39 / proposal-dynamic-import

Watch 110 Star 1.5k Fork 48

Code Issues 11 Pull requests 1 Actions Projects 0 Security Insights

Example

```
<script>
  const main = document.querySelector("main");
  for (const link of document.querySelectorAll("nav > a")) {
    link.addEventListener("click", e => {
      e.preventDefault();

      import(`./section-modules/${link.dataset.entryModule}.js`)
        .then(module => {
          module.loadPageInto(main);
        })
        .catch(err => {
          main.textContent = err.message;
        });
    });
  }
</script>
```

Intl.RelativeTimeFormat: Stage 4

Zibi Braniecki

Intl.RelativeTimeFormat

- Shipped in Chrome and Firefox

```
let rtf = new Intl.RelativeTimeFormat("en");

rtf.format(100, "day");
// "in 100 days"



new Intl.RelativeTimeFormat("zh").format(100, "day")

// "100天后"
```

Optional chaining: Stage 4

Daniel Rosenwasser
Claude Pache
Gabriel Isenberg
Dustin Savery

Optional Property Access

```
let x = foo?.bar;

// Equivalent to...

let x = (foo !== null && foo !== undefined) ?
  foo.bar :
  undefined;
```

Nullish coalescing: Stage 4

Daniel Rosenwasser
Gabriel Isenberg

Nullish Coalescing

```
let x = foo() ?? bar();

// Equivalent to...

let tmp = foo();
let x = (tmp !== null && tmp !== undefined) ?
  tmp :
  bar();
```

Stage 3 features

Basically final draft; ready to go

WeakRefs: Stage 3

Sathya Gunasekaran
Till Schneidereit
Mark Miller
Dean Tribble

Read consistency

```
const w = new WeakRef(someObject);

...

if (w.deref()) {

  w.deref().foo(); // w.deref() here can not fail

}
```

Private fields and methods: Stage 3

Why?

```
class Counter extends HTMLElement {
  #x = 0;

  connectedCallback() {
    this.#render();
  }
  #render() {
    this.textContent =
        this.#x.toString();
  }
}
```

- Private methods encapsulate behavior
- You can access private fields inside private methods

is the new _

for strong encapsulation

```
class PublicCounter {
  _x = 0;
}

let c = new PublicCounter();
console.log(c._x);    // 0
```

```
class PrivateCounter {
  #x = 0;
}

let p = new PrivateCounter();
console.log(p.#x);    // SyntaxError
```

Stage 3

Stage 2 features

We want to do this, and we have a first draft

Decorators: Stage 2

Yehuda Katz
Ron Buckton

Syntax abstraction for attrs/props in Web Components

```
// Polymer abstraction
class XCustom extends PolymerElement {
  @property({ type: String, reflect: true
})
  address = '';
}
```

```
// Salesforce abstraction
import { api } from '@salesforce';

class InputAddress extends HTMLElement {
    @api address = '';
}
```

Example of using decorators to improve ergonomics.

Temporal: Stage 2

Maggie Pint
Philipp Dunkel

What time did this presentation start in Berlin?

```
let startDateTime =                          // Temporal.DateTime
    Temporal.now.dateTime().with({ hour: 9, minute: 30 });

let startAbsolute =                          // Temporal.Absolute
    startDateTime.inTimeZone(Temporal.now.timeZone());

let localizedToBerlin =                      // Temporal.DateTime
    startAbsolute.inTimeZone("Europe/Berlin");

Intl.DateTimeFormat("zh", { hour: "numeric", minute: "numeric" })
    .format(localizedToBerlin);              // 上午2:30
```

Stage 1 features

An idea under discussion

Pipeline operator: Stage 1

Gilbert Garza
J.S. Choi
James DiGioia

library.js

```
export function doubleSay(str) {
  return str + ", " + str;
}
export function capitalize(str) {
  return str[0].toUpperCase() +
      str.substring(1);
}
export function exclaim(str) {
  return str + "!";
}
```

ordinary.js

```
import { doubleSay, capitalize, exclaim }
          from "./library.js";
let result =
  exclaim(capitalize(doubleSay("hello")));
// ===> "Hello, hello!"
```

with-pipeline.js

```
import { doubleSay, capitalize, exclaim }
          from "./library.js";
let result = "hello"
  |> doubleSay
  |> capitalize
  |> exclaim;
// ===> "Hello, hello!"
```

Records and Tuples: Stage 1

Robin Ricard
Richard Button

```
const marketData = #[
    #{ ticker: "AAPL", lastPrice: 195.855 },
    #{ ticker: "SPY", lastPrice: 286.53 },
];
```

Operator overloading: Stage 1

Vector overloading: Usage

```
// Usage example
import { Vector } from "./vector.mjs";
with operators from Vector;

new Vector([1, 2, 3]) + new Vector([4, 5, 6])   // ==> new Vector([5, 7, 9])
3 * new Vector([1, 2, 3])                       // ==> new Vector([3, 6, 9])
new Vector([1, 2, 3]) == new Vector([1, 2, 3])  // ==> true
(new Vector([1, 2, 3]))[1]                      // ==> 2
```

Stage 0 features

Not even really at a stage!

BigDecimal: Stage 0

Andrew Paprocki

@littledan

"Why are Numbers broken in JS?"

@littledan

Problem and solution (?)

```
// Number (binary 64-bit floating point)

js> 0.1 + 0.2

                    =>          0.30000000000000004

// BigDecimal (???)

js> 0.1d + 0.2d

                    =>          0.3d
```

BigInt

```
const x = 2n ** 53n;
```

⇒ `x === 9007199254740992n`

```
const y = x + 1n;
```

⇒ `y === 9007199254740993n`

n stands for BigInt

Ancient history

- long/ulong types proposed (1999)
  - ES3 mostly library additions
  - ActionScript added numeric types!
  - ES4 abandoned
  - ES5.1 intentionally minimal
- ES6 proposed “value types”, including int64
  - ES6 deferred value types
    - Currently being started as "Records and Tuples"

Modern history

- Int64 to Stage 1, Nov 2016
- Stage 4 in June 2019
- Details worked out over this process….

學而時習之、不亦說乎。

Isn't it a pleasure to study and practice what you have learned?

Stage 1

Branch: master ▾ proposal-bigint / README.md

Find file | Copy path

undeadcat remove parseInt from readme. Was removed from spec after #101 a0602f9 on Mar 23

8 contributors

208 lines (138 sloc) | 15.9 KB

Raw | Blame | History

BigInt: Arbitrary precision integers in JavaScript

Daniel Ehrenberg, Igalia. Stage 3

Thanks for help and feedback on this effort from Brendan Eich, Waldemar Horwat, Jaro Sevcik, Benedikt Meurer, Michael Saboff, Adam Klein and others.

Code samples:

Find the nth prime:

```
// Takes a BigInt as an argument and returns a BigInt
function nthPrime(nth) {
  function isPrime(p) {
    for (let i = 2n; i < p; i++) {
      if (p % i === 0n) return false;
    }
    return true;
```

tc39 / proposal-bigint

Unwatch 48 | Star 244 | Fork 20

Code | Issues 12 | Pull requests 1 | Projects 0 | Wiki | Insights | Settings

Filters | is:issue | Labels | Milestones | New issue

Clear current search query, filters, and sorts

12 Open | 91 Closed | Author | Labels | Projects | Milestones | Assignee | Sort

- **Regarding "Don't break user intuition"** 49
  #30 by gweax was closed on Jun 30, 2017
- **Distinct or explicitly non-distinct 0 and -0** 6
  #29 by ckknight was closed on Mar 28, 2017
- **valueOf should return an Integer value, not a Number**
  #27 by leobalter was closed on Mar 29, 2017
- **Collect perf-cliff avoiding patterns across implementations before stage 4** 5
  #26 opened on Mar 23, 2017 by syg
- **Should unary + be made goofy because of asm.js?** 19
  #25 by syg was closed on Jun 30, 2017
- **Support JSON serialisation of BigInt values** 47
  #24 by claudepache was closed on May 25, 2017
- **Asymmetry TypedArray/DataView types between Number and Integer** 1
  #23 by littledan was closed on Jun 30, 2017
- **Switch arguments to asIntN/asUintN?**
  #21 by littledan was closed on Mar 10, 2017
- **Consider adding more functions to the standard library** 3

BigInt vs Int64

Great investigation by Benedikt Meurer, Jaroslav Sevcik

@littledan

```
2n ** 53n + 1.5;
```

⇒ ????

TypeError!

No precise answer, so refuse to give an answer

Best practice for numeric types

- BigInt for integers which range > 2^{53}
- Number for always-small integers

Stage 2

OF CONTENTS
duction
neric Types
difications to the Number grammar
tract Operations
MAScript Language: Expressions
Int Objects
dified algorithms
edArrays and DataViews
mics.wake(*typedArray*, *index*, *count*)
mics.store(*typedArray*, *index*, *value*)
TypedArray%.prototype.sort (*comparefn*)
pedArraySpeciesCreate (*exemplar*, *argumentLi*...
TypedArray%.prototype.fill (*value* [, *start* [, *en*...
TypedArray%.prototype.set (*array* [, *offset*])
TypedArray%.prototype.set(*typedArray* [, *offse*...
pedArray (*typedArray*)
tViewValue (*view*, *requestIndex*, *isLittleEndian*, ...
taView.prototype.getBigInt64 (*byteOffset* [, *litt*...
taView.prototype.getBigUint64 (*byteOffset* [, *li*...
taView.prototype.setBigInt64 (*byteOffset*, *valu*...
taView.prototype.setBigUint64 (*byteOffset*, *val*...
gents

Stage 3 Draft / April 24, 2018

BigInt

Introduction

This proposal adds arbitrary-precision integers to ECMAScript. For motivation and a high-level introduction, see the explainer document.

> INTEGRATION PLANS
>
> I attempted to write this specification draft with an eye towards intelligibility and clarifying issues, rather than looking exactly like the final specification. In particular, the translation of the operations on the Number type into the new form are omitted, as they are identical to the previous definitions and would be expressed only as a lengthy refactoring to skip when reading this text.
>
> If you find any part of this specification unclear or confusing, please file an issue.

1 Numeric Types

ECMAScript has two built-in numeric types: Number and BigInt. In this specification, every numeric type *T* contains a multiplicative identity value denoted *T*::unit. The specification types also have the following abstract operations, likewise denoted *T*::*op* for a given operation with specification name *op*. Unless noted otherwise, argument and

tc39 / **proposal-bigint**

Unwatch 48 | Star 244 | Fork 20

Code | Issues 12 | Pull requests 1 | Projects 0 | Wiki | Insights | Settings

Branch: master | proposal-bigint / spec.html | Find file | Copy path

littledan Editorial: Fix variable references — 1c66d06 8 hours ago

12 contributors

2244 lines (2098 sloc) | 125 KB — Raw | Blame | History

```
<!DOCTYPE html>
<meta charset="utf-8">
<pre class="metadata">
title: BigInt
status: proposal
stage: 3
location: https://github.com/tc39/proposal-bigint
copyright: false
contributors: Daniel Ehrenberg, Brendan Eich
</pre>
<style>
emu-issue, emu-todo, emu-motivation, emu-my-example, emu-integration-plans {
    margin: 1em 0;
    padding: .5em;
    padding-left: 1em;
    display: block;
}

```

Bugzilla | Search Bugs | Browse | Advanced Search | New Bug | My Dashboard

Bug 1366287

Implement BigInt

NEW Assigned to terpri

Edit Bug | Copy Summary | Stop Following

Get help with this page

Status (NEW bug which should be worked on in the next release/iteration)

Product: Core
Component: JavaScript Engine
Importance: P2 normal
Status: NEW

Reported: a year ago
Modified: 8 hours ago

People (Reporter: terpri, Assigned: terpri)

Assignee: Robin Templeton [:terpri]

Reporter: Robin Templeton [:terpri]
Triage Owner: Steven DeTar [:sdetar]
CC: 29 people Add

Tracking (Depends on: 1 bug, Blocks: 7 bugs, {dev-doc-needed, leave-open})

Firefox Tracking Flags (Not tracked)

Details

Attachments (11 attachments, 77 obsolete attachments)

Part 1.1: Add a GDB printer for BigInt values. sfink : **review+** Details | Diff | Splinter Review
a month ago Robin Templeton [:terpri]
5.66 KB, patch

Part 1.2: Add a BigInt type to the Rust bindings. Ms2ger : **review+** Details | Diff | Splinter Review
a month ago Robin Templeton [:terpri]
10.86 KB, patch

BigInt/JSON interaction

- The JSON experts say, JSON can't be extended
  - E.g., we can't add comments in JSON, so how could we add 123n?
  - Too much compatibility risk, since there are so many parsers
- Should JSON.parse parse numbers into BigInts sometimes?
  - No, this risks breaking existing code, which can trust that it's always a Number
- Should BigInt just go into JSON as a Number?
  - No, because then it wouldn't round-trip
- "Solution": (the best we can do)
  - By default, throws an exception
  - Could be overridden by defining BigInt.prototype.toJSON()
  - Or by passing in a replacer function

Stage 3

OF CONTENTS
duction
neric Types
difications to the Number grammar
tract Operations
MAScript Language: Expressions
Int Objects
dified algorithms
edArrays and DataViews
mics.wake(*typedArray*, *index*, *count*)
mics.store(*typedArray*, *index*, *value*)
TypedArray%.prototype.sort (*comparefn*)
pedArraySpeciesCreate (*exemplar*, *argumentLi*...
TypedArray%.prototype.fill (*value* [, *start* [, *en*...
TypedArray%.prototype.set (*array* [, *offset*])
TypedArray%.prototype.set(*typedArray* [, *offse*...
pedArray (*typedArray*)
tViewValue (*view*, *requestIndex*, *isLittleEndian*, ...
taView.prototype.getBigInt64 (*byteOffset* [, *litt*...
taView.prototype.getBigUint64 (*byteOffset* [, *li*...
taView.prototype.setBigInt64 (*byteOffset*, *valu*...
taView.prototype.setBigUint64 (*byteOffset*, *val*...
gents

Stage 3 Draft / April 24, 2018

BigInt

Introduction

This proposal adds arbitrary-precision integers to ECMA[illegible]r motivation and a high-level introduction, see the explainer document.

INTEGRATION PLANS

I attempted [illegible] his specification d[illegible]ye towards intelligibility and clarifying issues, rather than looking exa[illegible] final specifica[illegible]cular, the translation of the operations on the Number type into the new [illegible]mitted, as t[illegible]ical to the previous definitions and would be expressed only as a lengthy refa[illegible]kip wh[illegible]is text.

If you find any pa[illegible]unclear or confusing, please file an issue.

1 Numeric Types

ECMAScript has two built-in numeric types: Number and BigInt. In this specification, every numeric type *T* contains a multiplicative identity value denoted *T*::unit. The specification types also have the following abstract operations, likewise denoted *T*::*op* for a given operation with specification name *op*. Unless noted otherwise, argument and

Note: Color blocks (like ▇ or ▇) mean that a user may not be available. Tooltip shows the reason.

Issue 6791

Starred by 22 users

Status: Assigned
Owner: neis@chromium.org
Cc: jkummerow@chromium.org
▇ littledan@chromium.org
jarin@chromium.org
gsat...@chromium.org
▇ adamk@chromium.org
Components: Language
HW: ----
NextAction: ----
OS: ----
Priority: 2
Type: FeatureRequest

Harmony
merge-merged-6.5
merge-merged-6.7
merge-merged-3396

Blocked on:
issue 7712
issue 6957

Implement BigInt

Project Member Reported by adamk@chromium.org, Sep 6 2017

Back t

Currently a stage 3 proposal at TC39: https://tc39.github.io/proposal-bigint/

Showing comments 38 - 137 of 137 Older ›

Project Member Comment 38 by bugdroid1@chromium.org, Oct 13 2017

```
The following revision refers to this bug:
  https://chromium.googlesource.com/v8/v8.git/+/3872ed6543feb63015f9bae2a426a3d6d6d7c0c5

commit 3872ed6543feb63015f9bae2a426a3d6d6d7c0c5
Author: Adam Klein <adamk@chromium.org>
Date: Fri Oct 13 19:04:02 2017

[bigint] Support parsing of BigInt literals

Reuses the existing logic for BigInt.parseInt, adapted slightly
to allow octal and binary radix prefixes (and to support parsing
of a raw character buffer, rather than a v8::internal::String).

Bug: v8:6791
Change-Id: I41904b2204721eac452e0765fa9ff0ab26ee343b
Reviewed-on: https://chromium-review.googlesource.com/711334
Commit-Queue: Adam Klein <adamk@chromium.org>
Reviewed-by: Leszek Swirski <leszeks@chromium.org>
```

WebKit Bugzilla

Bug 179001: [ESNext] Implement support for BigInt

Home | New | Browse | Search | Search [?] | Reports | Requests | Help | New Account | Log In | Forgot Password

Bug 179001 - [ESNext] Implement support for BigInt

Status: NEW
Alias: None
Product: WebKit
Component: JavaScriptCore (show other bugs)
Version: WebKit Nightly Build
Hardware: Unspecified Unspecified
Importance: P2 Normal
Assignee: Nobody
URL:
Keywords:
Depends on: 179002 179003 179900 179901 179902 179903 181144 184327 185929 ~~175359~~ ~~179000~~ ~~179004~~ ~~179005~~ ~~179006~~ ~~182470~~ ~~183721~~ ~~183996~~ ~~184474~~ ~~185379~~
Blocks:

Show dependency tree / graph

Reported: 2017-10-30 03:59 PDT by Caio Lima
Modified: 2018-05-23 17:25 PDT (History)
CC List: 1 user (show)
See Also:

Attachments

Add an attachment (proposed patch, testcase, etc.)

Comparison bugfix

- BigInt <-> String comparison casts the String to a BigInt
- Previously, sometimes rounded as Number!

```
// Either way
0x1000000000000001n ==
    "0x1000000000000001"
            ⇒ true

// Before change
0x1000000000000001n <
    "0x1000000000000001"
            ⇒ true

// After change
0x1000000000000001n <
    "0x1000000000000001"
            ⇒ false
```

BigInt

On this Page

Description
Constructor
Static methods
Instance methods
Usage recommendations
Examples
Specifications
Browser compatibility
See also

Related Topics

`BigInt` is a built-in object that provides a way to represent whole numbers larger than 2^{53} - 1, which is the largest number JavaScript can reliably represent with the `Number` primitive. **`BigInt`** can be used for arbitrarily large integers.

Description

A `BigInt` is created by appending `n` to the end of an integer literal — `10n` — or by calling the function `BigInt()`.

```
const theBiggestInt = 9007199254740991n;

```

BigInt and Number type mismatch tests for infix operators #1394

Edit

Merged rwaldron merged 1 commit into `tc39:master` from `thejoshwolfe:bigint-and-number` on Jan 25

33 test/language/expressions/addition/bigint-and-number.js

View

```
... @@ -0,0 +1,33 @@
+// Copyright (C) 2018 Igalia, S.L. All rights reserved.
+// This code is governed by the BSD license found in the LICENSE file.
+/*---
+esid: sec-addition-operator-plus-runtime-semantics-evaluation
+description: Mixing BigInt and Number produces a TypeError for addition operator
+features: [BigInt]
+info: |
+  Let lprim be ? ToPrimitive(lval).
+  Let rprim be ? ToPrimitive(rval).
+  ...
+  Let lnum be ? ToNumeric(lprim)
+  Let rnum be ? ToNumeric(rprim)
+  If Type(lnum) does not equal Type(rnum), throw a TypeError exception.
+---*/
+
+assert.throws(TypeError, function() { 1n + 1; }, "1n + 1 throws TypeError");
+assert.throws(TypeError, function() { 1 + 1n; }, "1 + 1n throws TypeError");
+assert.throws(TypeError, function() { Object(1n) + 1; }, "Object(1n) + 1 throws TypeError");
+assert.throws(TypeError, function() { 1 + Object(1n); }, "1 + Object(1n) throws TypeError");
+assert.throws(TypeError, function() { 1n + Object(1); }, "1n + Object(1) throws TypeError");
```

BigInt on Tiny Hardware with XS

January 29, 2019 | Peter Hoddie, Principal

The JavaScript language is preparing to add built-in support for integers of arbitrary size with a new feature named `BigInt`. The built-in `Number` type is accurate to 53-bits of precision, which is insufficient for many tasks. By standardizing `BigInt` into the language, developers benefit from having a single API available in all environments and JavaScript engines can more easily optimize `BigInt` performance.

Moddable's XS JavaScript engine is focused on providing full JavaScript support on the smallest of devices, where memory and performance are limited. At first, `BigInt` might seem

Search ... Search

Recent Posts

Scripts & Things Recap: November 2019

XS Update

Faster Flash

Firmata Remote Device Control in the Moddable SDK

XS: Secure, Private JavaScript for Embedded IoT

Tweets by @moddabletech

Moddable Retweeted

STEFFIE ステフィー
@steffieharner
@JavaScript is stepping into the embedded universe

it was awesome hearing petter hoddie @phoddie speak on the potential of javascript in the world of #IoT w/ @moddabletech at @codechrysalis today!

moddable.con

Host environment integrations

- Node Core APIs using BigInt by Joyee Cheung
- HTML and WebIDL integration by Ms2ger and Anne van Kesteren

Stage 4

BigInt to Stage 4?

- Full test262 tests ✔
- Specification PR ✔
- 2+ implementations ✔
- ES2020!

Some late fixes

- Clarifying 0123n is a syntax error -- no legacy octal BigInt literals
- Various cross-references around the specification

Last-second grammar bug fix

Fixes for integrating into the main spec

Many great contributors; sadly insufficient recognition

人不知而不慍、不亦君子乎。

If one remains not annoyed when he is not understood by people around him, isn't he a sage?

My suggestions for contributing to TC39 at different stages

Stage 0/1

- Document use cases
- Begin seeking feedback, and keep doing it the whole time!
- Discuss the problem space/big-picture questions
- Prototype implementations:
  unstable, rough, maybe downstream

Stage 2

- Nail down the proposal details, from discussion and prototyping
- Draft documentation and tests in the proposal repo
- Prototype implementations:
  Ideally nearing completion, but still considered unstable
- Distribute the prototype more broadly to gather more solid feedback

Stage 3

- Upstream tests into test262 and fill in any gaps.
- Place documentation in MDN and fill in the gaps
- Implement in engines and consider shipping
- Integrate usage into environments (e.g., the Web, Node.js)
- Distribute high-level explanations more broadly to developers

Stage 4

- Ideally nothing!
- Tie up any loose ends in the spec language
- Implement in trailing engines if needed
- Further changes: separate PRs/proposals

TC39 changes over time

Older TC39 mode

- Specification: Big MS Word doc
- Communication: Meetings and es-discuss list
- Large, occasional specification; no stages

 © Ecma International 2015

ECMAScript 2015 Language Specification

1 Scope

This Standard defines the ECMAScript 2015 general purpose programming language.

2 Conformance

A conforming implementation of ECMAScript must provide and support all the types, values, objects, properties, functions, and program syntax and semantics described in this specification.

A conforming implementation of ECMAScript must interpret source text input in conformance with the Unicode Standard, Version 5.1.0 or later and ISO/IEC 10646. If the adopted ISO/IEC 10646-1 subset is not otherwise specified, it is presumed to be the Unicode set, collection 10646.

A conforming implementation of ECMAScript that provides an application programming interface that supports programs that need to adapt to the linguistic and cultural conventions used by different human languages and

Search...

TABLE OF CONTENTS

Introduction
1 Scope
2 Conformance
3 Normative References
4 Overview
5 Notational Conventions
6 ECMAScript Data Types and Values
7 Abstract Operations
8 Executable Code and Execution Contexts
9 Ordinary and Exotic Objects Behaviours
10 ECMAScript Language: Source Code
11 ECMAScript Language: Lexical Grammar
12 ECMAScript Language: Expressions
13 ECMAScript Language: Statements and Declarati...
14 ECMAScript Language: Functions and Classes
15 ECMAScript Language: Scripts and Modules
16 Error Handling and Language Extensions
17 ECMAScript Standard Built-in Objects
18 The Global Object
19 Fundamental Objects
20 Numbers and Dates
21 Text Processing
22 Indexed Collections
23 Keyed Collections
24 Structured Data

Draft ECMA-262 / April 23, 2018

ECMAScript® 2019 Language Specification

ecma INTERNATIONAL

Contributing to this Specification

This specification is developed on GitHub with the help of the ECMAScript community. There are a number of ways to contribute to the development of this specification:

GitHub Repository: https://github.com/tc39/ecma262
Issues: All Issues, File a New Issue
Pull Requests: All Pull Requests, Create a New Pull Request
Test Suite: Test262
Editor: Brian Terlson (E-mail, Twitter, GitHub)

Community:

- Mailing list: es-discuss
- IRC: #tc39 on freenode

This repository Search Pull requests Issues Marketplace Explore

tc39 / ecma262

Unwatch 836 Star 6,408 Fork 446

Code Issues 157 Pull requests 52 Projects 0 Wiki Insights Settings

Status, process, and documents for ECMA262 https://tc39.github.io/ecma262/

Edit

ecmascript javascript Manage topics

1,191 commits 10 branches 10 releases 85 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

jmdyck and bterlson Editorial: Misc Editorial (#1182) ... Latest commit cbe4679 20 hours ago

img	Editorial: Store and rethrow module instantiation/evaluation errors	9 months ago
scripts	Meta: Build ES2016 in a subfolder.	2 years ago
workingdocs	Editorial: Use HTTPS for ecma-international.org URLs (#1017)	6 months ago
.editorconfig	Add .editorconfig	3 years ago
.gitignore	Editorial: reword InstanceofOperator copy and rename vars (closes #894)	11 months ago
.travis.yml	meta: require node 8 to build the spec	4 months ago
CODE_OF_CONDUCT.md	Add CoC.md reference file (#1168)	13 days ago
CONTRIBUTING.md	Meta: Add paragraph about resolving conflicts with master. (#1099)	2 months ago
FAQ.md	Fix example file name in FAQ.	3 years ago
README.md	meta: Add Community section to readme.md	a year ago
figure-2.uxf	Editorial: fix size of figure 2.	2 years ago

Invited experts

Ecma Rules (June 2019)

6.2.6
Individuals can participate in the work of a TC as invited experts. They participate only at the invitation of the SG at the request of the TC. They have no voting rights. Invited experts shall comply with the Ecma policies and sign the appropriate form before participating. The invitation to participate may be withdrawn by the SG at any time.

js-outreach / **js-outreach-groups**

Unwatch 26 | Star 20 | Fork 3

Code | Issues 6 | Pull requests 0 | Actions | Projects 0 | Wiki | Security | Insights | Settings

No description, website, or topics provided. Edit

Manage topics

56 commits | **2** branches | **0** packages | **0** releases | **3** contributors

Branch: **master** | New pull request | Create new file | Upload files | Find file | Clone or download

bkardell Create notes-2019-11-21.md ... Latest commit `c094b13` 23 hours ago

educators	Create notes-2019-11-14.md	17 days ago
frameworks	Create notes-2019-11-21.md	23 hours ago
tools	Create notes-2019-11-26.md	23 hours ago
CODE_OF_CONDUCT.md	Create CODE_OF_CONDUCT.md	11 months ago
README.md	Add Tools and transpilers group to existing groups list	3 months ago

README.md

JS outreach groups

Some TC39 delegates are organizing meetings with various stakeholders in order to get more input in the process and share information with the community. Current calls include:

- Educators
- Frameworks
- Tools and transpilers

all categories all tags Categories Latest Top

Category	Topics
Announcements Welcome to TC39's focal communication hub.	0
Proposals Discuss existing ECMAScript proposals.	2
Ideas New ideas, proposals looking for a champion, and proposal mentoring. Get feedback on your new ideas here: what's been done, where to find prior art, the polish your proposal needs to be considered by the committee.	27
I have questions Specifying a language is a complex affair, filled with confusing language and sometimes bewildering processes. It's okay not to understand, and we encourage you to ask questions! Spec Reading Delegates AMA	9

Top

What should the categories be? Meta	18 Jun 7
bigint enhancements. Proposals proposal	11 Oct 15
Try-catch oneliner Ideas proposal	14 14d
Force HTTPS usage Meta	9 Apr 5
New well-known Symbol: Symbol.typeofTag Ideas proposal	12 17d
JSON: Add Datetime, Timedelta and Binary Data Types	9 9d

Introduction

Like the technical community as a whole, TC39 is made up of a mixture of professionals and volunteers from all over the world. To ensure a fair and balanced standards process, to avoid communication issues and unhappiness, and to promote inclusiveness, we have a few ground rules that we ask people to adhere to.

This isn't an exhaustive list of things that you can't do. Rather, take it in the spirit in which it's intended a guide to make it easier to enrich all of us and the technical communities in which we participate and empower others to speak.

This Code of Conduct is enforced within all spaces managed by TC39. This includes IRC channels moderated by TC39, mailing lists such as esdiscuss, issue trackers on projects hosted by TC39, and TC39 events and meetings.

If you believe someone is violating the Code of Conduct, we ask that you report it by emailing tc39-conduct-reports@googlegroups.com. For more details, please see our Reporting Guidelines.

Be respectful

Respect is a fundamental value of the standardization work. Not all of us will agree all the time, but disagreement is no excuse for poor behavior and poor manners. We might all

README.md

Ecma International's TC39 is the standards committee which defines ECMAScript (JavaScript). This repository is intended to provide documentation into how TC39 works.

This repository is an early work in progress!

For an introduction to getting involved in TC39, see CONTRIBUTING.md.

Table of Contents

- Meta
  - Introduction and Table of Contents for this repo (this file)
  - List of suggested additions for this repo
- Proposals
  - Championing a proposal at TC39
  - How to write a good explainer
  - How to make a Pull Request against the ECMAScript specification

TC39

TC39 | Specs | Contribute | State of Proposals

Specifying JavaScript.

TC39

Ecma International's TC39 is a group of JavaScript developers, implementers, academics, and more, collaborating with the community to maintain and evolve the definition of JavaScript.

We are part of ecma INTERNATIONAL

Contribute

TC39 welcomes contributions. You can help by giving feedback on proposals, improving documentation, writing tests or implementations, or suggesting language feature ideas. See our contributor guide for details.

To participate in TC39 meetings as a member, join Ecma.

Specs

We develop the JavaScript (formally, ECMAScript) specification on GitHub and meet every two months to discuss proposals. To learn more about the process, please take a look at the four stages for new language feature proposals. See our meeting agendas and minutes to learn more.

State of Proposals

Current Candidates for the Specification

JavaScript and evidence-based language design

By Yulia Startsev

Posted on May 29, 2019 in Featured Article, JavaScript, and Standards

Share This

Author's note: Hi, I'm an engineer at Mozilla working on the Firefox DevTools server. I'm also a TC39 representative. This post focuses on some of the experiments I am trying out at the TC39, the standards body that manages the JavaScript specification. A follow up post will follow...

Integrating the traditional and the new values

- First-principles reasoning
- "The future is bigger than the past"
- Long, complete design cycles
- Rigorous debate
- Welcoming participants who join
- Language specialists/theorists

- Practicality
- Integration into the existing ecosystem
- Quick, incremental iteration
- Letting points be made without interruption
- Actively seeking out feedback
- More JS/frontend dev participation

All of these are useful,
complementary perspectives

溫故而知新、可以爲師矣。

If a man keeps cherishing his old knowledge,
so as continually to be acquiring new,
he may be a teacher of others."

谢谢!

https://tc39.es

Add me on WeChat

Talk feedback, Q&A

igalia

@littledan