BigInt: Integers as big as you want in JavaScript

Daniel Ehrenberg
Igalia
In partnership with Bloomberg

Web Engines Hackfest 2017

Why add something?

- Represent bigger Integers:
- inode numbers
- Microseconds since the Unix epoch
- Hashes/checksums
- FFIs to languages with larger integer types
- Basis for users implementing a Decimal type
- Manipulate binary data containing 64-bit ints
- Project Euler...

What would make sense in JS?

- Sadness: `1` is `1.0`
  - Resolution: Use 1n for integers
- Sadness: `12374699872164983276428373n + 1` = ?
  - Resolution: Throw a `TypeError`
- Sadness: Entire ecosystem is based on Numbers
  - Resolution: Throw more `TypeErrors`
- Sadness: `Number` means float, so we can’t call it `BigNum`
  - Resolution: `BigInt`!

Add BigInt or Int64?

- For Int64
    - Most programmer requests fit in this range (or Uint64)
    - Programmers assume Int64 will be faster
- For BigInt
    - Overflowing is usually a bug
    - High-level dynamic languages usually opt for BigInt
    - If we don’t do BigInt, need multiple size types
    - BigInt.asIntN/asUintN provides overflow
    - Implementers say BigInt should be fast
- Resolution: BigInt

Enter TC39

- JavaScript standards committee
- Meets every two months
- Representatives from
  - Browser vendors
  - JavaScript programmers
  - Framework/library authors
  - Language experts
  - Node.js
- Current draft spec at https://tc39.github.io/ecma262/

History of Int64/BigInt in TC39

- ES1 has only Number--double float
- Integer types in Waldemar Horwat's ES2 proposal (1999)
- ES2, ES3 are editorial/library changes
- Proposed for ES4 (~2004-2008) -- abandoned
- ES5, ES5.1 is intentionally minimal
- ES6 proposed “value types” -- deferred
- November 2016, Brendan Eich proposes Int64/Uint64

TC39 Stage process

- Stage 1: An idea in a GitHub repo
- Stage 2: Committee supports initial draft
- Stage 3: Solid draft resolving all committee concerns
- Stage 4: Two implementations and conformance tests; joins the spec working draft

- In the background: Annually, annual official versions declared, for “reasons”

History of Int64/BigInt in TC39

- November 2016, Brendan Eich proposes Int64/Uint64
    - Stage 1!
- V8 team proposes BigInt rather than Int64
- January 2017, agreement on change to BigInt
- March 2017, BigInt to Stage 2
- May 2017, gradually work out more spec issues
- July 2017, BigInt to Stage 3
- Draft BigInt spec

Meanwhile, in code...

- Implementations in progress in
    - SpiderMonkey
    - V8
    - Babel polyfill
- Conformance tests (test262) in progress
- Other browsers expressed interest
- My prediction: Usable in multiple browsers next year

Questions?

- TC39 heckling welcome