SmallJS

Back to elegance

by: Richard Ronteltap E JS



Website: small-js.org



About the author

- My name is Richard Ronteltap (Yes, that is a bit of a strange last name :)
- Live in Amsterdam, The Netherlands
- Studied Computer Science at the University of Amsterdam where I met my wife (she insisted I say this :), where I work as an IT department head.
- My CS thesis in '93 was on OO databases, with a query optimizer written in Smalltalk-80.
- I also discovered: Smalltalk/V, Dolphin, Pharo, Amber...











What is SmallJS?

- Transpiler from Smalltalk to efficient, lightweight JavaScript.
- JS code runs in modern browsers and in Node.js.
- Smalltalk-80 language syntax support
 - Class and method names left like familiar JS.
- Source file based (not image based).
- Development in Visual Studio Code IDE
 - With syntax coloring and step debugging!
- Built-in unit-testing and GUI-testing (browser).
- Playground available (local or on small-js.org).
- Completely free and open source.



- Develop modern apps in ST, specifically web apps, using the same language for the front-end and the back-end.
- Promote the elegant Smalltalk language specifically to JS, TS, PHP, and Python devs, who don't know what they're missing...:)
- Looked at: Amber Smalltalk, PharoJS, Seaside But though it could be more JS native and integrated. With a few trade-offs.
- And a separate IDE and a modular image don't seem so bad...



How does it look?

×	$\equiv \qquad \leftarrow \rightarrow$	P ShopClient (Workspace) □ □ □ □ □ □ □ □
Сh	EXPLORER ···	□ ShopOrder.st ×
-	〜 Shopalient (Worksp 🖣 日井 ひ 白	📁 Client > src > Smalltalk > 🗋 ShopOrder.st 💠 🙂 🔲 🔆 🏌 🎝 🔲
Q	🗸 🚅 Smalltalk	loadOrders
	> 📫 Browser	ordersApi
مړ	> 📫 Core	ordersAni :- Window default location hostDath '/ani/orders'
0	> 🖬 Node	Object
2	✓ Client	fetchObject: ordersApi
201	> 📫 .vscode	then: [:object self onLoadOrders: object]
<u>д</u> 0	V is src	onError: [:error self onLoadOrdersError: error] .
ليل		onloadOrders: object
		orders := (object atProperty: 'orders') map: [:object Order
-10	D ShopLogin.st	<pre>fromObject: object].</pre>
	ShopOrder.st	<pre>products := (object atProperty: 'products') map: [:object Decduct fromObject: object]</pre>
Ψ.	ShopProduct.st	Product fromobject: object j.
	> 👷 Tools	PROBLEMS OUTPUT DEBUG CONSOLE ··· Filter (e.g. text, lexclude) 🚍 ^ ×
	TS App.ts	
	> 📫 web	
	ShopClient.code-workspace	
Q	Start static web server.cmd	
~	T& tsconfig.json	
503	> OUTLINE	
~	> TIMELINE	>
*	⊗ 0 🛆 0 🏾 🎗 1 🔥 Launch Edge (Client) 👩	Live Share Ln 35, Col 2 Tab Size: 4 UTF-8 CRLF SmallUS 🖗 🕻





Playground (live demo possible)

Playground

Help

Enter any Smalltalk expression or statements below and then press [>Evaluate]. For more help on the syntax of SmallJS, look at <u>Smalltalk.md</u>

Some examples:

'Hello, ', 'World', \$! toString 1 + 2 < 4 ifTrue: ['smaller'] #(1 2 3) map: [:n | n squared] | a | a := 4. a squared.

=

Expression

#(1 2 3) map: [:n | n squared]

▷ Evaluate

Result

#(149)

Note:

It's not an IDE in your browser, because VSCode is much nicer and compiling is lightning quick :)



SmallJS library - Core

- Basic ST classes you'd expect.
- Full ST magnitude (number) hierarchy.
 - With also: BigInt, Fraction, Date, Point(3D), Rect.
- Events, streams, fetch.
- Async & await.
- Unit test framework
 - That auto-finds new test classes and methods.



SmallJS library - Browser

- HTML base classes like: Document, Window, Location.
- All common HTML elements.
- CSS styles and rules.
- Full Canvas 2D support for drawing.
- Lightweight HTML component system to easily create SPAs.
- ✓ Desktop app support using Electron or NodeGui !
- Example apps to get you started quickly.



SmallJS library - Node.js

- ✓ HTTP server, ExpressJS.
- Worker threads, File.
- ✓ Databases: SQLite, Postgres, MariaDB, MySQL.
 - \checkmark With an ORM for easy CRUD operations on objects.
- AI: Ollama, OpenAI, Deepseek, Google AI, Anthropic.
 - With shared classes for browser clients.
- Example apps to get you started quickly.



Why use SmallJS?

- The Smalltalk language syntax fits on a postcard
- Objects all the way down
 - Customizable on every level.
 Need to add a complex number type? Easy.
- Well defined behaviors, compared to JS.
 - Integers are really integers. Controlled type conversions.
- Uses familiar JS names and functionality
- Seamless integration of ST with JS/TS and HTML/CSS.

🔊 📲 Smalltalk's syntax fits on a postcard

POST CARD

HERE UNITED STATES AND CANADA ONE CENT. FOREIGN, TWO CENTS

PLACE STAMP

exampleWithNumber: x

"A method that illustrates every part of Smalltalk method syntax except primitives. It has unary, binary and keyword messages, declares arguments and temporaries (but not block temporaries), accesses a global variable (but not an instance variable), uses literals (array, character, symbol, string, integer, float), uses the pseudo-variables true, false, nil, self and super, and has sequence, assignment, return and cascade. It has both zero-argument and one-argument blocks. It doesn't do anything useful, though."

PHOSTINT

TRADE MARK - REG U S PAT OFF MADE ONLY BY DETROIT PUBLISHING CO

y |
true & false not & (nil isNil) ifFalse: [self halt].
y := self size + super size.
#(\$a #a 'a' 1 1.0)
do: [:each | Transcript
show: (each class name);
show: (each printString);
show: '].
^ x < y</pre>

Complete Syntax on a Post Card

for Lovers of Programming everywhere

THIS SPACE MAY BE USED FOR MESSAGE.

THIS SPACE FOR THE ADDRESS.



Code example ST vs JS

SmallJS

JavaScript

Code:					
<pre>squared ^ self * self.</pre>					
Call with > result:					
10 squared	> 100				
1.5 squared	> 2.25				
(1 / 3) squared	> (1 / 9)				
99999999 squared	> right answer, BigInt				
's' squared	> error, stops				

> Number is a base class for Integer, Float, Fraction and BigInt, but not String.

NumUtil.squared	(n)

Code:

{ return n * n; }

Call with > result:

NumUtil.squared(10)	> 100
NumUtil.squared(1.5)	> 2.25
NumUtil.squared(1 / 3)	> 0.111111111111
NumUtil.squared(99999999)	> wrong answer, float
NumUtil.squared('s')	> NaN, continues

> No (safe) integers, no fractions, BigInt not integrated, error prone type coercion.

But what about new JS/TS features?

JS feature	ST solution
functional programming	Don't use state vars. Use array iterators for list processing.
interface	Abstract base class
record, tuple, immutable	Class with only getters
decorator	Pass anonymous function (block)
static	Class method
private	No getter method
variable argument list	Array
optional argument	Extra 1-line method
import / export	Automatic
async / await	Implemented, unfortunately.
type checking	Add type hints like Python? (todo)

The philosophy is that retaining simplicity is worth some extra lines of encapsulated code.

(E.g.: Pharo, Dolphin, Cincom, Squeak)

File-based (not image-based)

- Easy source control in clear hierarchy. IDE safely separate from code.
- Modular class loading iso unsafe image stripping.
- Can use rich and familiar IDE (VSCode).
- Focus on your *project* files, that are separated from the library (image).

Run anywhere

- One language for front-end and back-end apps in all browsers and Node.js. So also runs on mobile devices.
- Integrates smoothly with the rich JS ecosystem
 - Typically 1 line of interfacing code per encapsulated JS method.
 - ST can call JS and vice versa.
 - Newly available JS features and libraries can be integrated quickly.



SmallJS / ST trade-offs

- ST is dynamically typed
 - An IDE enhancement cloud help here.
 - And optional typing like TS is an idea...
- Debugging integrated with JS
 - Would making a VSCode Language Server be worth it?
 - Now you can easily see what's really going on..
- Tiny community, currently...:-)





SmallJS summary

- Elegant and safe Smalltalk language!
- Integrates tightly with JS in browser and Node.js.
- Familiar JS class and method names in both.
- Use it with your favorite IDE.
- Incremental use possible, mix and match with JS/TS.



Examples - basic

With live demo's & code inspection

Арр	Notes
Counter	Standard minimal example web app.
Todo	Todo list web app with CRUD operations, sorting and multi-language support.
Balls	Colorful bouncing balls, using the HTML Canvas 2D API, which is fully supported.
Shop	Basic web shop front-ends, traditional and as SPA and back-end, with database support on SQL ORM mapper.



Examples - advanced With live demo's & code inspection

Notes
Number crunching in SmallJS for primes, Fibonacci, Pi.
True multithreading using the Web Workers API. (Node 'worker threads' are also supported)
Platform independent desktop apps, when you don't care about memory and complexity :)
Platform independent desktops apps, when you <i>do</i> care about memory usage and simplicity.
SmallJS web client with Pharo web server using Zinc, so quite standard on both sides.

