

ESUG 2022, Нови Сад (Novi Sad), Србија (Serbia)



for

# Real World Applications

Noury Bouraqadi & Dave Mason



# New to Smalltalk ?

# **Smalltalk is dangerous. It is a drug.**

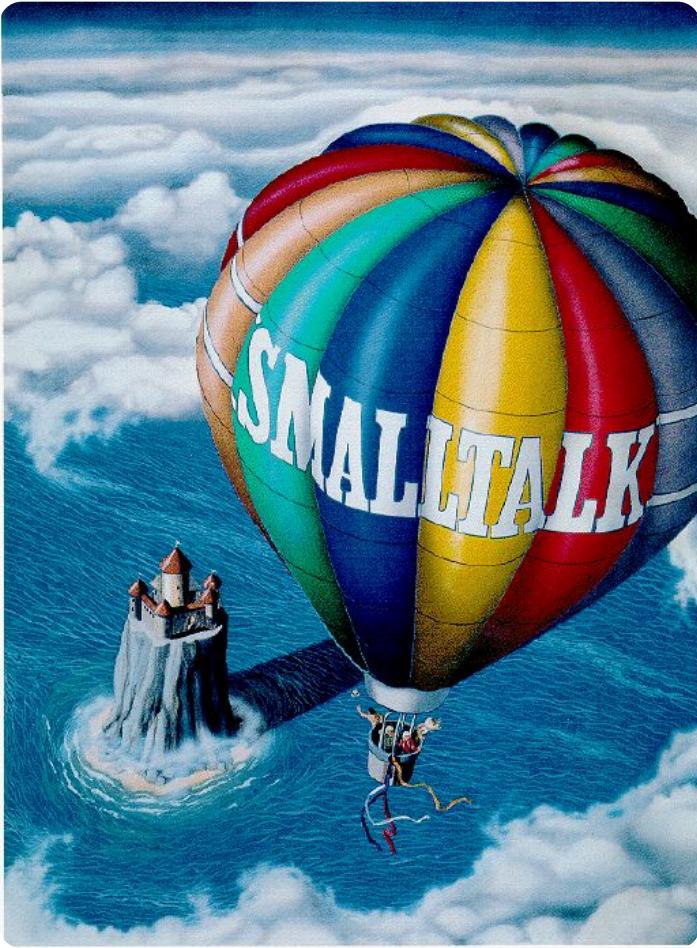
My advice to you would be  
**don't try it.**

It could ruin your life



– Andy Bower, CEO of Object Arts Ltd.

Great  
Language,  
Libraries,  
Tools,  
Community



We want to develop in

**Smalltalk**

All the Time

Everywhere

# What to do with non-Smalltalk Resources?

**HTML**



**CSS**



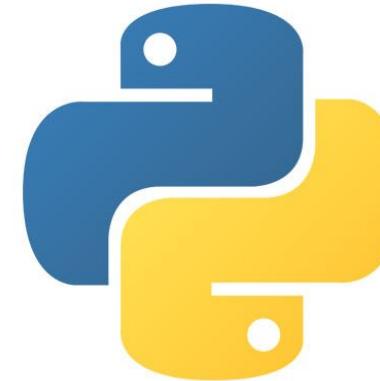
**JS**



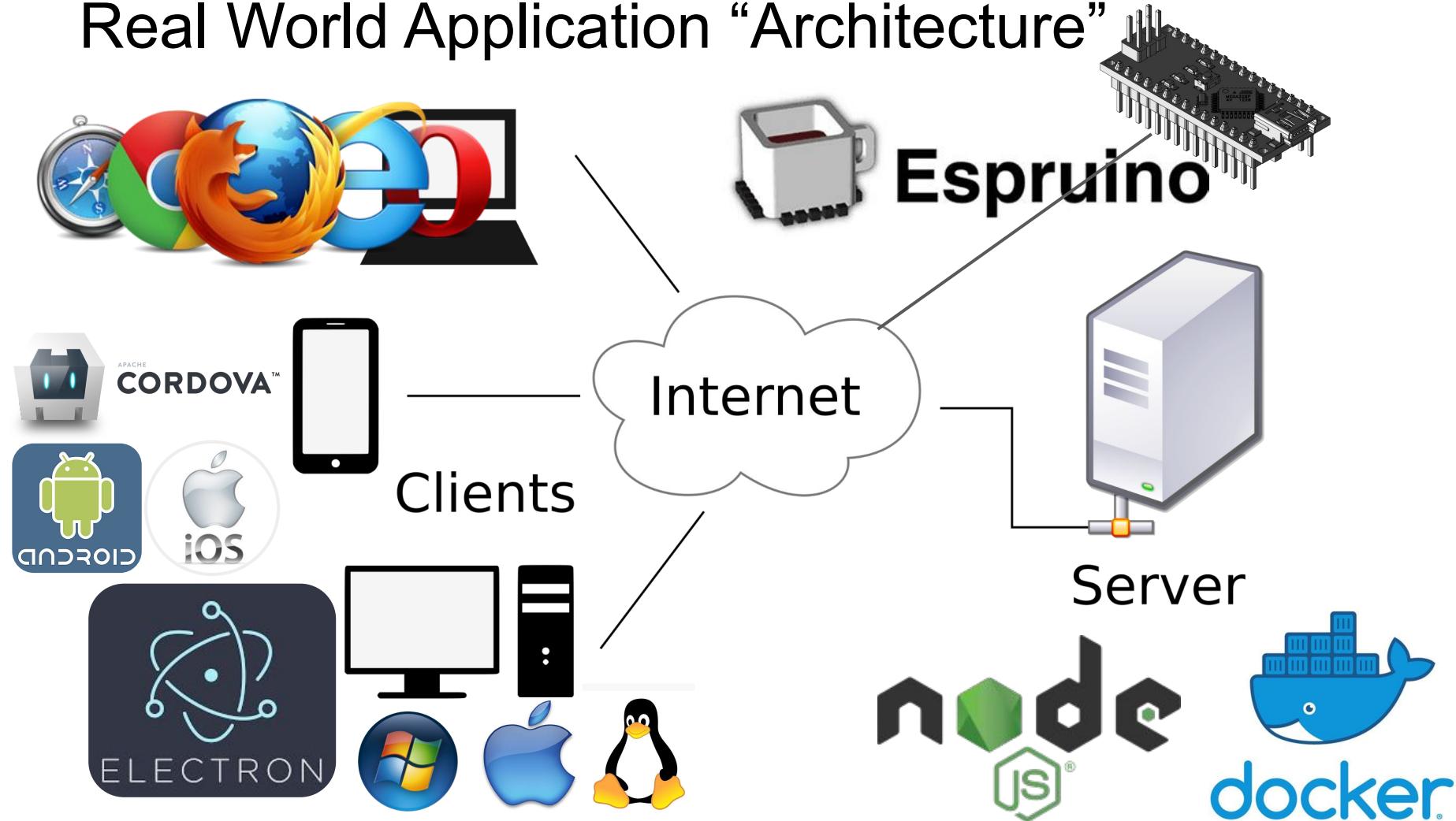
**PHP**



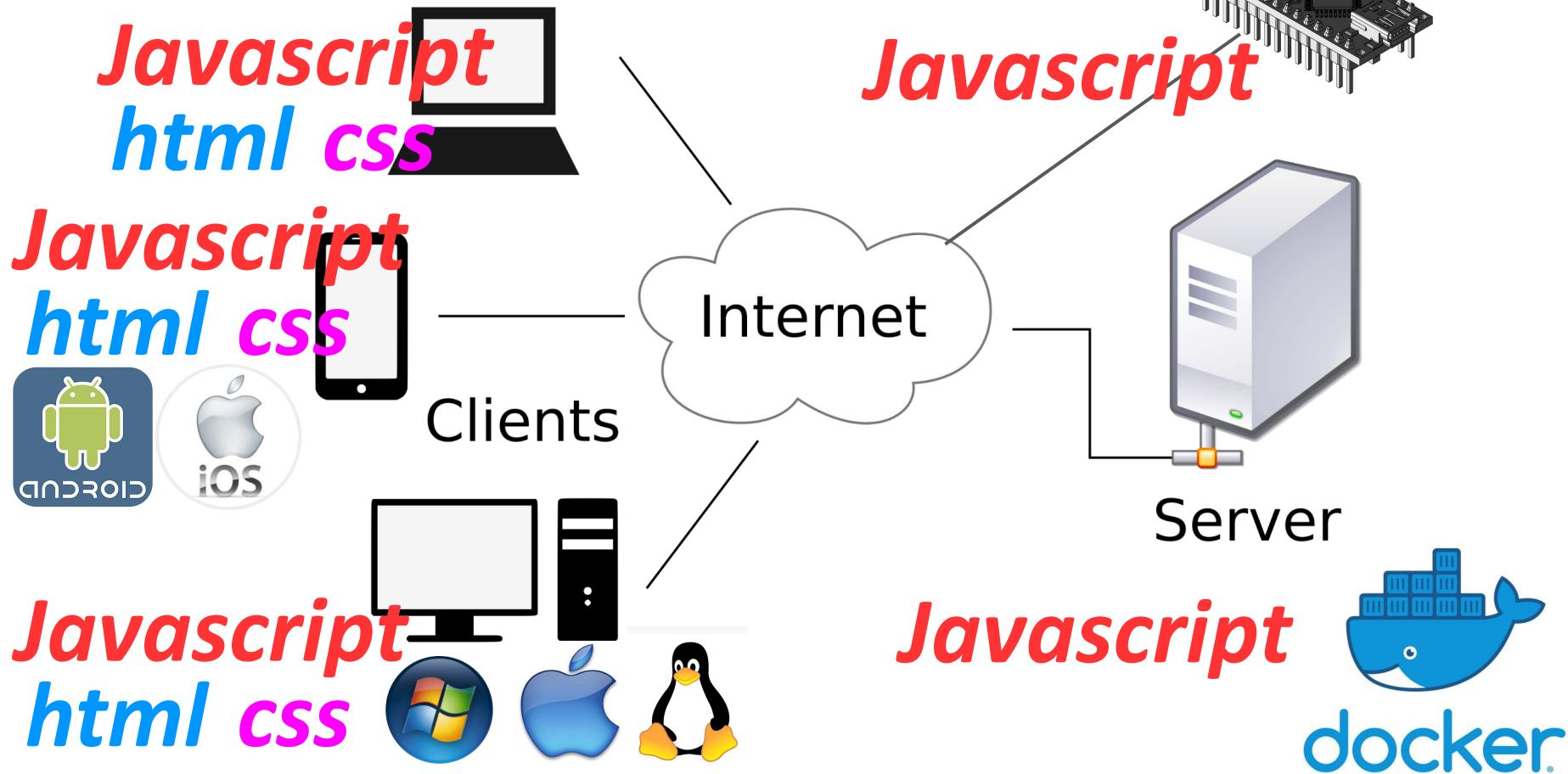
**SQL**



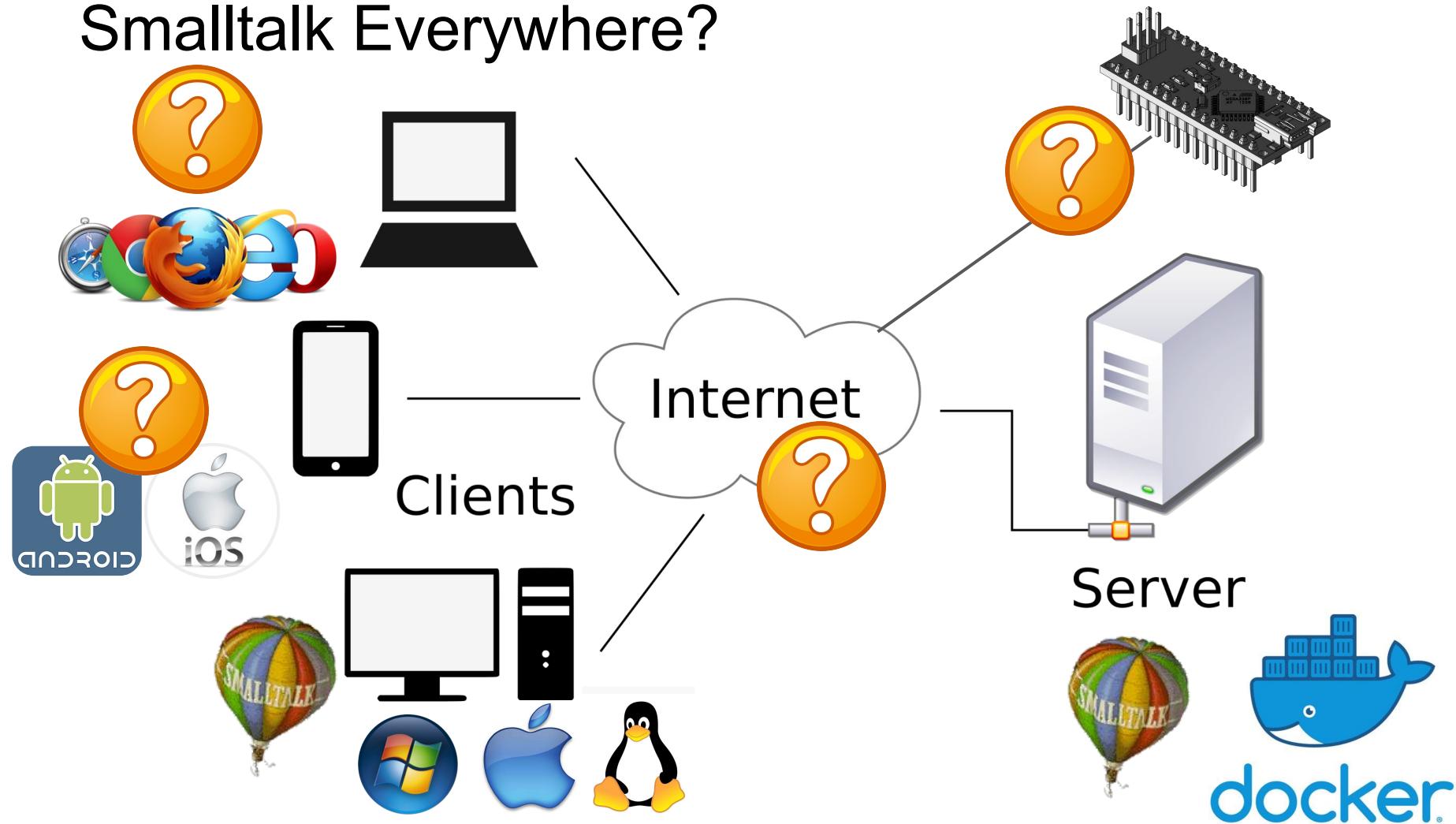
# Real World Application “Architecture”



# Real World Application “Architecture”



# Smalltalk Everywhere?





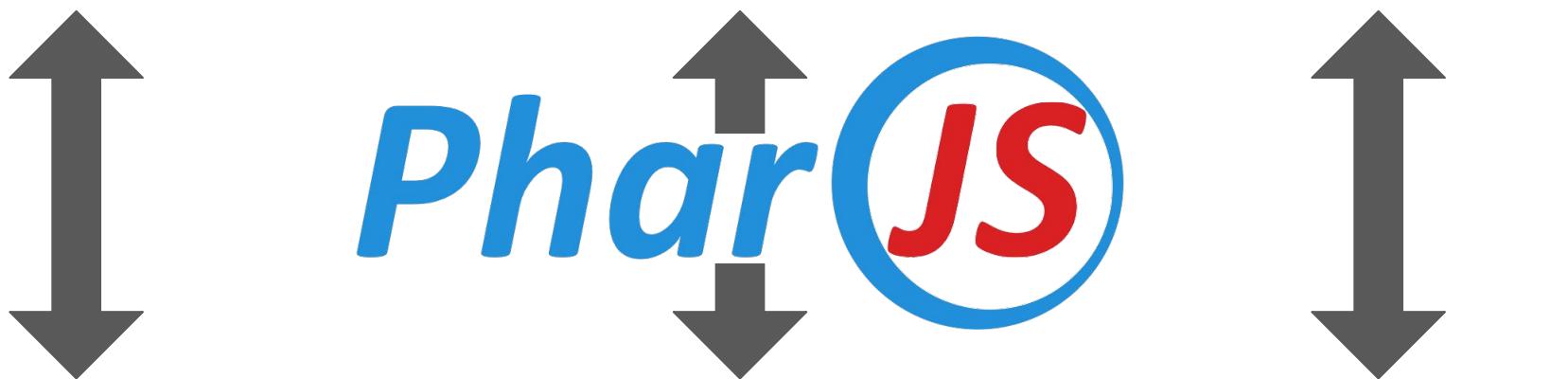
The screenshot shows the Pharo IDE interface. In the top left, the Pharo logo is displayed. The top bar includes tabs for "Some help tools on Pharo.", "Playground", and "Inspector on a Set [10 items] (23 46 8 33 99 73 27 96 64 4)". The left sidebar lists packages like "Random-Core", "Random-Tests", and "Refactoring-Changes". The main workspace contains the following code:

```
1 |numbers|  
2 numbers := Set new.  
3 10 timesRepeat: [ numbers add: 100 atRandom ].  
4 numbers inspect.
```

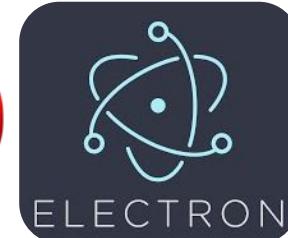
A stack trace window is open below the workspace, showing:

Class	Method	Package
UndefinedObject	Dolt	-
CompiledMethod	valueWithReceiver:arguments:	Kernel

The right side of the interface shows an "Inspector" window for a set of 10 items, with tabs for "Items", "Raw", "Breakpoints", and "Meta". The "Value" tab displays the contents of the set.



*Javascript*





- Develop in Pharo Smalltalk all the time!



**Espruino**

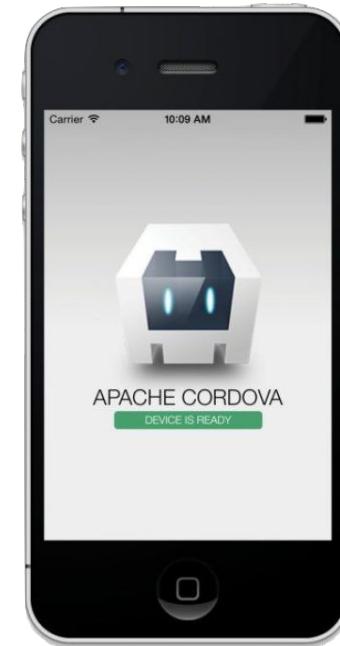
- Reuse existing JS libraries



- JS Portability



- JS Run-time Speed





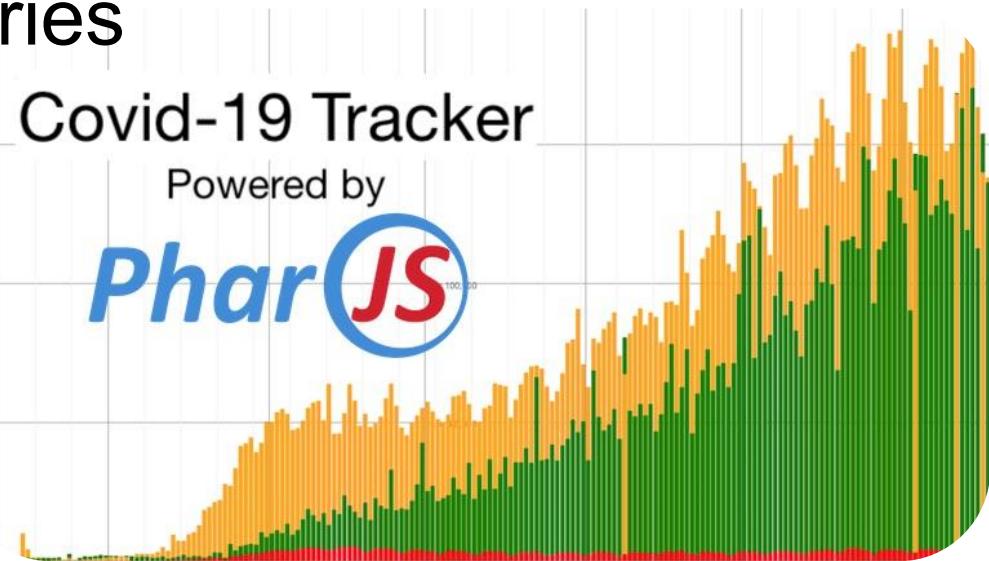
- **Transpiler:** Converts Pharo Code to JavaScript
- **Framework:** Develop JS applications in Pharo
- **Libraries:** Extend JS Objects with Pharo's Behavior
- **Tools:** Playground + Inspector for JS Objects
- **Test Framework:** Test JS Code

# PharJS Success Stories



Covid-19 Tracker

Powered by

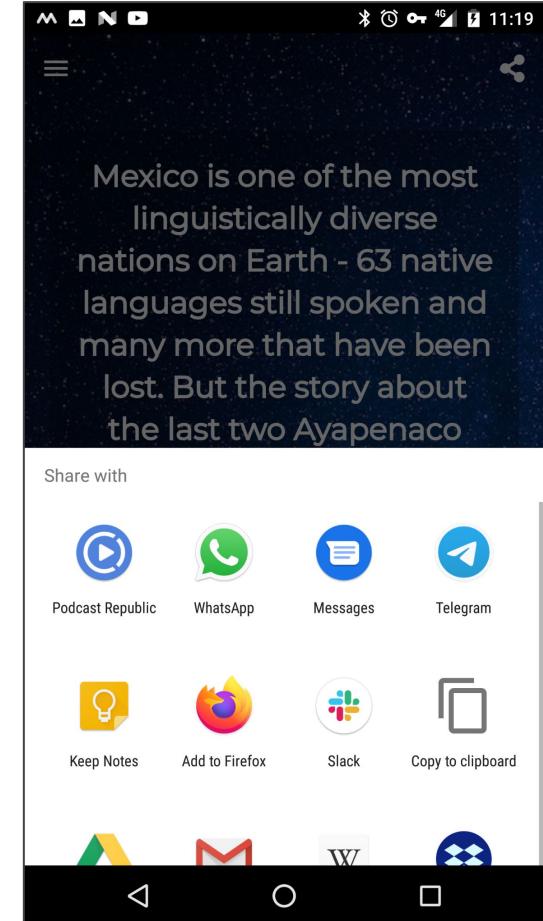
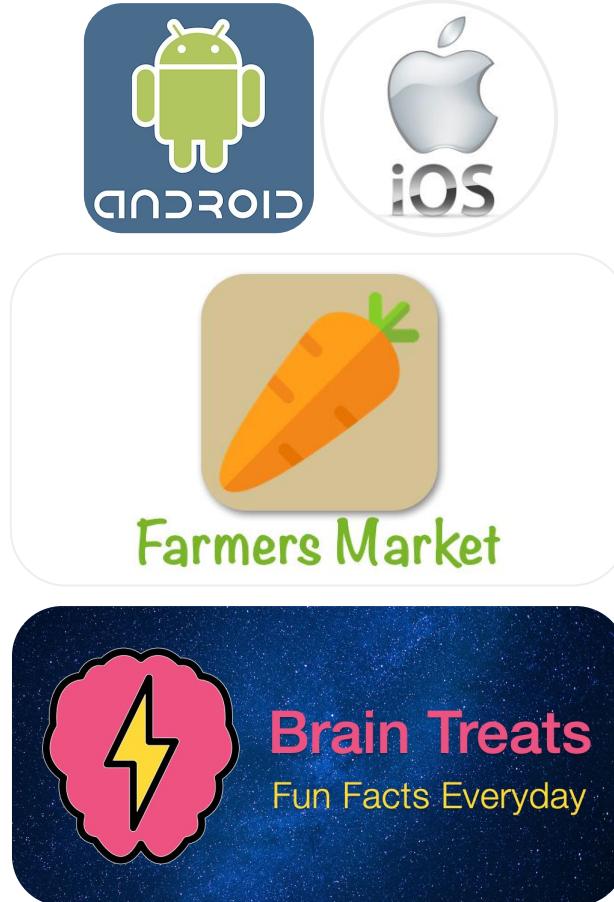
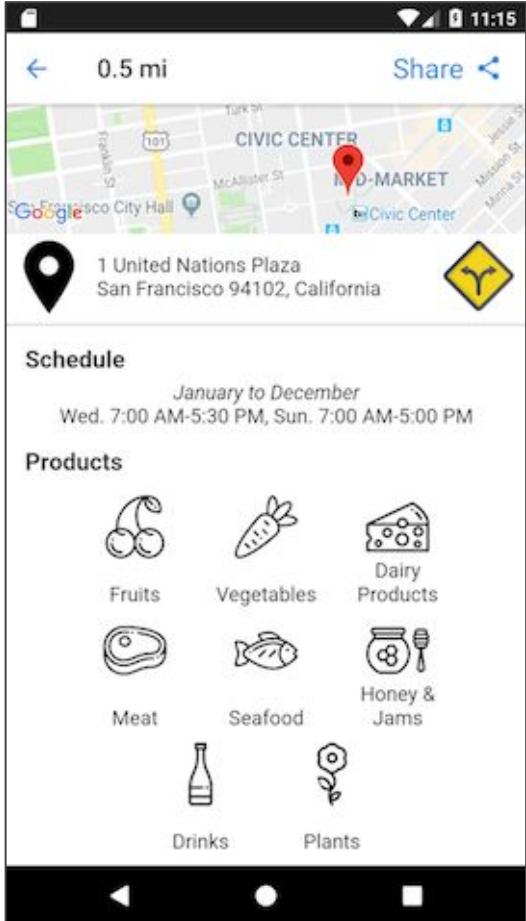


*Smalltalk REPL*

PLC3000

Teaching PLC Automation  
Made Easy

# PharJS Success Stories

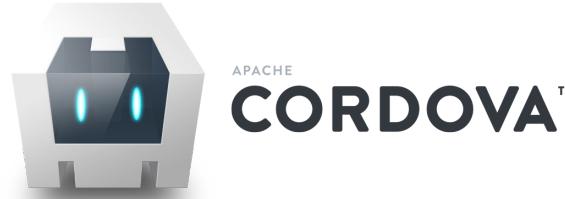


*Pharo*  
100%



Development

Production

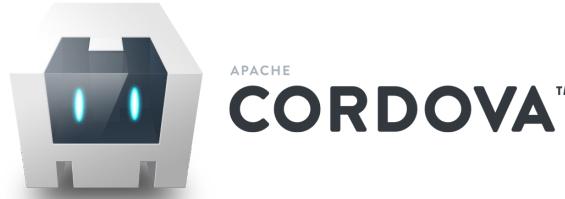


*100%  
Javascript*

# Pharo 100%



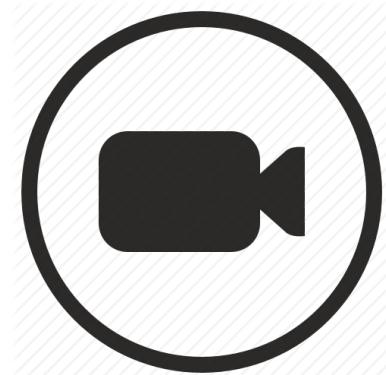
1. Write Tests
2. Pass the tests
3. Export to JS



**100%**  
*Javascript*



# Testing JS Generated Code





# Tests Talk to Web Browsers

20

The screenshot shows the Pharo IDE's Test Runner window. At the top, there are tabs for 'Pharo', 'Tools', 'PharoJS Playground', and 'System'. Below the tabs, a 'Running Tests' progress bar is shown. The main area is titled 'Test Runner' and displays a list of test cases under 'ligh|pharo'. The list includes: LjDiceListAppTest, LjDiceAdderAppTest, LjDiceAdderAppTestJS, LjDiceListAppTestJS, LjSingleDiceAppTest, LjSingleDiceAppTestJS, PjCounterBrowserAppT, PjJavascriptCounterBr, PjHelloWorldAppTest, PjJavascriptHelloWorld, and PjCounterTest. A dropdown menu is open over 'Test CaseA|TestCaseA'. The status bar at the bottom contains buttons for 'Run Selected', 'Run Profiled', 'Run Coverage', 'Run Failures', 'Run Errors', and 'File out results'.

1. Start Server

The screenshot shows a web browser window with the title 'PharoJS Dice'. The address bar shows 'file:///Users/noury/Documents/'. The page itself is titled 'Dice Group with Total using Lightweight Observer'. It features a row of six dice icons with faces showing 1, 4, 6, 5, 4, and 2. Below the dice is a green button labeled '22' and a blue button labeled 'Roll Dice'. There are also '+' and '-' buttons between the dice icons.



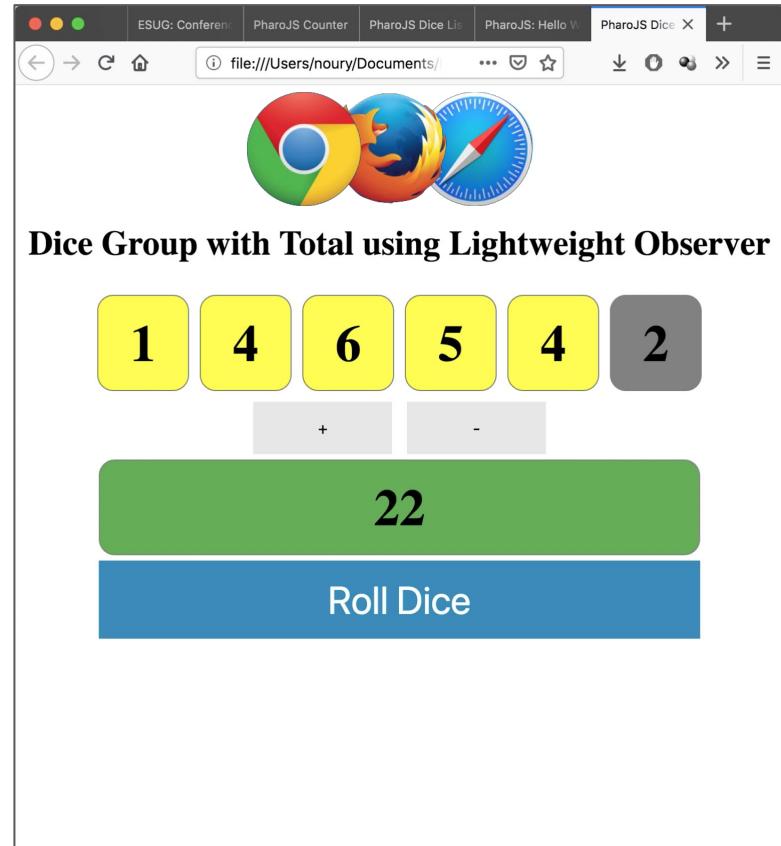
# Tests Talk to Web Browsers

21

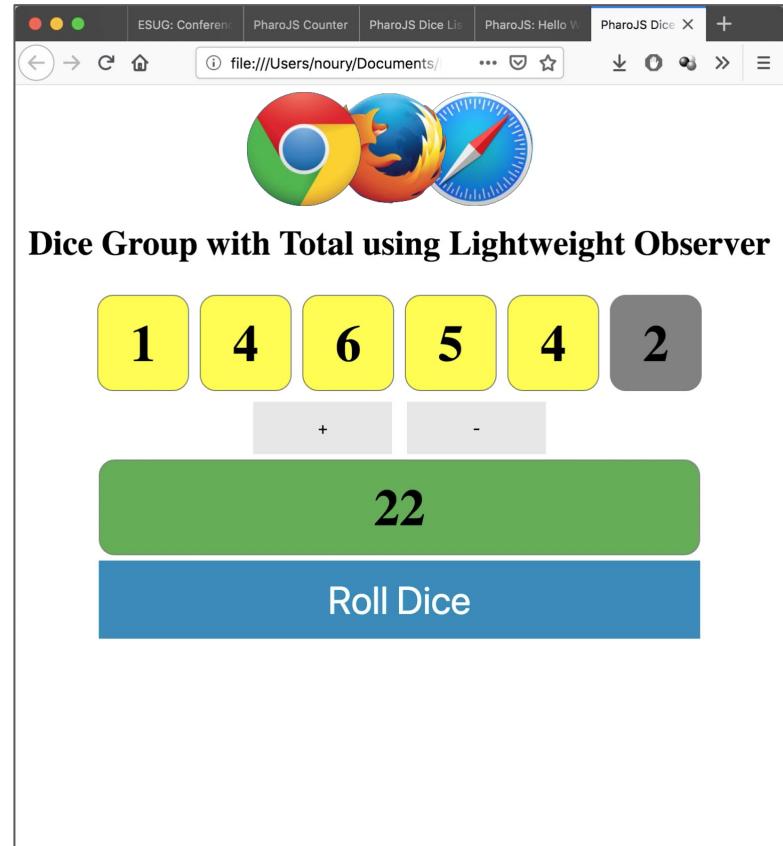
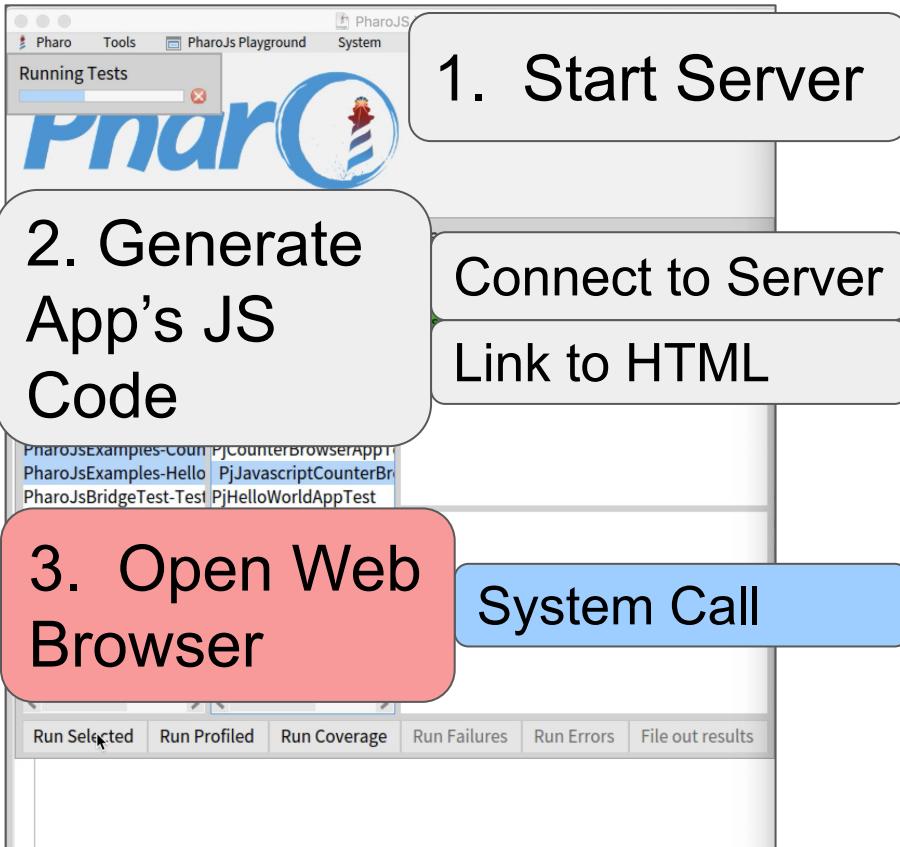
The screenshot shows the Pharo IDE interface. At the top, there's a menu bar with 'Pharo', 'Tools', 'PharoJS Playground', and 'System'. Below the menu is a toolbar with icons for 'Running Tests', 'PharoJS Counter', 'PharoJS Dice List', 'PharoJS Hello', and 'PharoJS Dice'. A large blue 'Pharo' logo is on the left. In the center, a light gray box contains the text '1. Start Server'. To the right, a pink box contains '2. Generate App's JS Code'. Below these, a blue box contains 'Connect to Server' and 'Link to HTML'. At the bottom, a list of test classes is shown in a browser-like view:

- PharoJSExamples-Count
- PjCounterBrowserAppTest
- PharoJSExamples-Hello
- PjHelloWorldAppTest
- PharoJSBridgeTest-Test
- PjJavascriptCounterBrowserTest
- PharoJSCoreLibrariesTest
- PjJavascriptHelloWorldTest
- PharoJSCoreLibrariesTest
- PjCounterTest
- PharoJSTranspilerOptIn
- LightweightObserver-Example
- LightweightObserver-Test
- LightweightObserverJS
- LightweightObserverJS

At the very bottom are buttons for 'Run Selected', 'Run Profiled', 'Run Coverage', 'Run Failures', 'Run Errors', and 'File out results'.

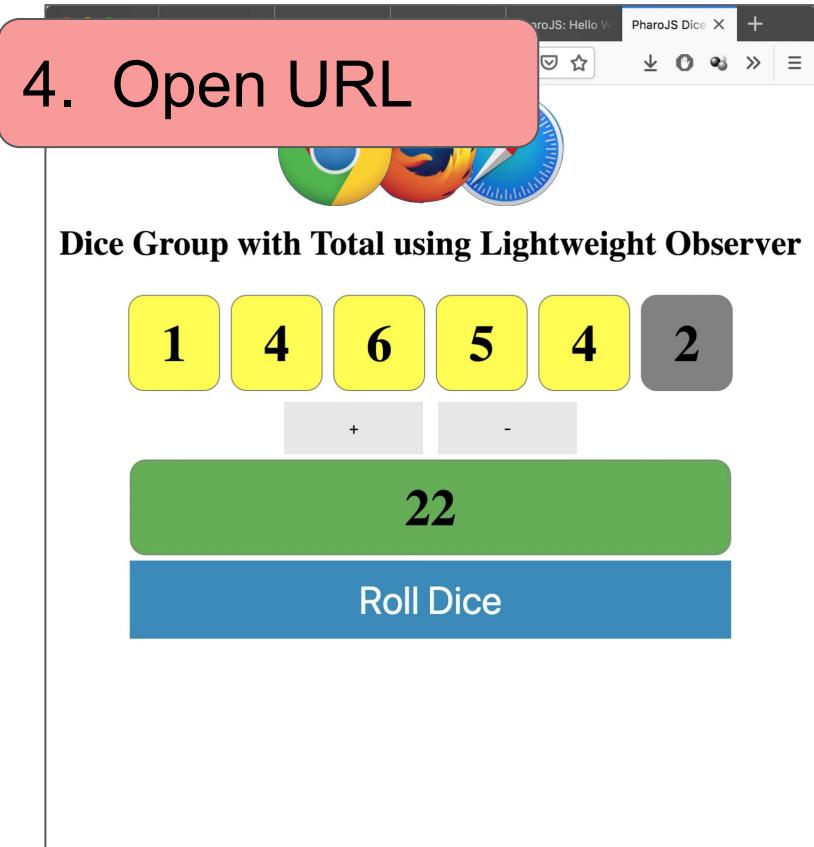
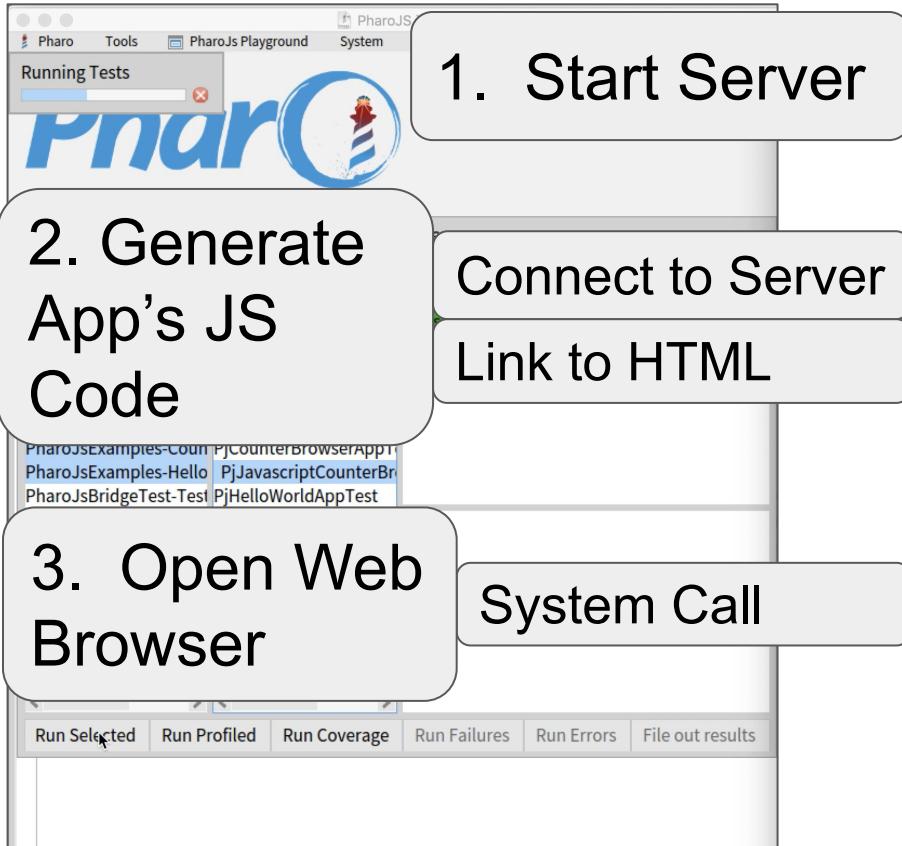


# Phar<sup>JS</sup> Tests Talk to Web Browsers



# Phar<sup>JS</sup> Tests Talk to Web Browsers

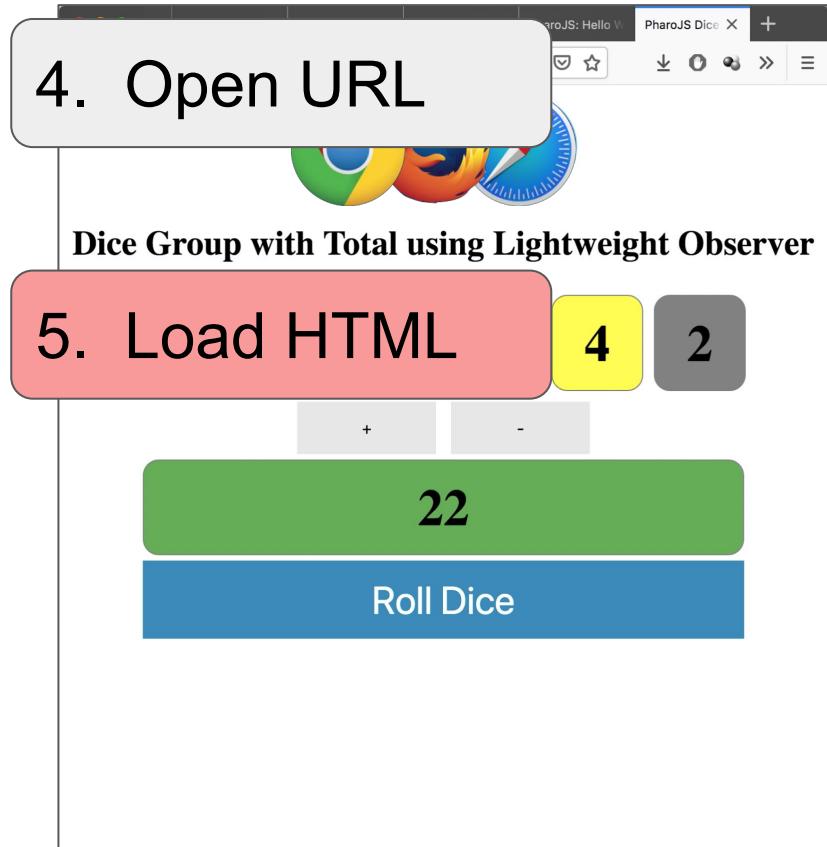
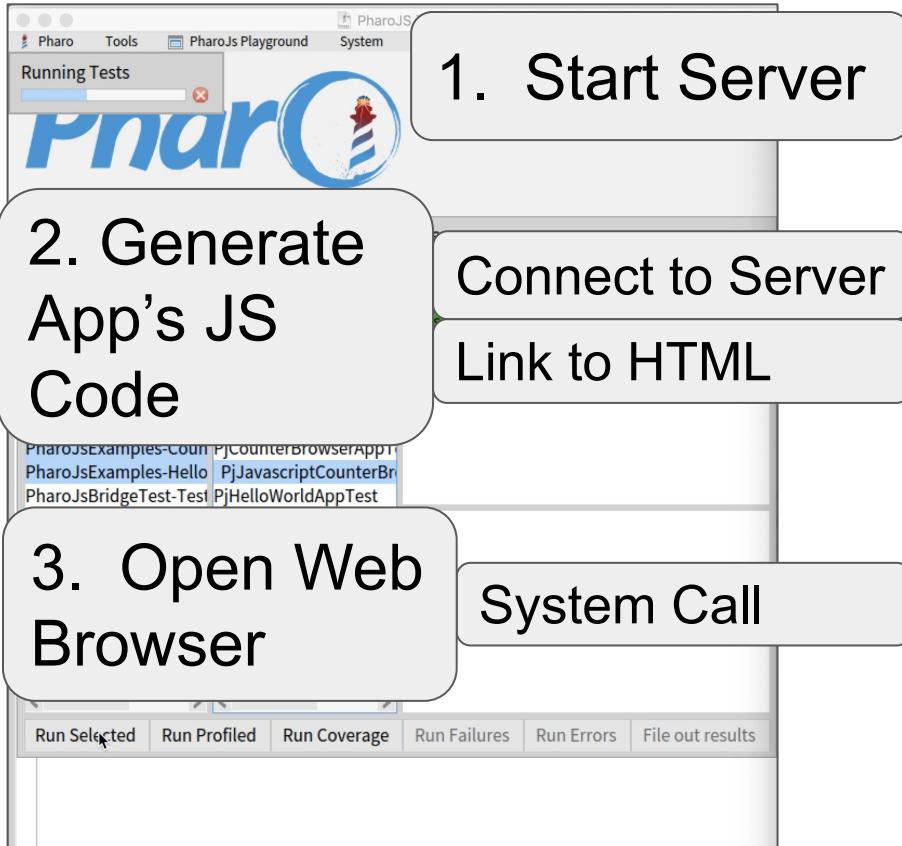
23





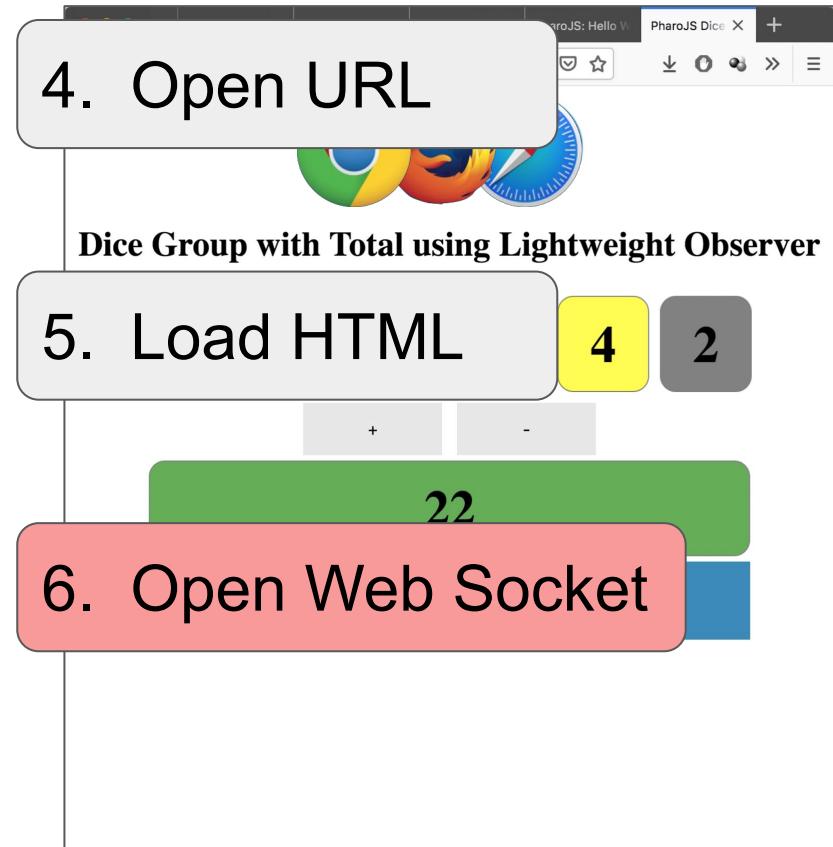
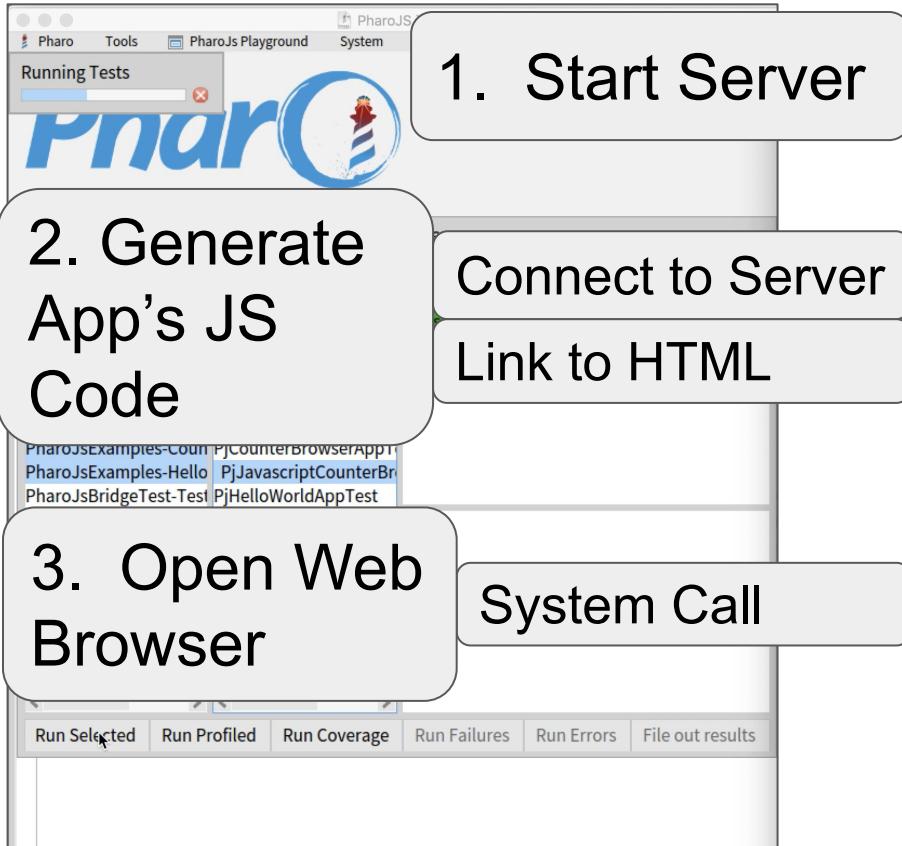
# Tests Talk to Web Browsers

24



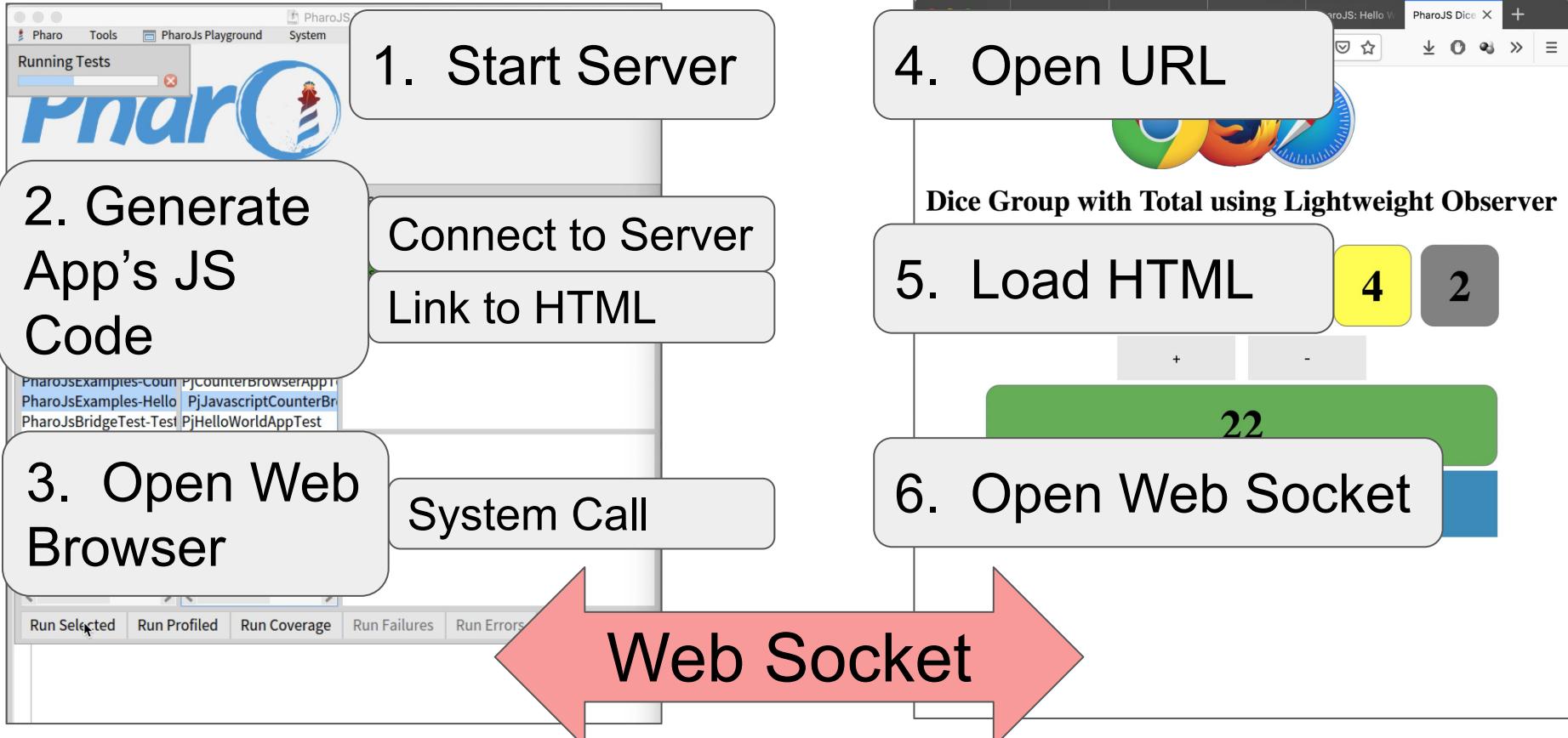
# PharJS Tests Talk to Web Browsers

25



# PharJS Tests Talk to Web Browsers

26



Smalltalk REPL using PharoJS X +

← → ⌂ ⌂ https://pharojs.org/repl

https://pharojs.org/repl

27

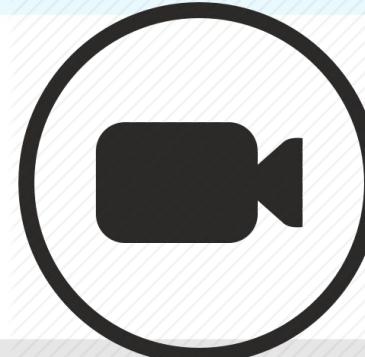


## *Smalltalk REPL using PharoJS*

```
I loveString |  
loveString := String streamContents: [ : stream |  
    stream  
        << $I;  
        space;  
        << 'love Pharo!' ].  
Transcript cr; show: loveString.
```

Transcript

I love Pharo!



Your Smalltalk code here

Eval

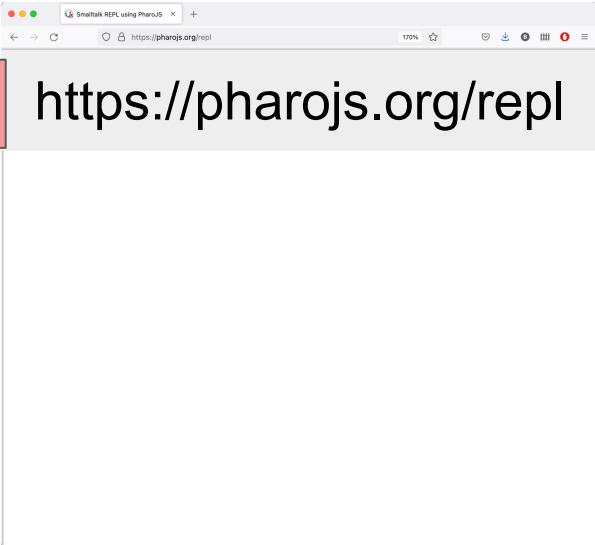
# HTML is a String in the image



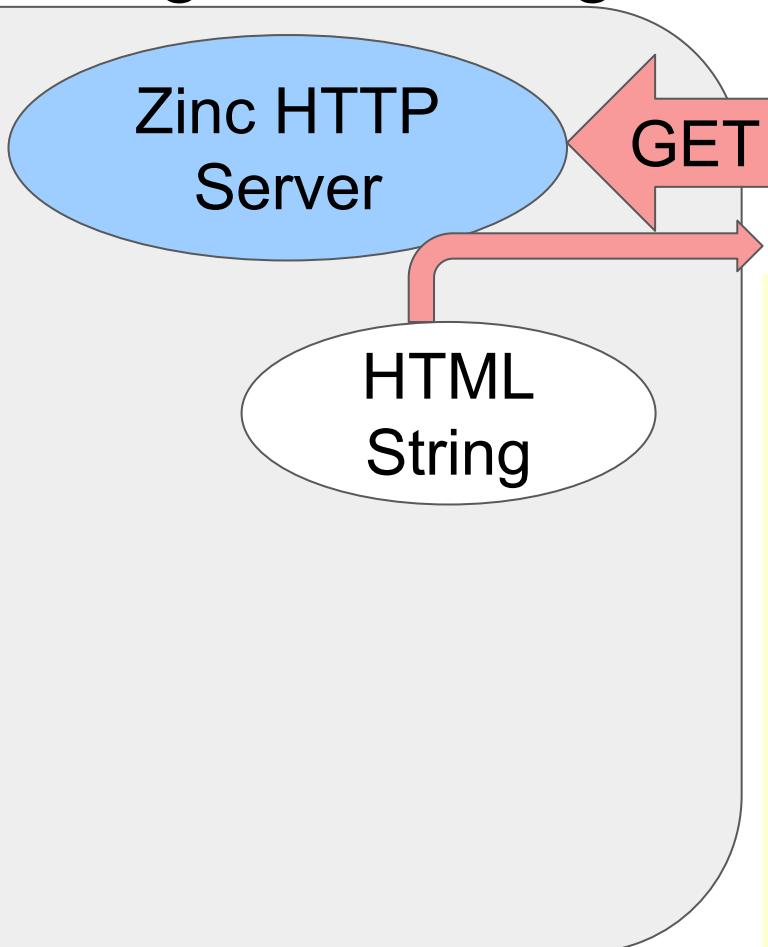
Zinc HTTP  
Server

GET

HTML  
String



# HTML is a String in the image



```
<!DOCTYPE html>
<html>
...

</script>
</body>
</html>
```

# Browser Processes the HTML



Zinc HTTP  
Server



```
<!DOCTYPE html>
<html>
...

</script>
</body>
</html>
```

# Browser Loads Resources



Zinc HTTP  
Server



```
<!DOCTYPE html>
<html>
...

</script>
</body>
</html>
```

# Browser Requests JavaScript Code



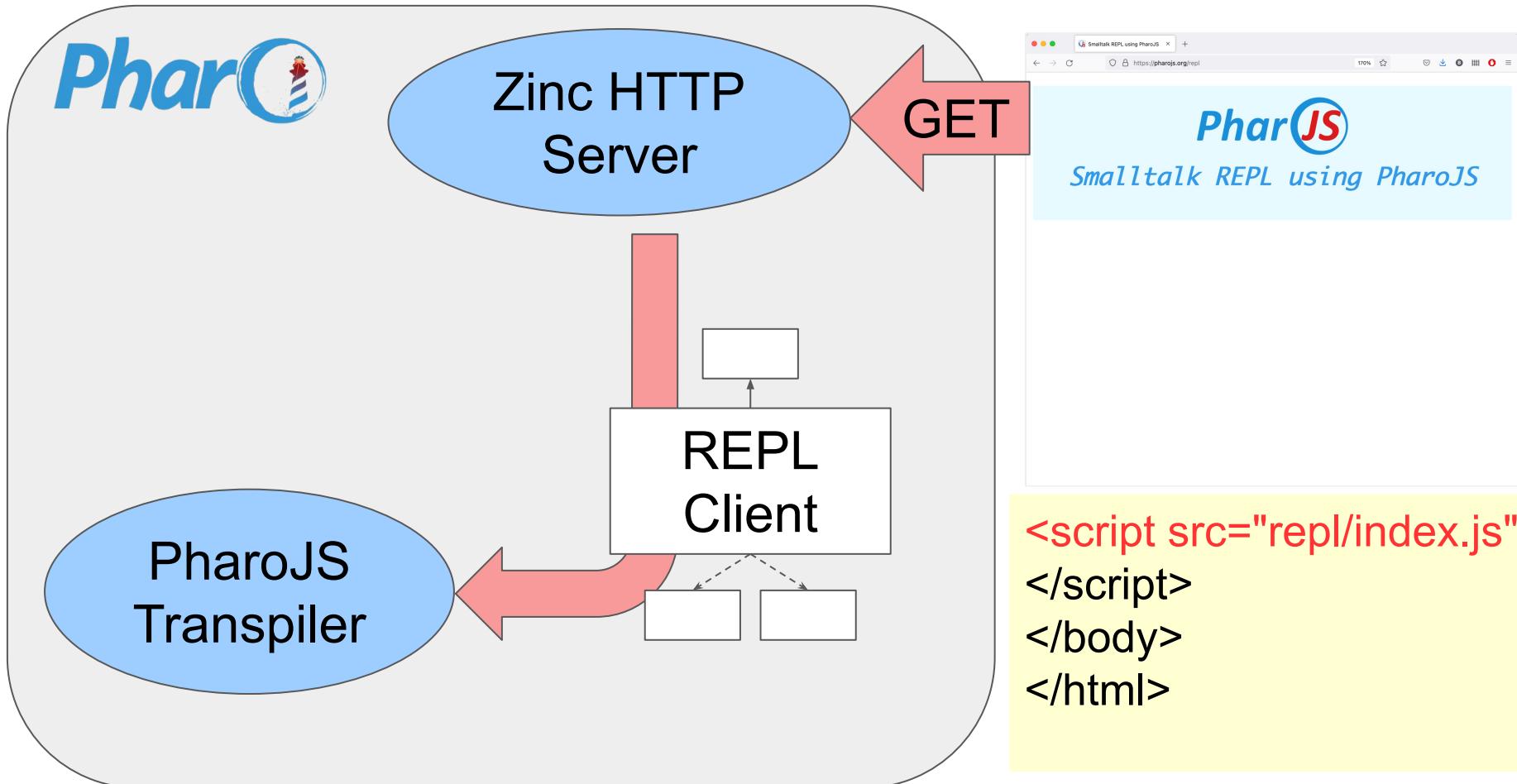
Zinc HTTP  
Server

GET

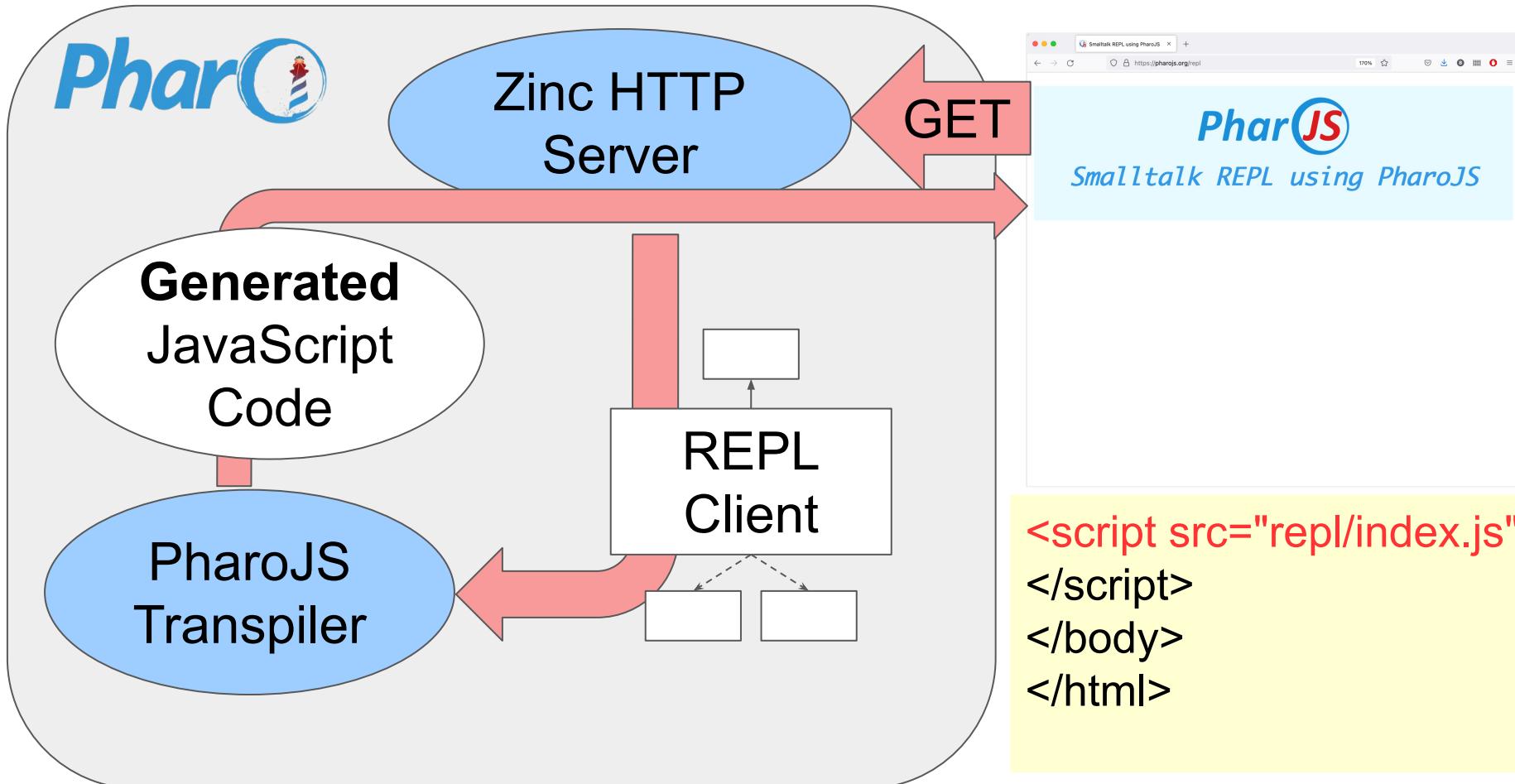


```
<script src="repl/index.js">
</script>
</body>
</html>
```

# REPL Client JS Code is Generated



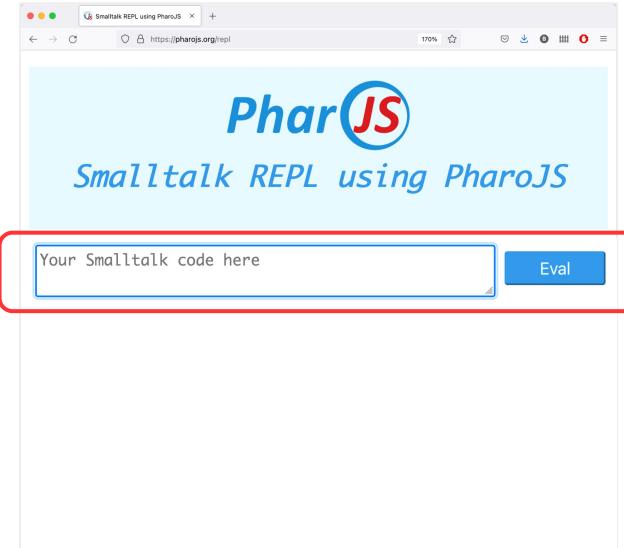
# REPL Client JS Code is Generated



# Client Creates and Links DOM Elements

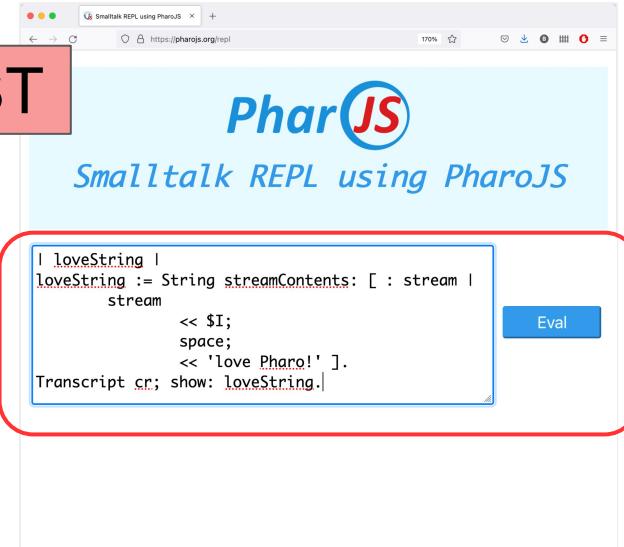


Zinc HTTP  
Server

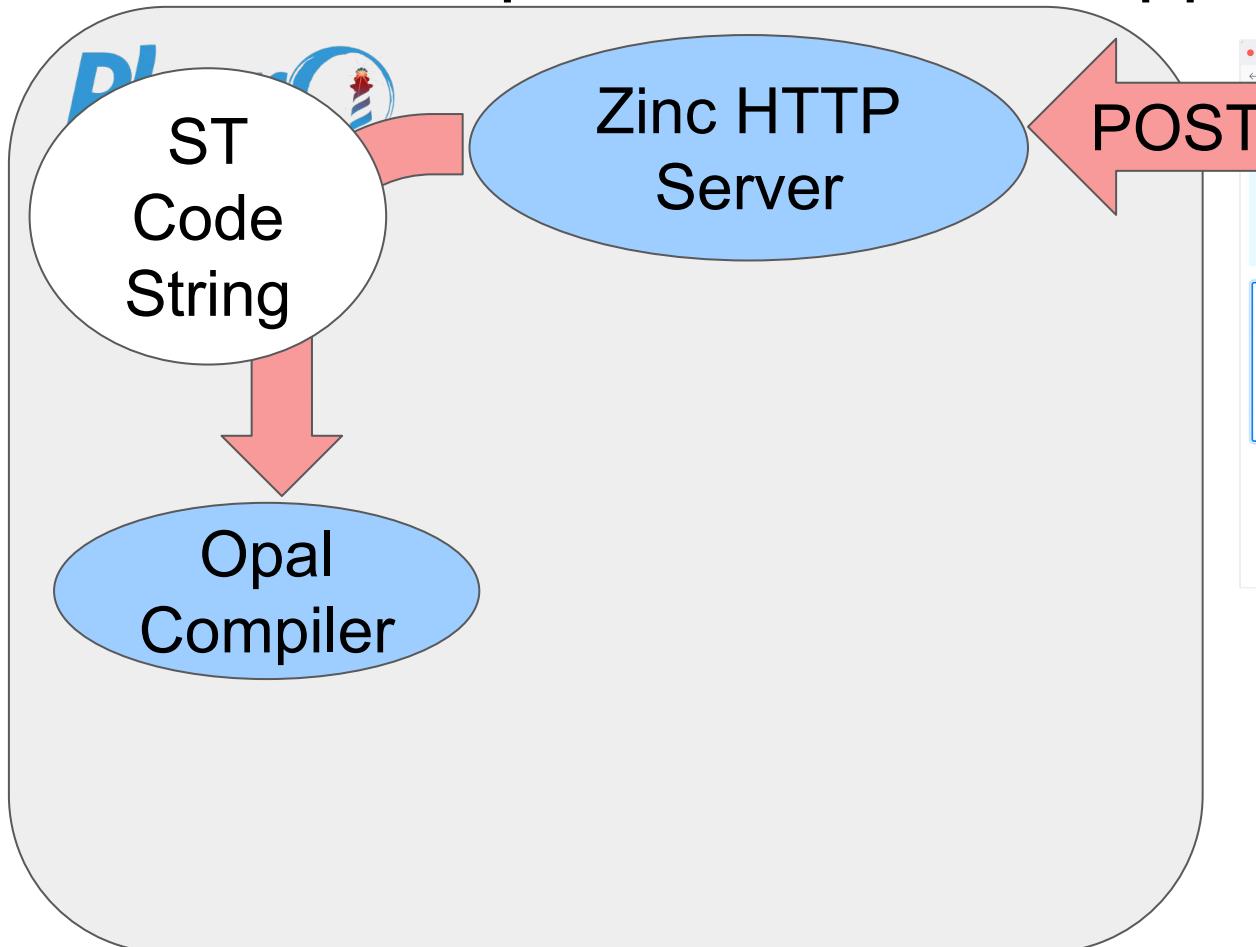


```
<script src="repl/index.js">  
</cript>  
</body>  
</html>
```

# Client Sends ST Code Snippet



# Server Compiles ST Code Snippet

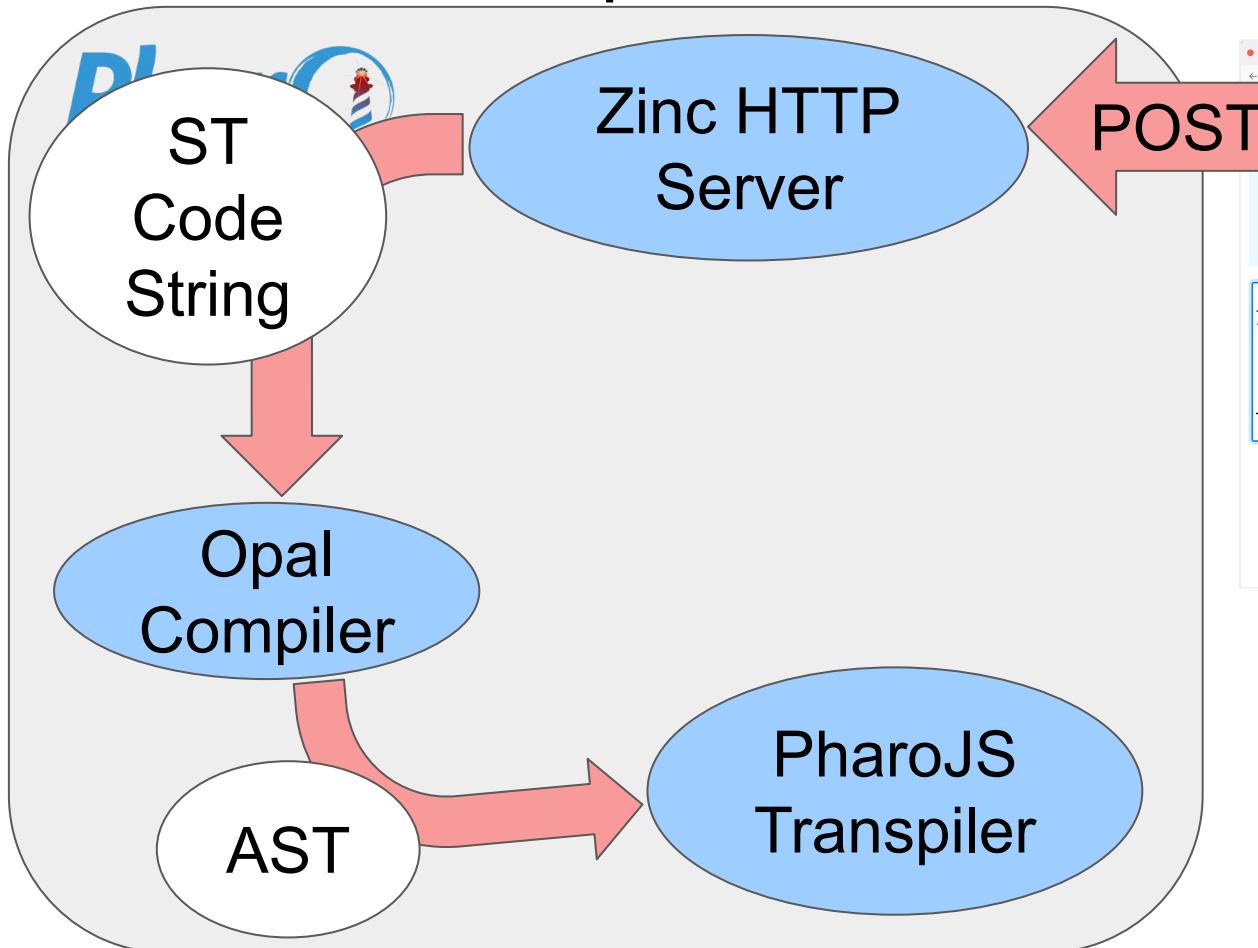


A screenshot of a web browser window titled "Smalltalk REPL Using PharoJS". The URL is <https://pharojs.org/repl>. The page has a light blue header with the "PharJS" logo and the text "Smalltalk REPL using PharoJS". Below the header, there is a code editor area containing the following Smalltalk code:

```
I loveString !
loveString := String streamContents: [ : stream |
    stream
        << $I;
        space;
        << 'love Pharo!' ].
Transcript cr; show: loveString.
```

On the right side of the code editor, there is a blue "Eval" button.

# PharoJS Transpiles AST



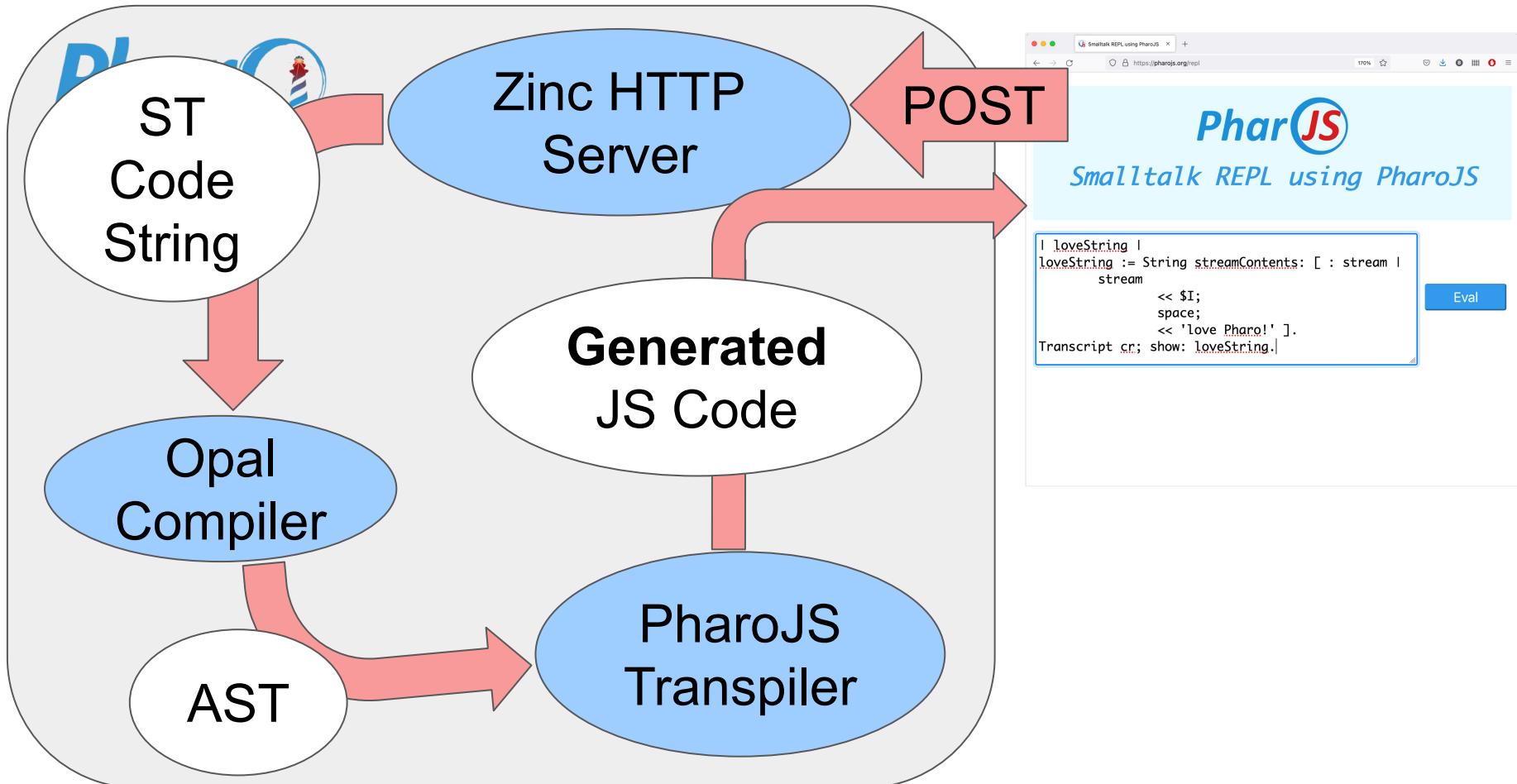
Smalltalk REPL using PharoJS

```
I loveString !
loveString := String streamContents: [ : stream |
    stream
        << $I;
        space;
        << 'love Pharo!' ].
Transcript cr; show: loveString.
```

Eval

The screenshot shows a web browser window titled "Smalltalk REPL Using PharoJS". The page displays a code editor containing Smalltalk code. The code defines a method `loveString` that creates a string containing the text "love Pharo!". It then prints this string to the transcript. A blue button labeled "Eval" is visible on the right side of the code editor.

# Server Sends Generated JS



# Client Executes Generated JS Code



Zinc HTTP  
Server

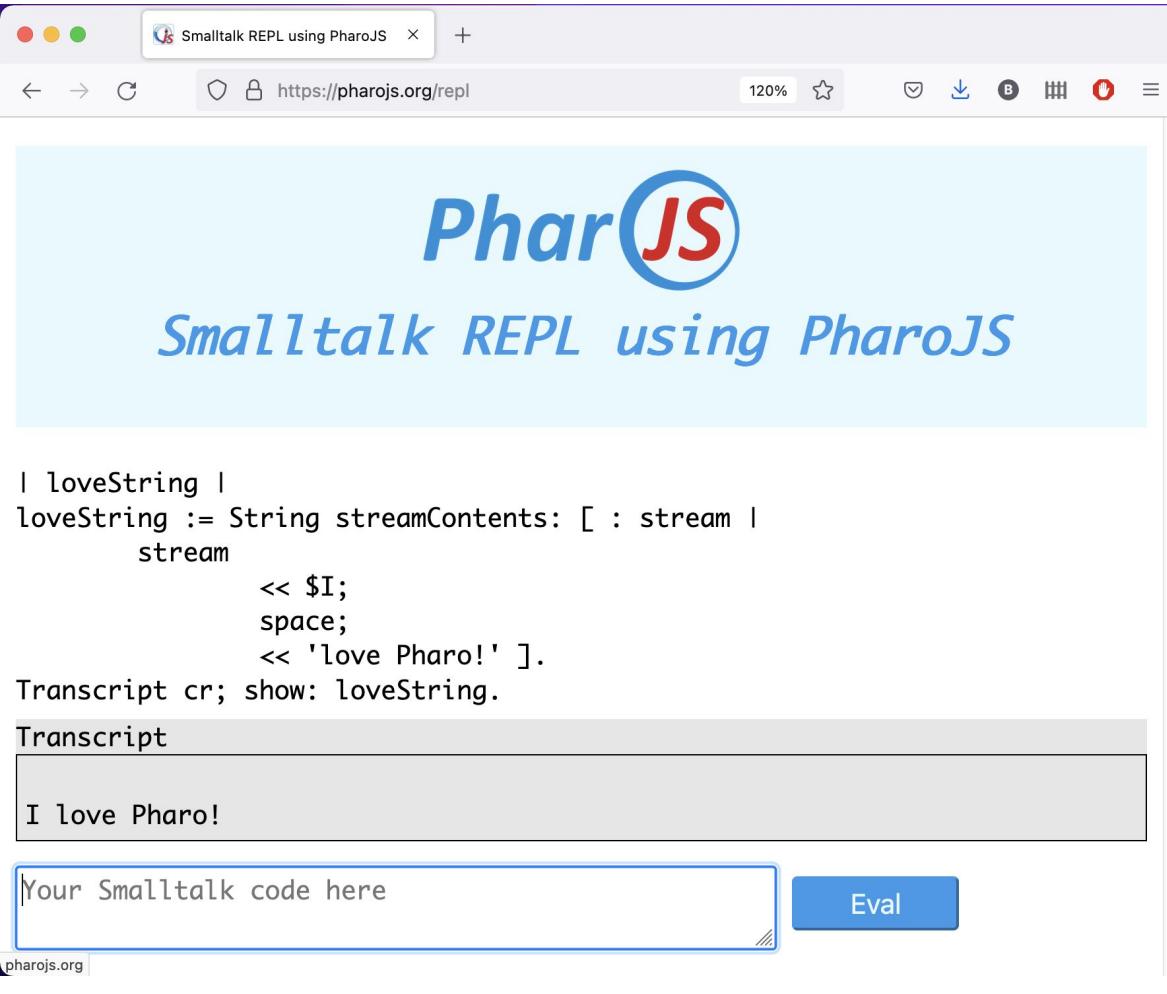
The screenshot shows a web browser window titled "Smalltalk REPL Using PharoJS". The URL is <https://pharojs.org/repl>. The page has a light blue header with the "PharoJS" logo and the text "Smalltalk REPL using PharoJS". Below the header, there is some Smalltalk code:

```
IloveString := String streamContents: [: stream |  
    stream << $I;  
    space;  
    << 'Love Pharo!' ].  
Transcript cr; show: loveString
```

A red box highlights the "Transcript" section, which contains the output "I love Pharo!". Below it is a text input field with placeholder text "Your Smalltalk code here" and a blue "Eval" button.

# Small App

- Client+Server
  - 10 classes
  - 64 methods
- Tests
  - 1 class
  - 20 methods



The screenshot shows a web browser window titled "Smalltalk REPL using PharoJS". The URL is <https://pharojs.org/repl>. The page features the PharoJS logo and the text "Smalltalk REPL using PharoJS". Below this, there is a code editor area containing the following Smalltalk code:

```
I loveString |
loveString := String streamContents: [ : stream |
  stream
    << $I;
    space;
    << 'love Pharo!' ].
```

Transcript cr; show: loveString.

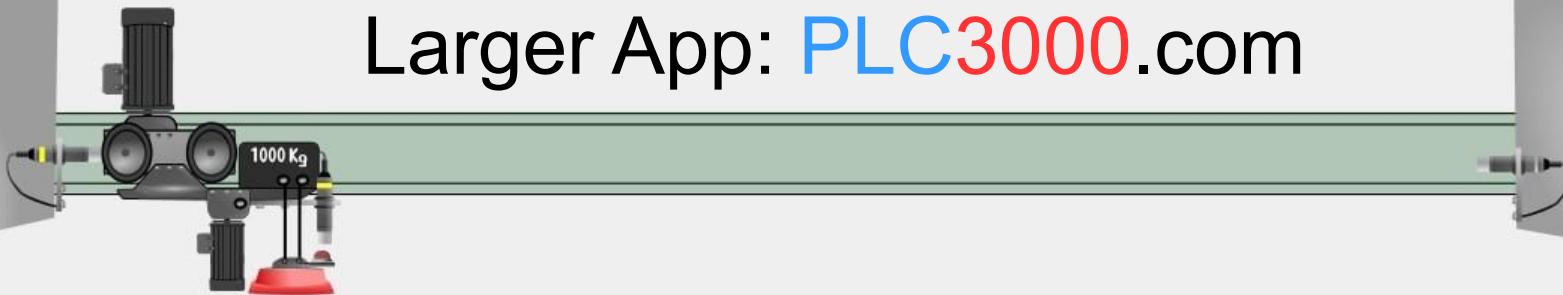
Below the code editor is a "Transcript" output area showing the result:

```
I love Pharo!
```

At the bottom, there is a text input field with placeholder text "Your Smalltalk code here" and a blue "Eval" button.

Generated JS  
**267KB (+3KB)**

Larger App: [PLC3000.com](http://PLC3000.com)



# Teaching PLC Programming & Factory Automation



# PLC3000.com Metrics



- Client+Server
  - 342 classes
  - 2529 methods
- Tests
  - 108 classes
  - 1184 methods
  - 876 test runs



# PLC3000.com = Educational Software + Contents

**AUTOMATION OF AN IN/OUT CONTROL OF A CROSSROAD – TIMER, COUNTER**

The specifications are very simple. It aims at setting the light (output %Q4) as soon as the publication (%I1) is pressed.

Among possible solutions, we proposed to answer to the specifications with a Grafcet with 2 steps:

According to this Grafcet, as soon as %I1 is true, the step 1 has to be activated causing the forcing of %Q4 to 1.

**Wrong Code**

The first way to code this Grafcet in IL with the method Transition/Transition without Crossing Bits is:

```

LD %I1
S %Q4

```

**Code**

Ladder Logic:

Programmable Logic:

```

LD %I1
AND %M22
AND DetectionBas
LD %M1
OR %M40
LD %I1
AND %M22
AND DetectionHaut
LD %M40
OR %M43
LD %M43
AND %M22
LD %I1
AND %M22
AND DetectionBas
LD %M1
OR %M40
LD %I1
AND %M22
AND DetectionHaut
LD %M40
OR %M43
LD %M43
AND %M22
IF %M14 THEN
    %M2 := TRUE;
    %M1 := FALSE;
    %M4 := FALSE;
END;

```

**PLC**

System Bits:

S0	S1	S2
----	----	----

Memory:

M0	M1	M2	M3	M4	M5	M6	M7	M8
----	----	----	----	----	----	----	----	----

Inputs:

I0	I1	I2	I3	I4	I5	I6	I7
----	----	----	----	----	----	----	----

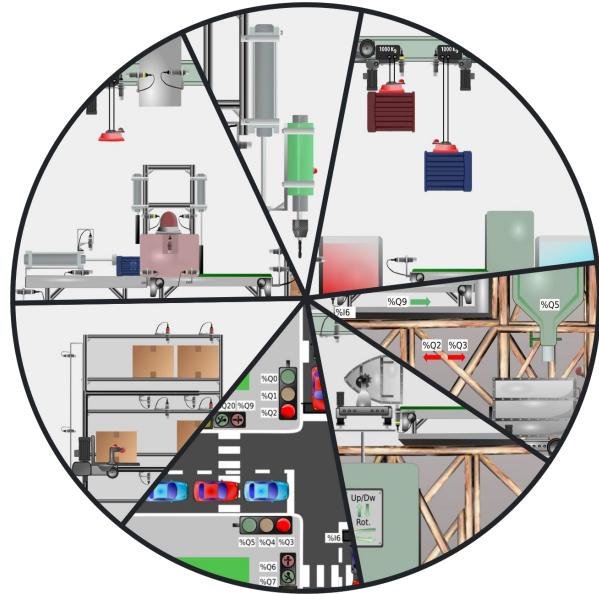
Outputs:

Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19
Q20	Q21	Q22	Q23						

Counters:

C0	0	R	LD	C1	42	R	LD
CD	QD	CU	QU	CD	QD	CU	QU
PV	100	÷	PV	42	÷		

Timers:



**27 (13 + 14)**  
**Exercises & Tutorials**

**3**  
**Programming Languages**

**PLC Simulator**

**7 (4+3)**  
**Physics Simulations**

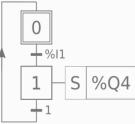
# PLC3000.com = Educational Software + Contents

**AUTOMATION OF AN IN/OUT CONTROL OF A CROSSROAD – TIMER, COUNTER**

The objective is to show the relevance of the *Crossing Bits* in relation to the reading direction of written code. To show it, we first propose erroneous programs, then an explanation of the error, and finally the good solution.

The specifications are very simple. It aims at setting the light (output %Q4) as soon as the publication (%I1) is pressed.

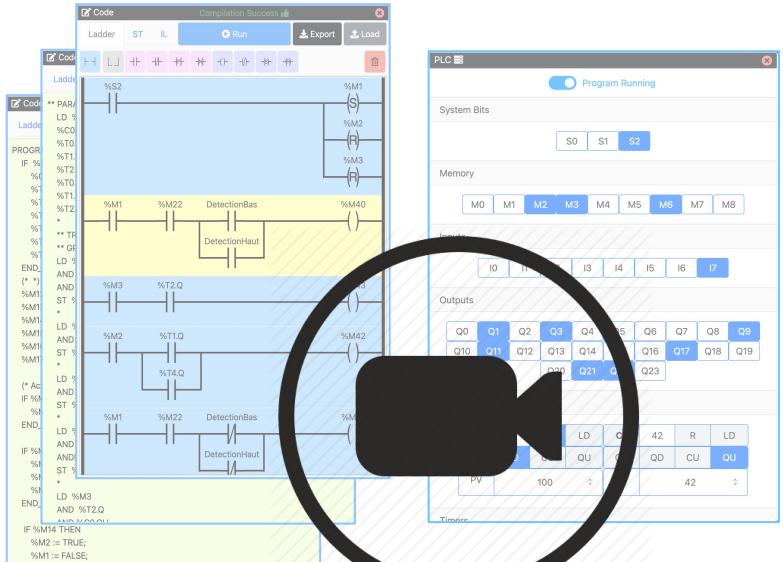
Among possible solutions, we proposed to answer to the specifications with a Graftec with 2 steps:



According to this Graftec, as soon as %I1 is true, the step 1 has to be activated causing the forcing of %Q4 to 1.

**Wrong Code**

The first way to code this Graftec in IL with the method Transition/Transition without *Crossing Bits* is:



**Ladder Logic:**

```

** PAR
LD %I0
LD %T0
LD %T1
LD %T2
LD %T3
LD %T4
LD %M1
LD %M2
LD %M3
LD %M4
LD %T2.0
LD %T1.Q
LD %T4.Q
LD %M1
LD %M2
LD %M3
LD %M4
LD %M13
AND %M1
AND %M2
AND %M3
AND %M4
AND %T2.0
AND %T1.Q
AND %T4.Q
AND %M1
AND %M2
AND %M3
AND %M4
IF %M14 THEN
    %M02 := TRUE;
    %M01 := FALSE;
    %Q04 := TRUE;
END.

```

**Instruction List (IL):**

```

LD %I0
LD %T0
LD %T1
LD %T2
LD %T3
LD %T4
LD %M1
LD %M2
LD %M3
LD %M4
LD %T2.0
LD %T1.Q
LD %T4.Q
LD %M1
LD %M2
LD %M3
LD %M4
LD %M13
AND %M1
AND %M2
AND %M3
AND %M4
AND %T2.0
AND %T1.Q
AND %T4.Q
AND %M1
AND %M2
AND %M3
AND %M4
IF %M14 THEN
    %M02 := TRUE;
    %M01 := FALSE;
    %Q04 := TRUE;
END.

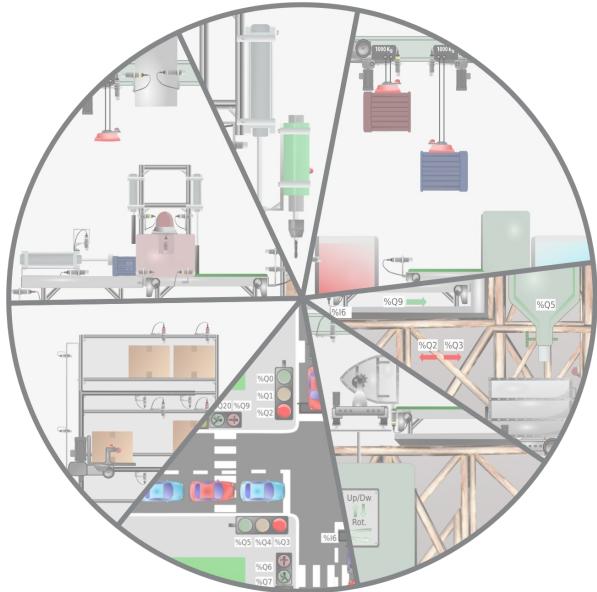
```

**PLC Memory View:**

S0	S1	S2						
M0	M1	M2	M3	M4	M5	M6	M7	M8

I0	I1	I3	I4	I5	I6	I7
----	----	----	----	----	----	----

Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19
D20	D21	D22	D23	D24	D25	D26	D27	D28	D29



**27 (13 + 14)**  
**Exercises & Tutorials**

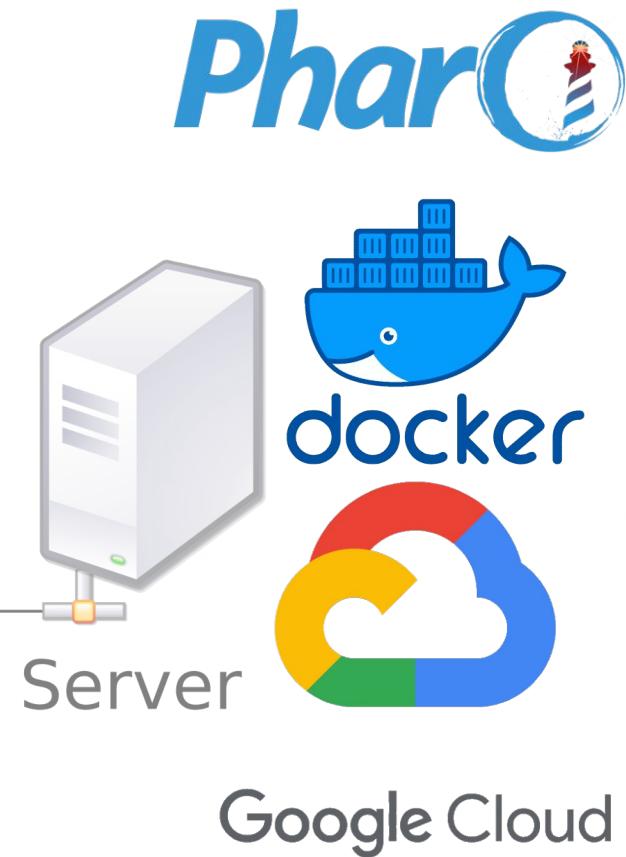
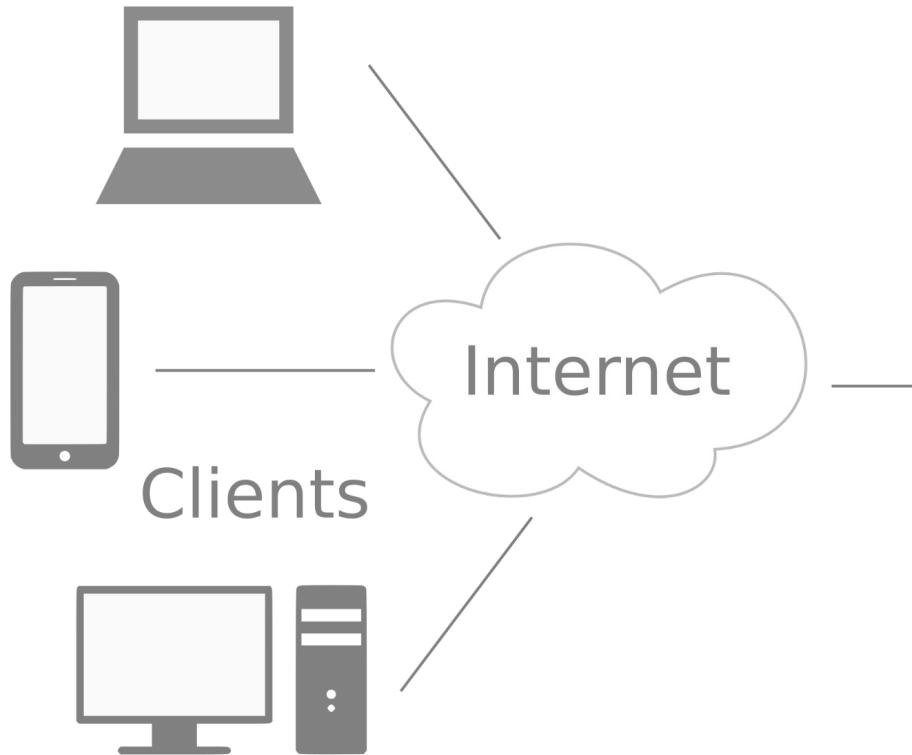
**3**  
**Programming Languages**

**PLC Simulator**

**7 (4+3)**  
**Physics Simulations**

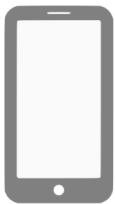
# PLC3000.com Server Side

46



# PLC3000.com Clients Run in Web Browsers

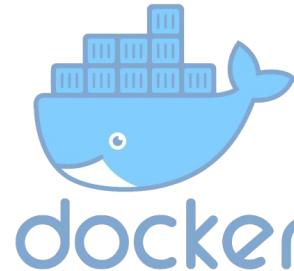
47



Clients



Server

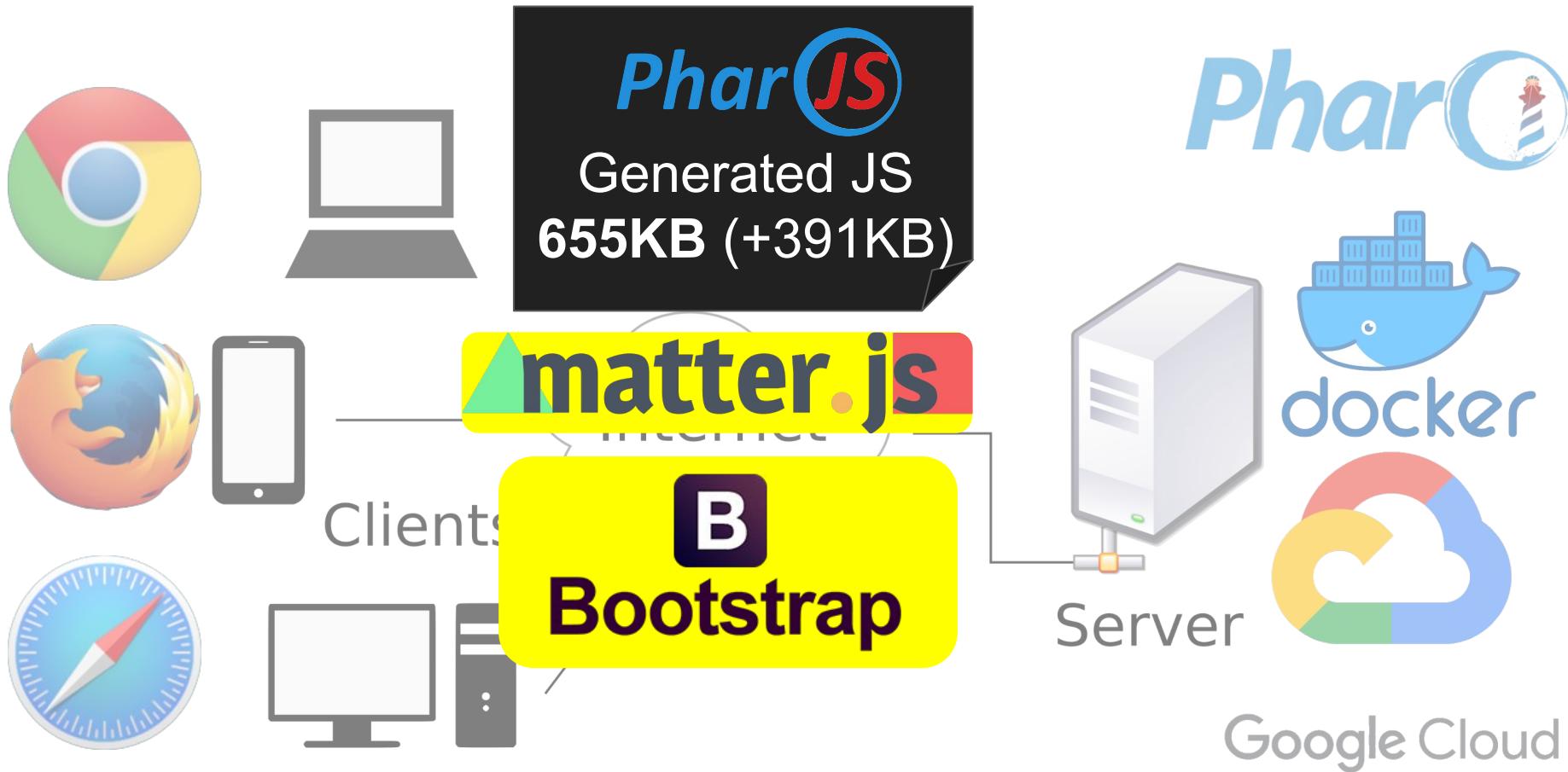


docker



Google Cloud

# PLC3000.com Client JS Code

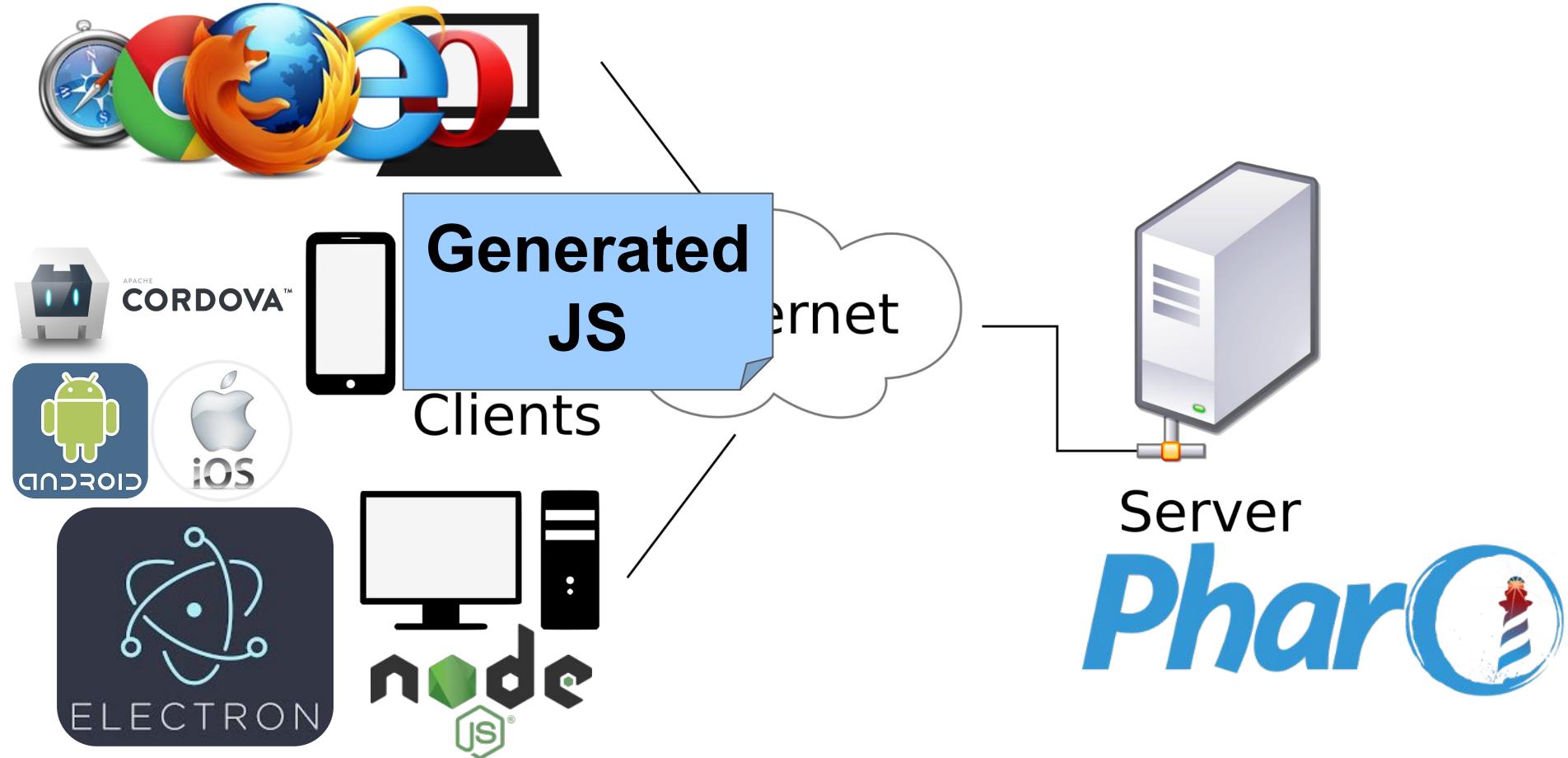


# Summary

 **PharJS** Supports Real World Applications

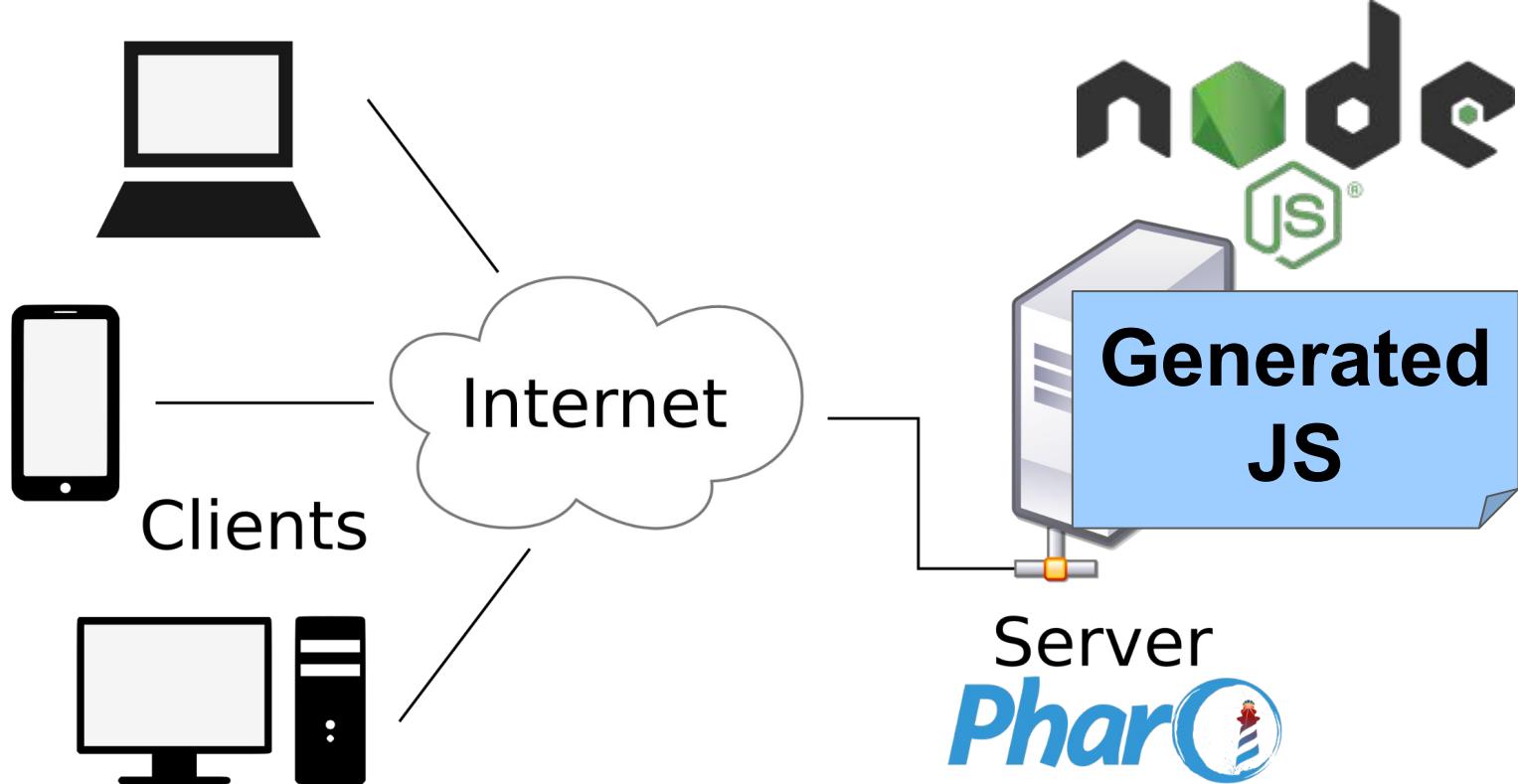
- Write 100% Pharo Code
- Reuse JS Libraries
- Tests + debugging in Pharo
  - Pharo talks to JavaScript
- Different Architectures are Possible

# Pharo on the Client Side = *PharJS*



# PharJS is for Server Side Too!

51



# PharJS Supports Different Workflows

52

Run-Time

vs

Development-Time

- HTML, CSS

- Handwritten Files ●
- Generated ● ●
- DOM Elements Creation & Setup ●
- Reuse Third-Party Libraries ●

- Javascript

- Generated ● ●
- Reuse Third-Party Libraries ●



← Tweet



Noury Bouraqadi  
@nourybouraqadi

...



PharoJS for Pharo 10 is out ! [github.com/PharoJS/PharoJS](https://github.com/PharoJS/PharoJS)... @pharoproject @pharojs #SmallTalk #javascript



for  
Pharo X  
is now Beta ;-)



April 1st, 2021

# PharJS Future Development

54

- Improved Middleware
  - Framework for Client-Server Apps
- Support latest JS constructs to reuse JS Frameworks
- Support more Pharo concepts (threads, slots, ...)
- Extended Support for Live/Interactive Programming
  - Hot code update : easy
  - Debugging generated JS code : complex

# Develop in Pharo, Run on JavaScript

**PharoJS.org**

Kindly supported by



Thanks to all the contributors

