ESUG 2023, Lyon (France)

PharJS

Hijack the JavaScript Ecosystem

Noury Bouraqadi & Dave Mason

IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

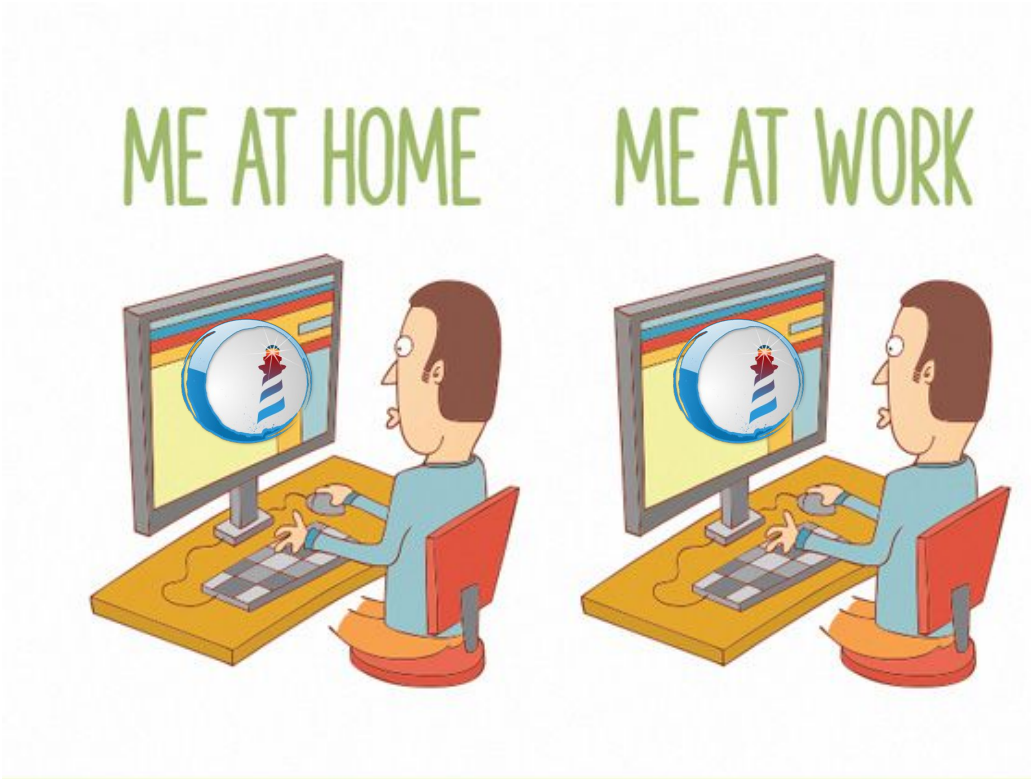
Toronto Metropolitan University

I am an **Addict**!

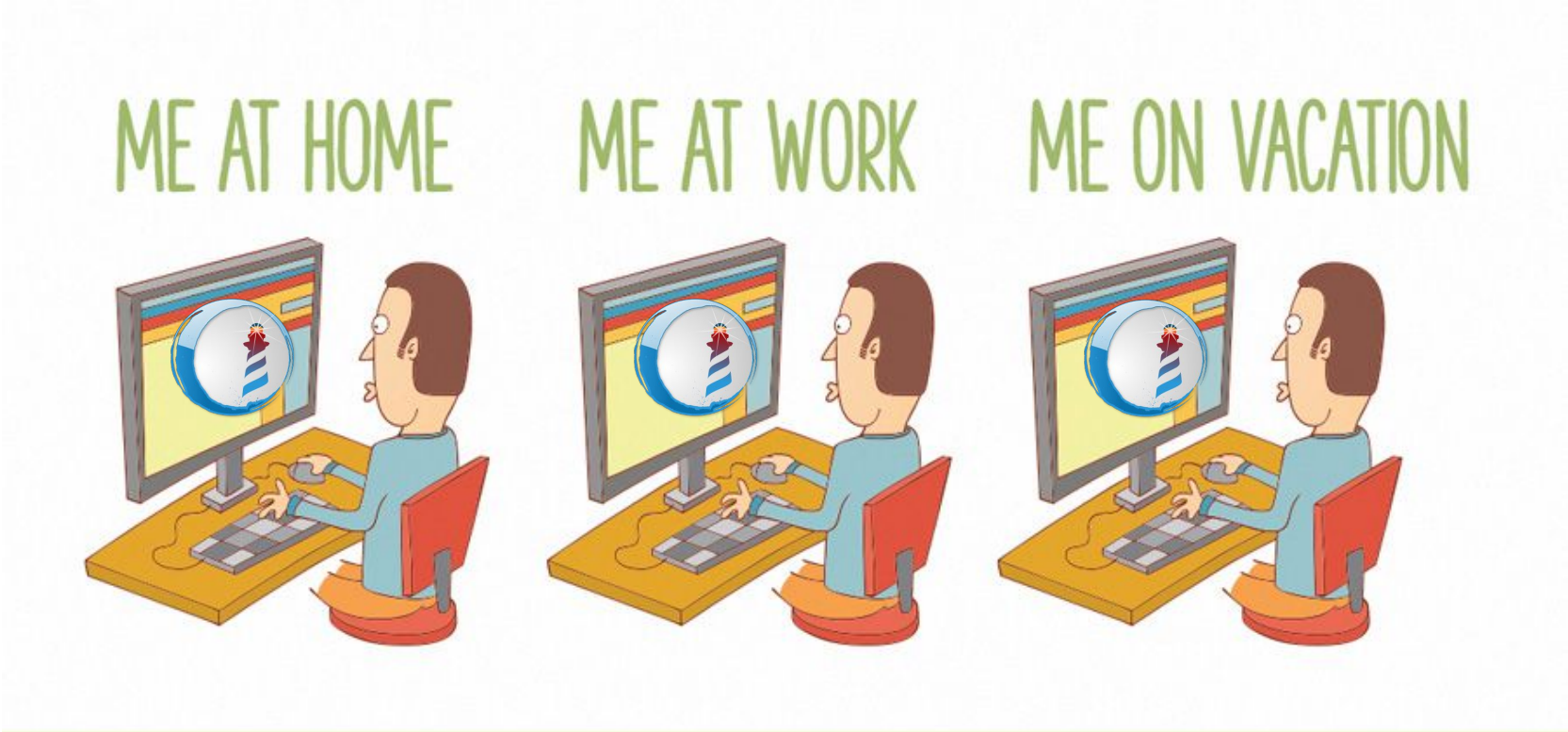
Not That Kind of Addict!

I am a Smalltalk Addict!

I am a Smalltalk Addict!



I am a Smalltalk Addict!



I am **Lazy**!

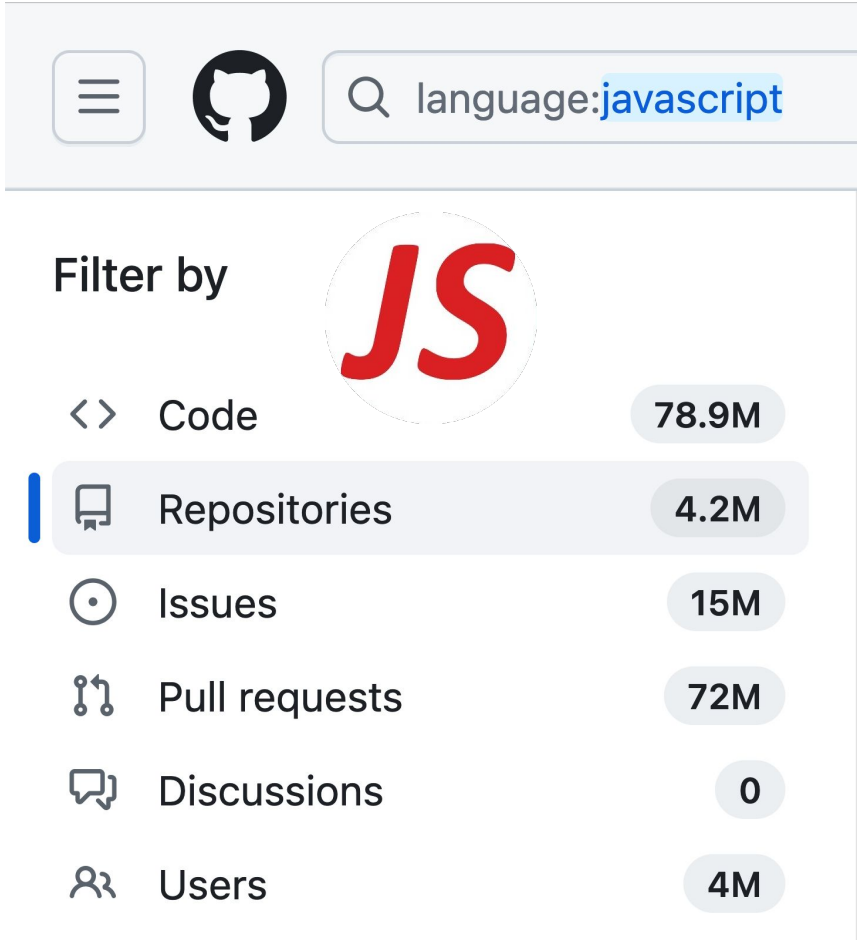
I am a Lazy Developer!

pharo-open-documentation /
awesome-pharo

A categorized community-driven collection of awesome Pharo libraries, tools, frameworks and software.

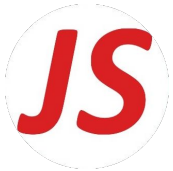
Thank You
Cyril Ferlicot

I am a Lazy Developer!



Addict + Lazy = PharJS

+

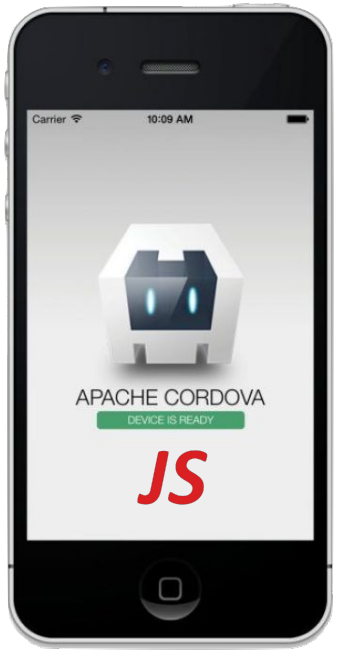


Development

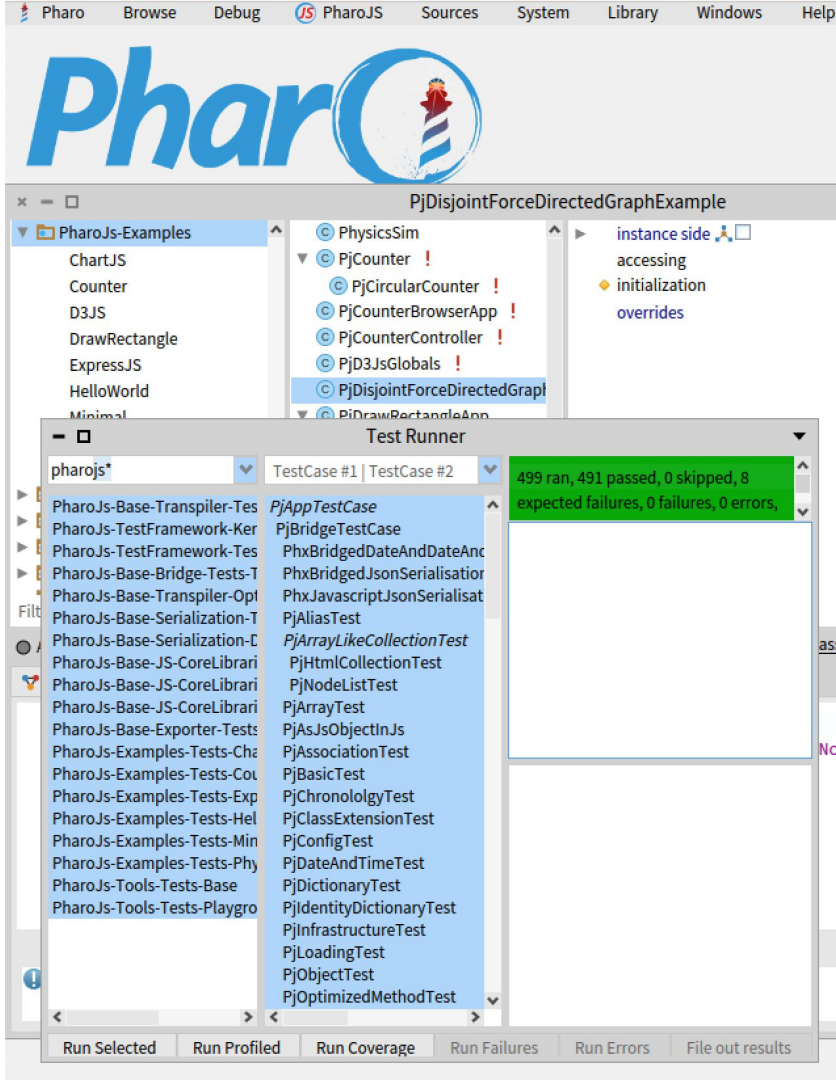
Libraries
&
Run-Time

- **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

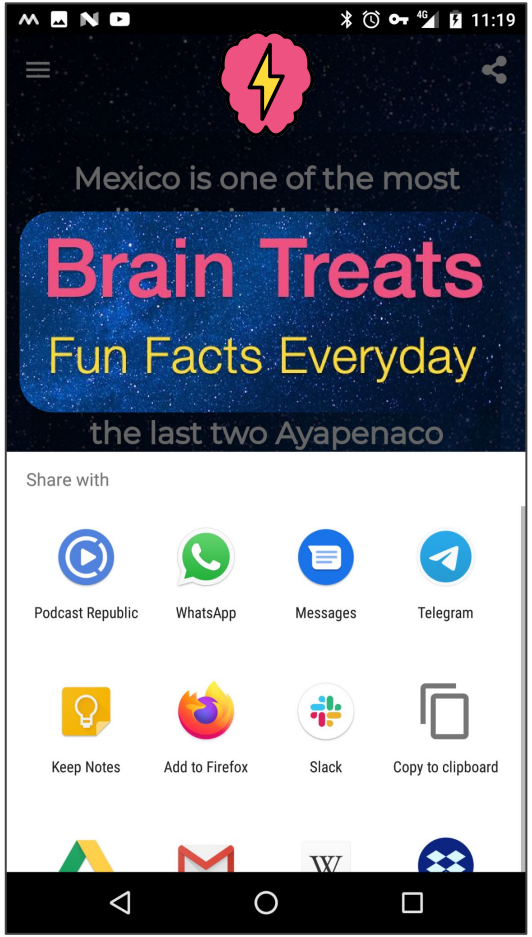
Mobile Apps with PharJS



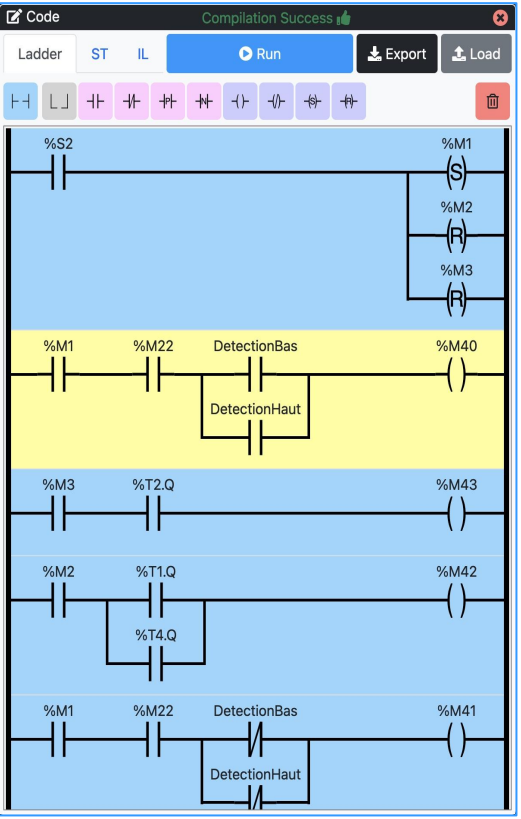
PharJS

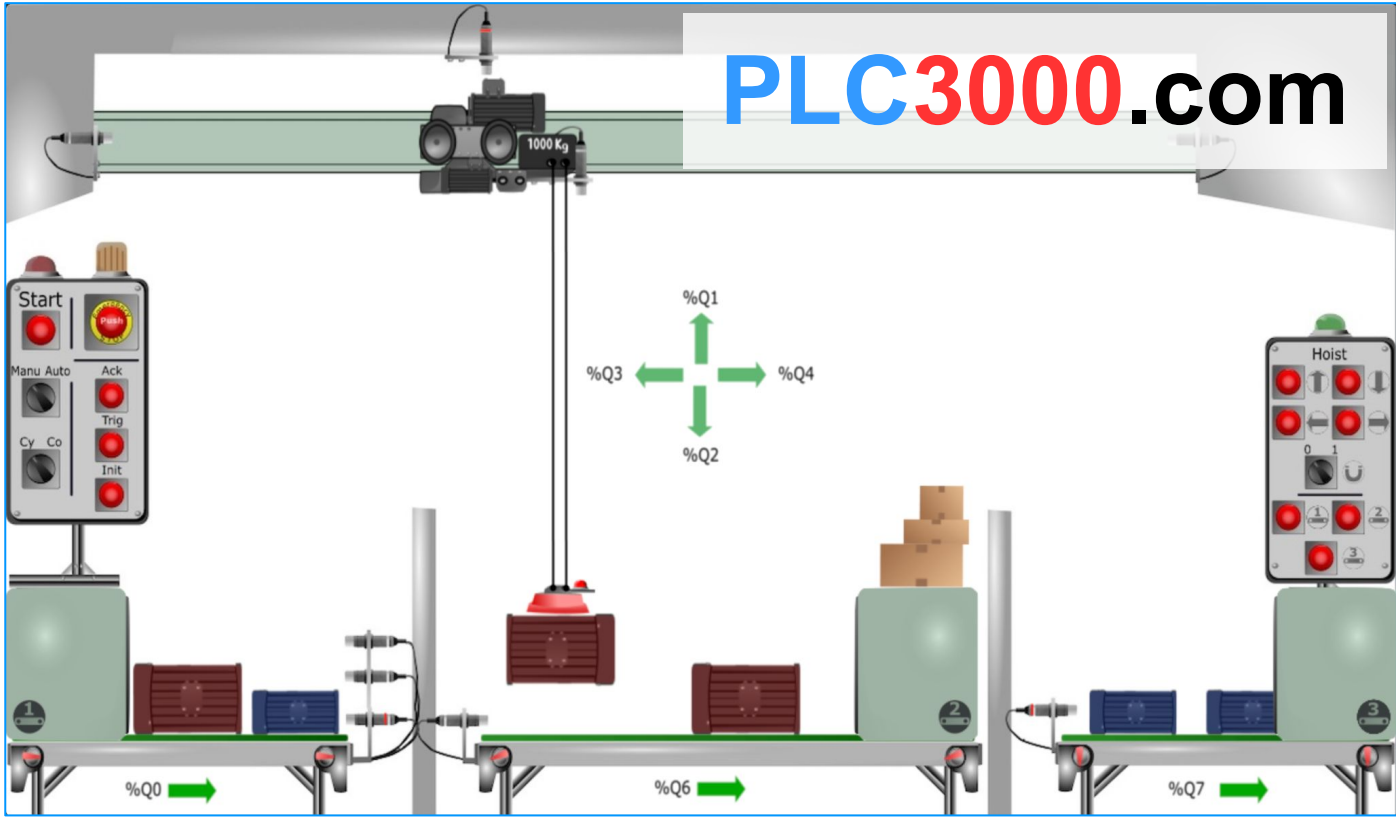


Mobile Apps with PharJS

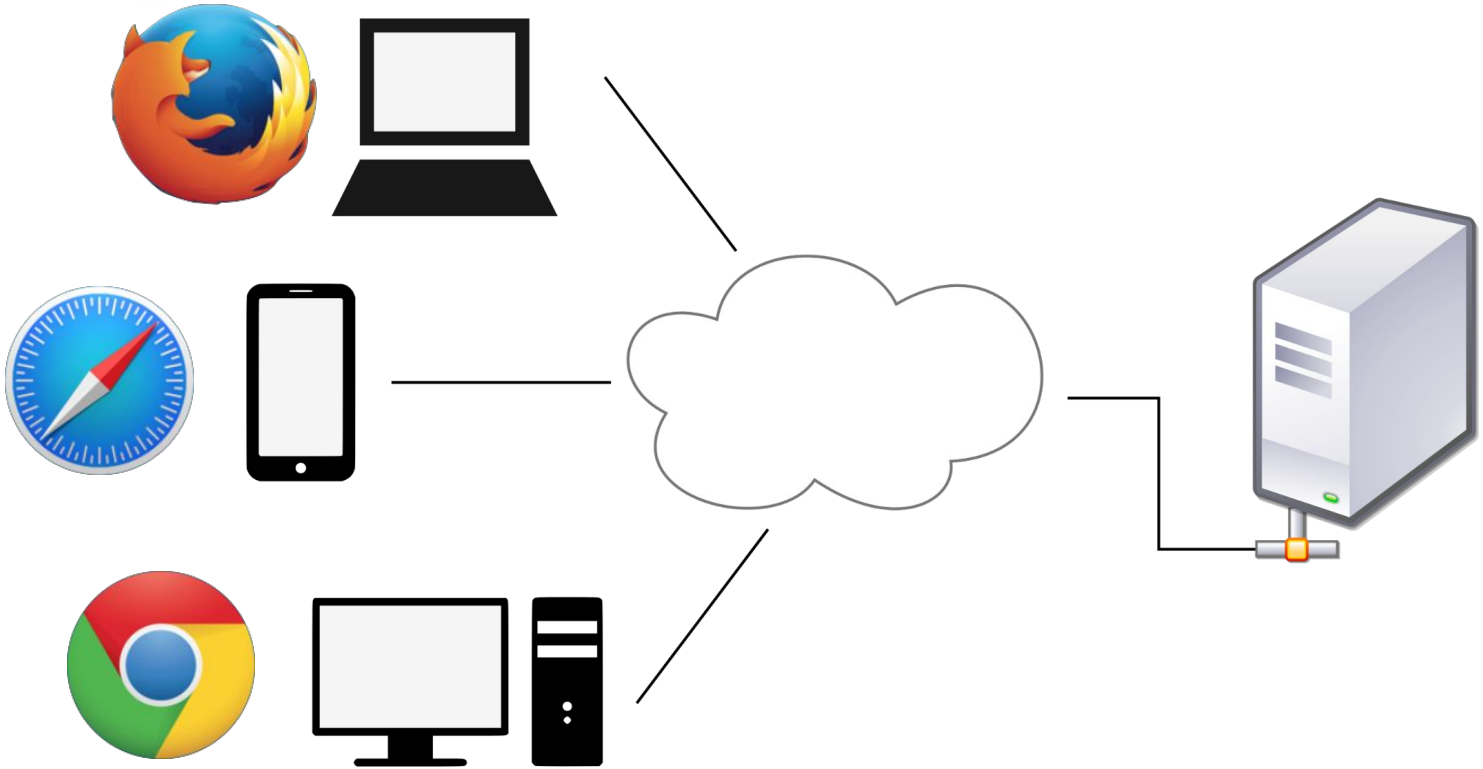


Web Apps with PharJS

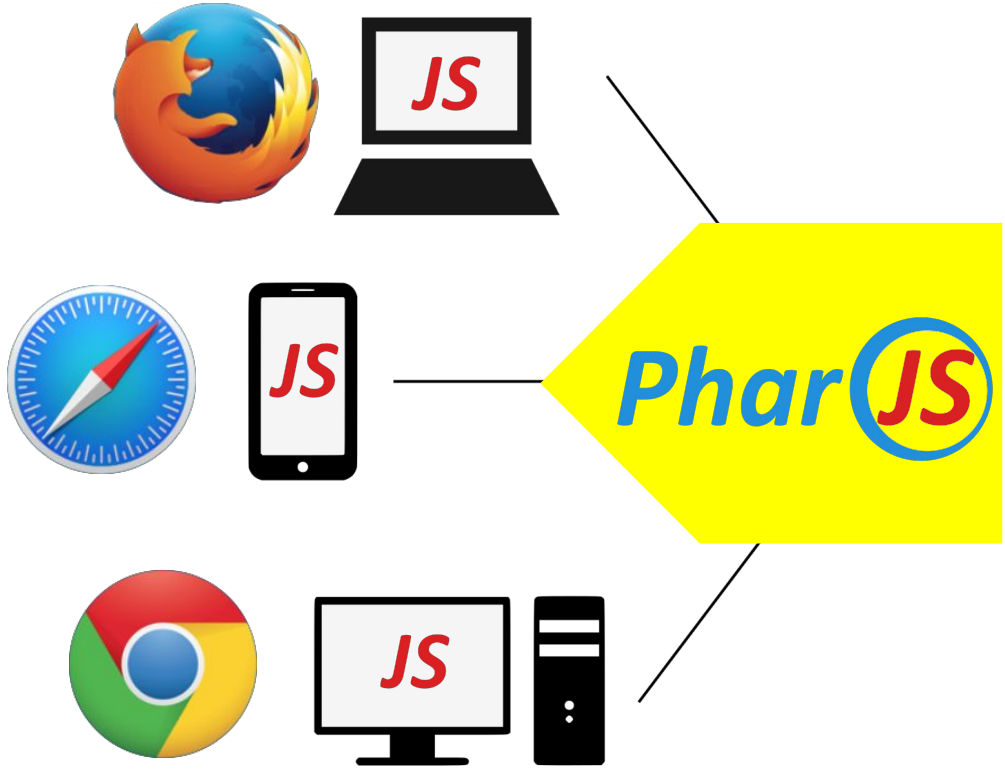


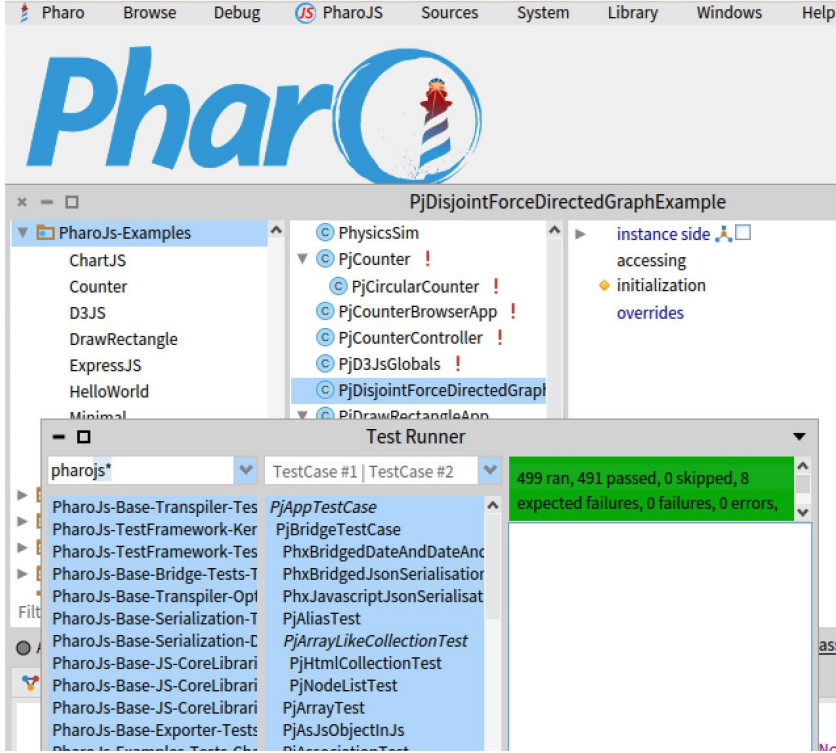


PLC3000.com Architecture



PLC3000.com Clients

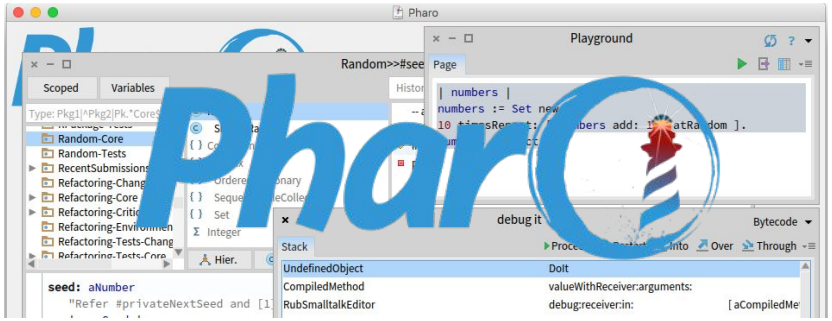


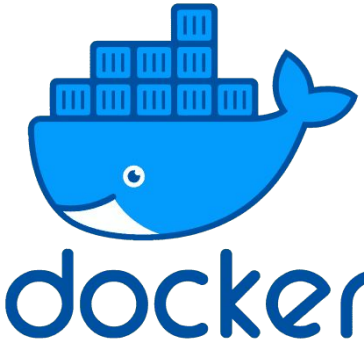


+

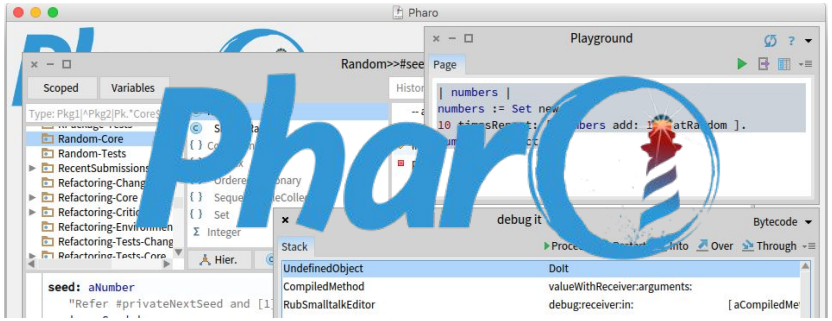
matter.js

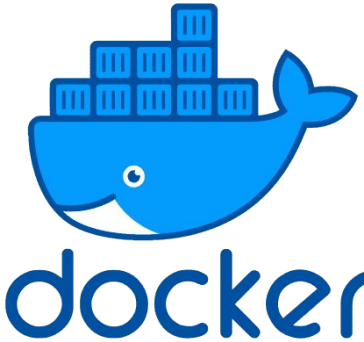
PLC3000.com Server





PLC3000.com Server

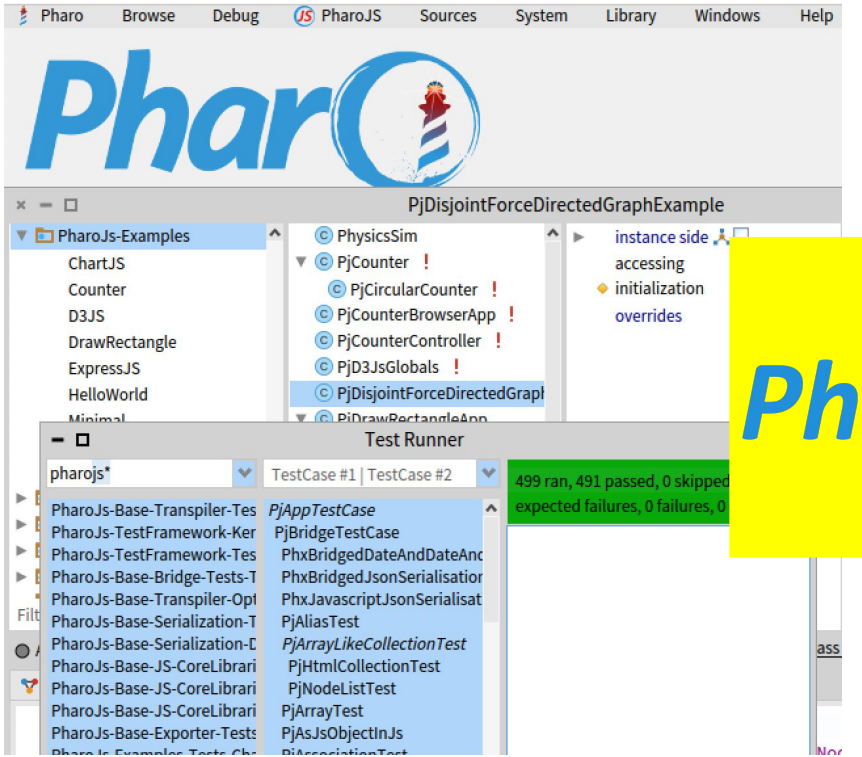




Thank you
Buenos Aires
Smalltalk

Thank you
Gabriel Cotteli

JS Server Generation?



PharJS

ExpressJS

node
JS

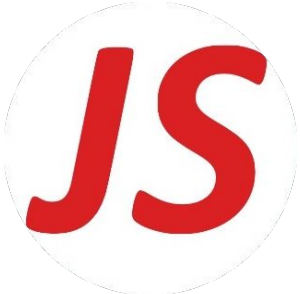
Example: Minimal Web Server

ExpressJS

- GET /data = read a string from memory
- POST /data = save a string to memory

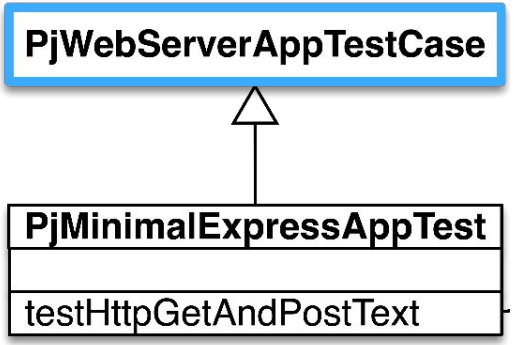
Pharo 100%

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

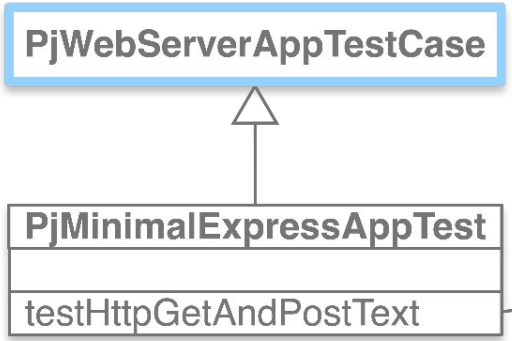


100% Javascript

Testing Minimal Web Server with PharJS

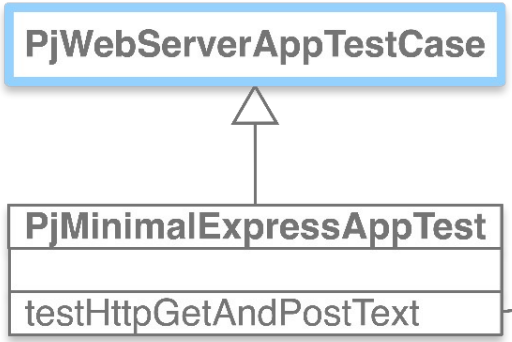


Testing Minimal Web Server with PharJS




```
testHttpGetAndPostText
    | response |
    self assert: (self get: '/data') contents isNil.
    response := self post: '/data' text: 'PharoJS'.
    self assert: response code equals: 200.
    self assert: (self get: '/data') contents equals: 'PharoJS'
```

Testing Minimal Web Server with PharJS

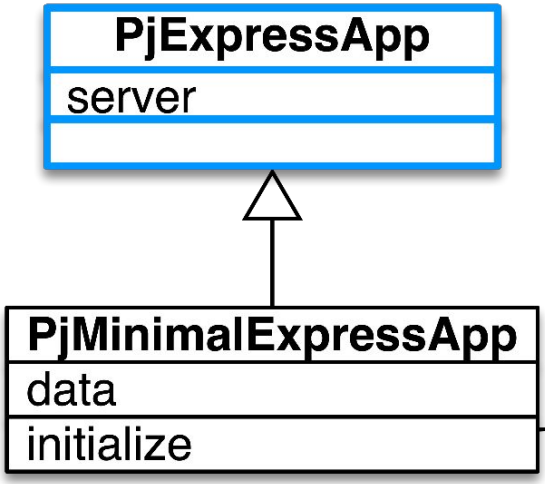



```
testHttpGetAndPostText
    | response |
    self assert: (self get: '/data') contents isNil.
    response := self post: '/data' text: 'PharoJS'.
    self assert: response code equals: 200.
    self assert: (self get: '/data') contents equals: 'PharoJS'
```

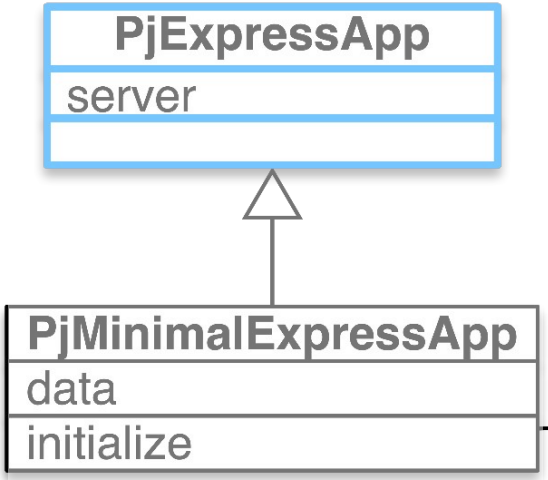
Relies on **Zinc HTTP**

Thank you Sven

Minimal Web Server with PharJS

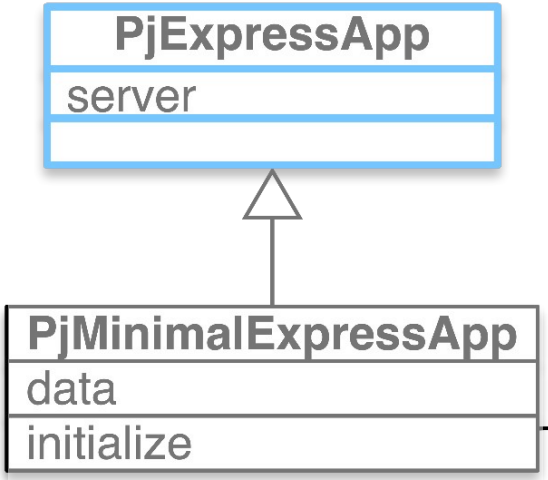


Minimal Web Server with PharJS



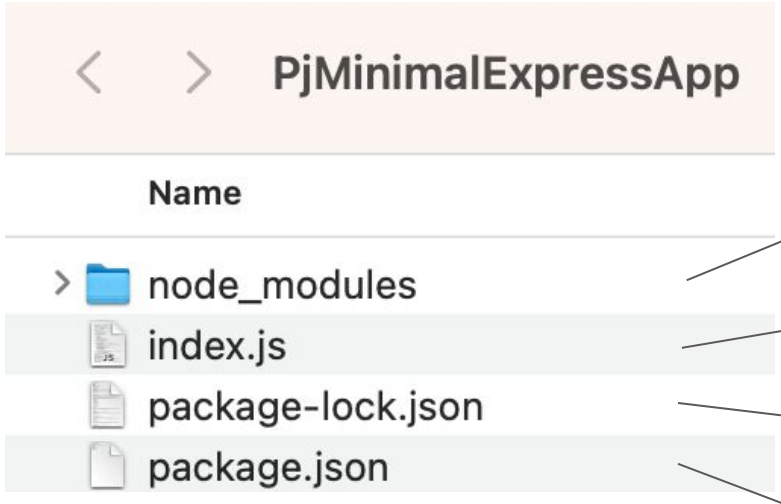

```
initialize
    super initialize.
    server
        get: '/data'
        handler: [ :req :res | res send: data].
    server
        post: '/data'
        handler: [ :req :res |
                    data := req body.
                    res send: data ].
```

Minimal Web Server with PharJS



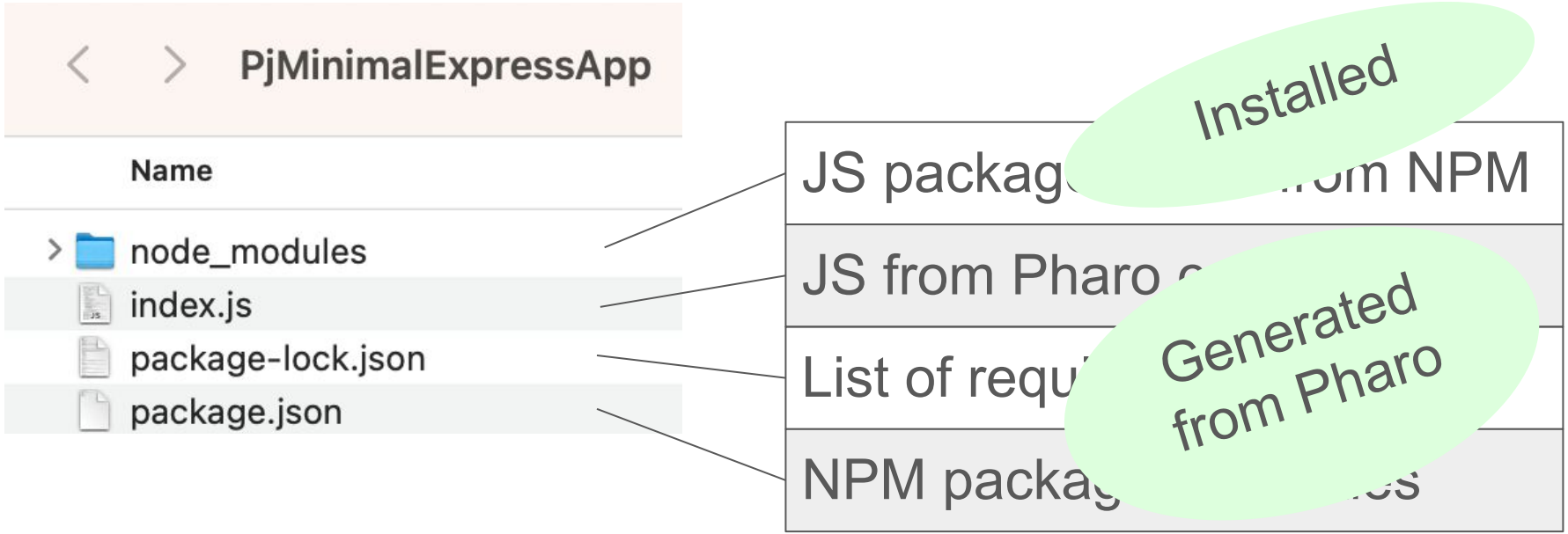

```
initialize
    super initialize.
    server
        get: '/data'
        handler: [ :req :res | res send: data].
    server
        post: '/data'
        handler: [ :req :res |
                    data := req body.
                    res send: data ].
```

All JS Code is Managed by PharJS



- JS packages code from NPM
- JS from Pharo code
- List of required JS packages
- NPM package properties

All JS Code is Managed by PharJS

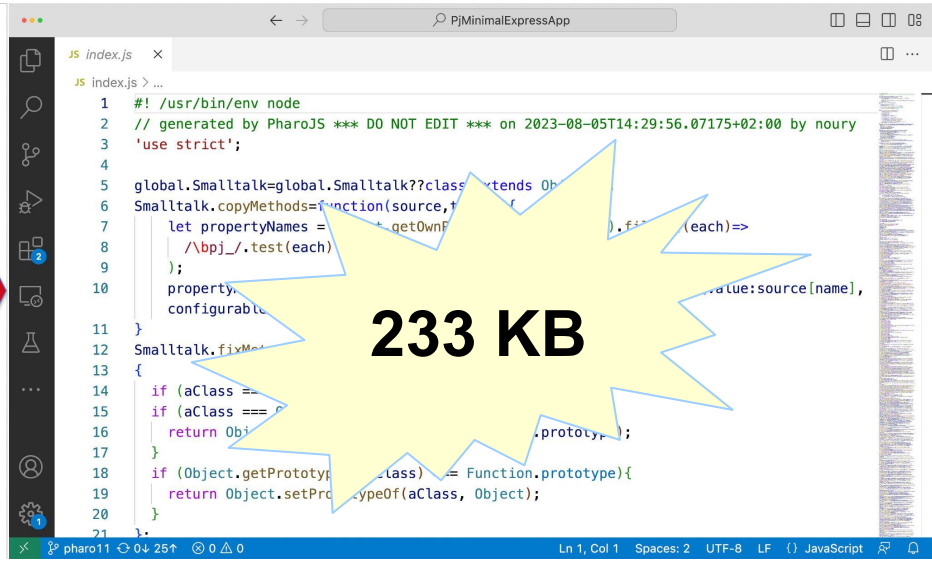


Server JS Code Generated with PharoJS

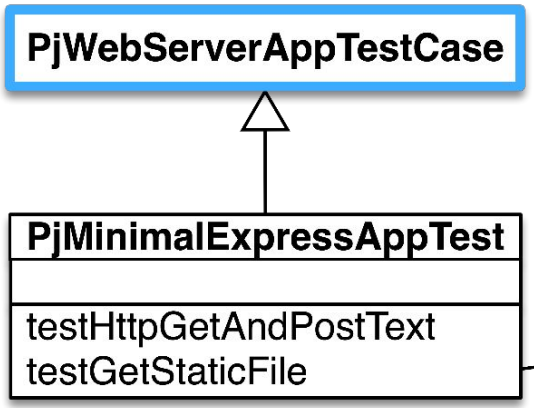
Pharo 100%

Development → Production

JS 100% Javascript

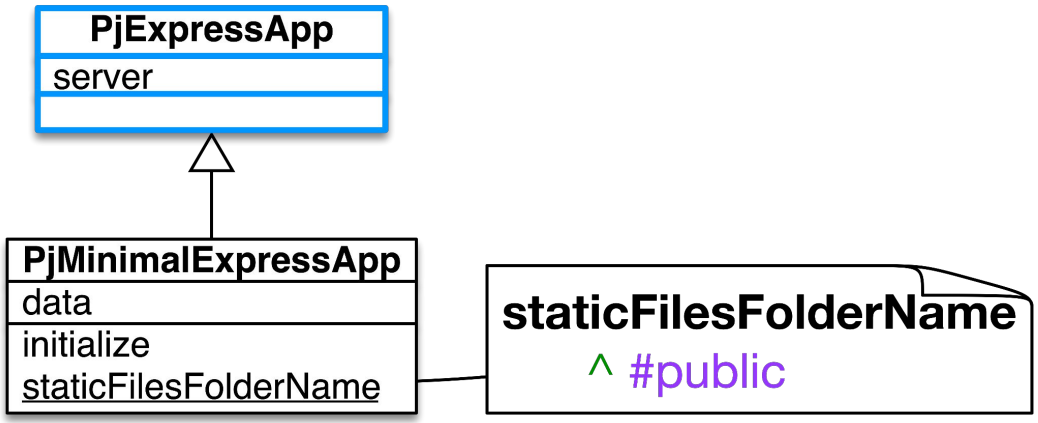


Testing Static File Serving with PharJS

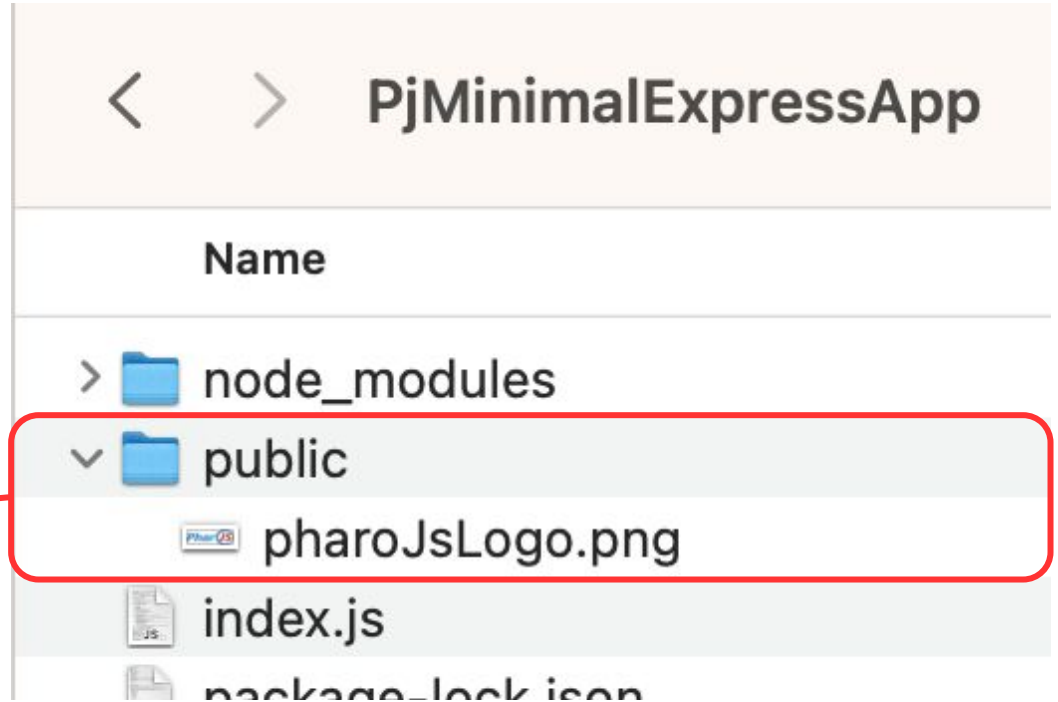



```
testGetStaticFile
    | response |
    response := self get: '/pharoJsLogo.png'.
    self assert: response code equals: 200.
    self assert: response contents size equals: 31251
```

Serving Static Files with PharJS



Serving Static Files with PharoJS

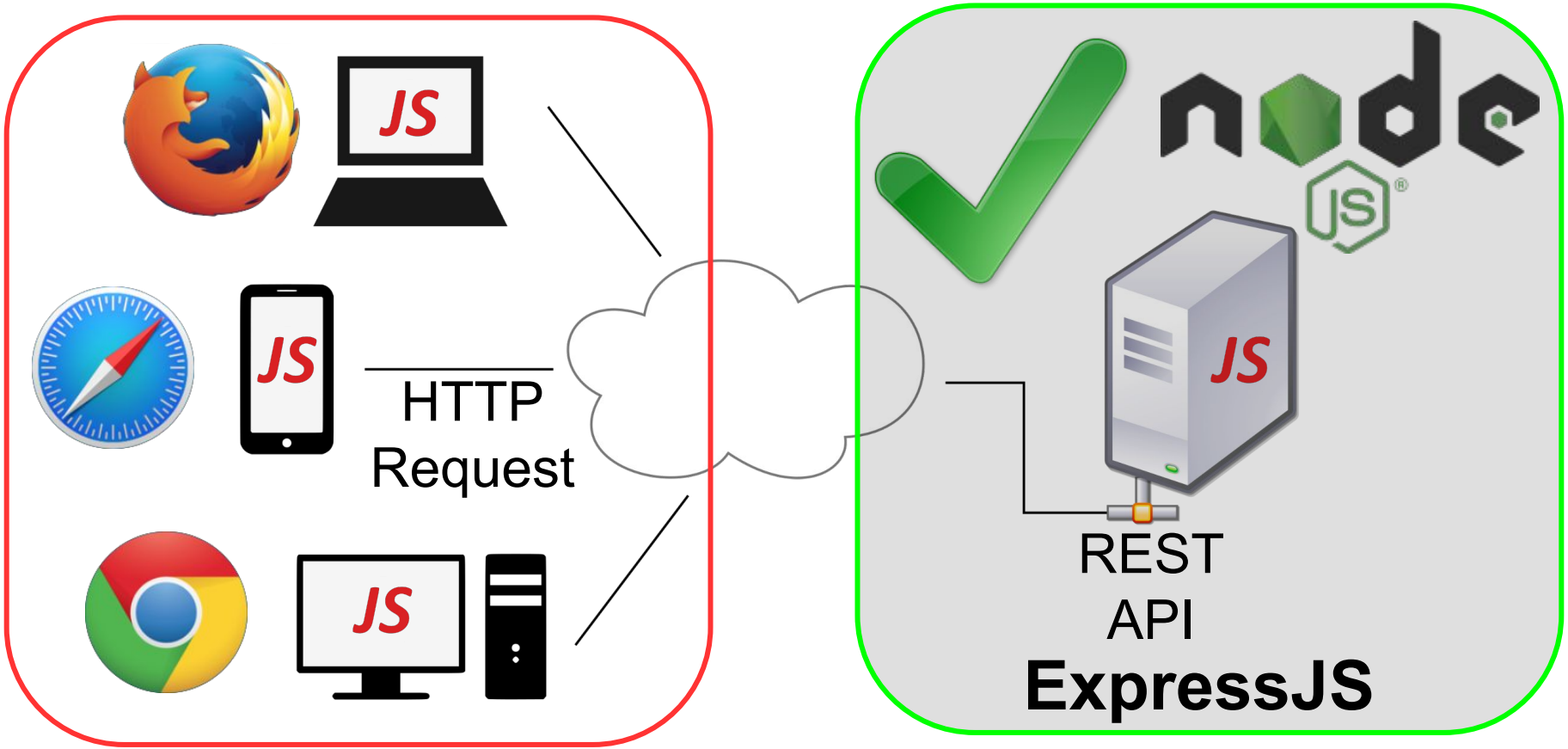


PjExpressApp
server

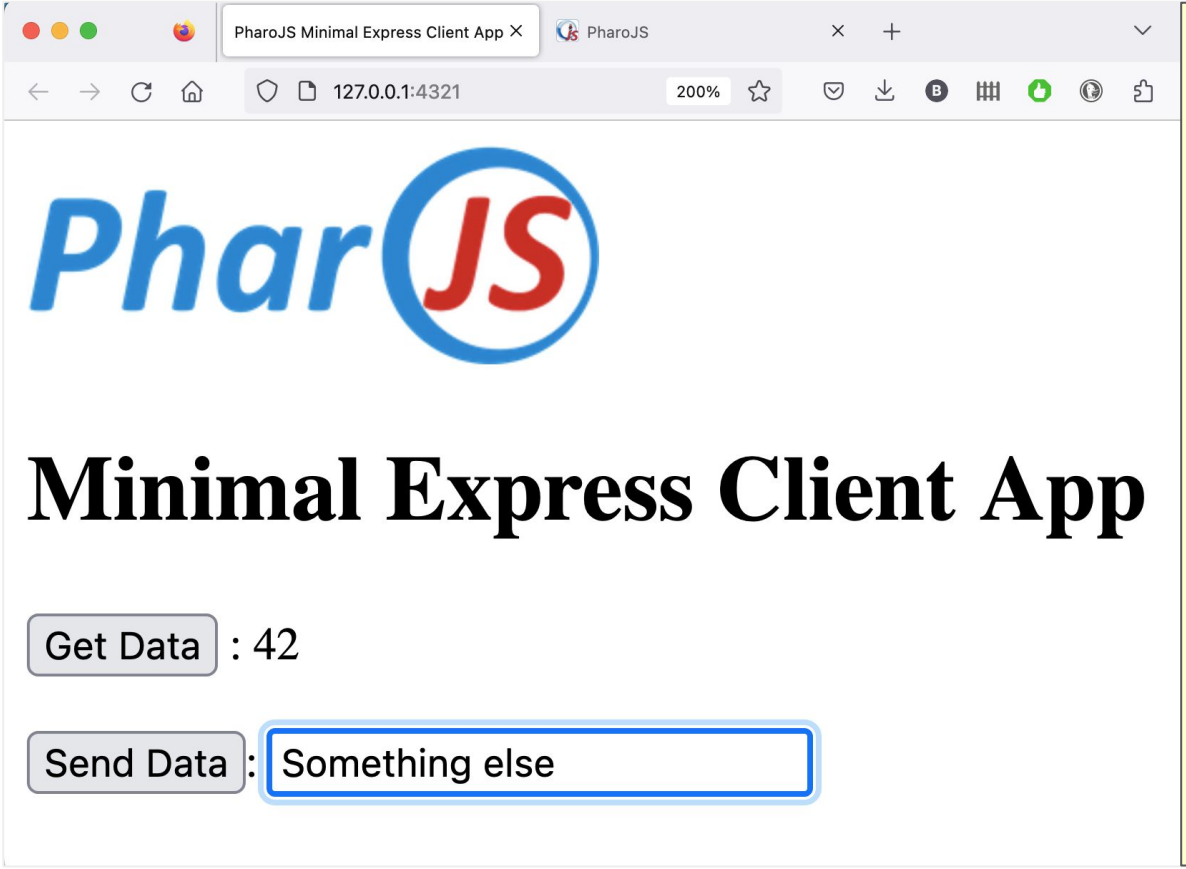
PjMinimalExpressApp
data
initialize
staticFilesFolderName

staticFilesFolderName
^ #public

Let's Address the Client



Web Client HTML



index.html

```
<img…>
<h1>...</h1>
<button…>...</button>
<span…>...</span>
<button…>...</button>
<input …>
<script …></script>
```

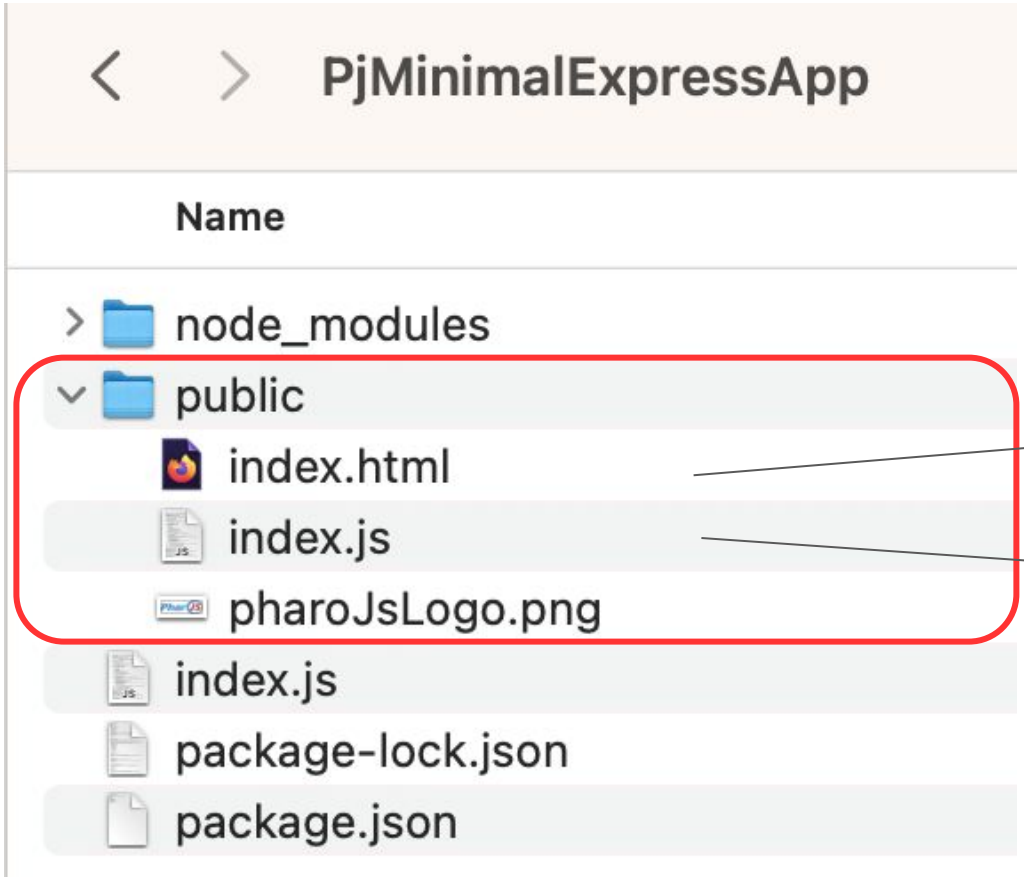
Html Loads JS Generated by PharJS

```
<html>
<body>
    <img src="pharoJsLogo.png">
    <h1>Minimal Express Client App</h1>
    <button id="getDataButton">Get Data</button> :
    <span id="dataDisplaySpan"></span>
    <br><br>
    <button id="postDataButton">Send Data</button> :
    <input type="text" id="dataInput">
    <script src="index.js"></script>
</body>
</html>
```

Html IDs for Integration with PharJS

```
<html>
<body>
    <img src="pharoJsLogo.png">
    <h1>Minimal Express Client App</h1>
    <button id="getDataButton">Get Data</button> :
    <span id="dataDisplaySpan"></span>
    <br><br>
    <button id="postDataButton">Send Data</button> :
    <input type="text" id="dataInput">
    <script src="index.js"></script>
</body>
</html>
```

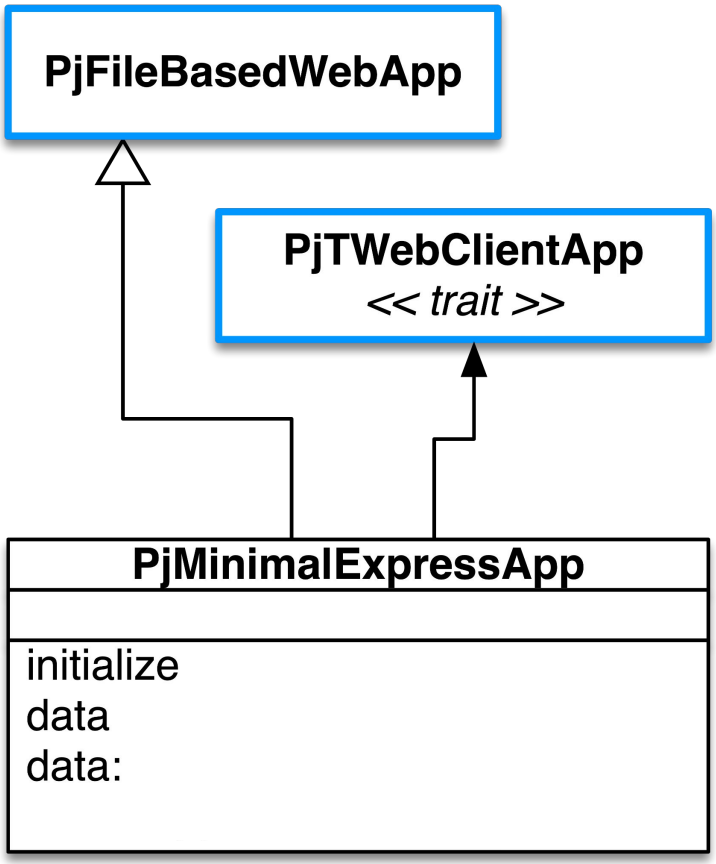
Web Client HTML & JS Served Statically



Client HTML file

Client JS from Pharo code

Web Client Pharo Code with PharJS




```
initialize
    super initialize.
    self
        onClick: #getDataButton
        do: [ self get: '/data' withTextDo: [ :text | self data: text ] ].
    self
        onClick: #postDataButton
        do: [ self post: '/data' text: self data ]
```

```
data
    ^ (self elementAt: #dataInput) value
```

```
data: aString
    (self elementAt: #dataDisplaySpan) textContent: aString
```

PharJS Talks to Server

```
initialize
    super initialize.
    self
        onClick: #getDataButton
        do: [ self get: '/data' withTextDo: [ :text | self data: text ] ].
    self
        onClick: #postDataButton
        do: [ self post: '/data' text: self data ]
```

```
data
    ^ (self elementAt: #dataInput) value
```

```
data: aString
    (self elementAt: #dataDisplaySpan) textContent: aString
```

PharJS Links DOM Objects using HTML IDs

```
initialize
    super initialize.
    self
        onClick: #getDataButton
        do: [ self get: '/data' withTextDo: [ :text | self data: text ] ].
    self
        onClick: #postDataButton
        do: [ self post: '/data' text: self data ]
```

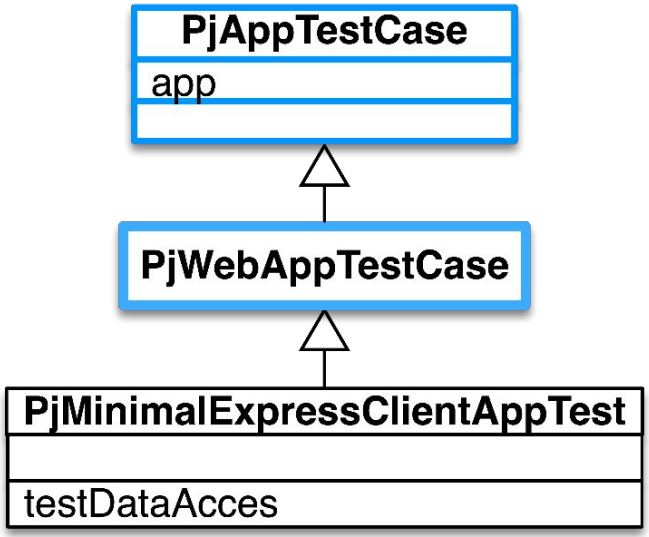
```
data
    ^ (self elementAt: #dataInput) value
```

```
data: aString
    (self elementAt: #dataDisplaySpan) textContent: aString
```

Web Client Test with PharoJS

Instance of
PjMinimalExpressClientApp

JS Object
in the Web Browser



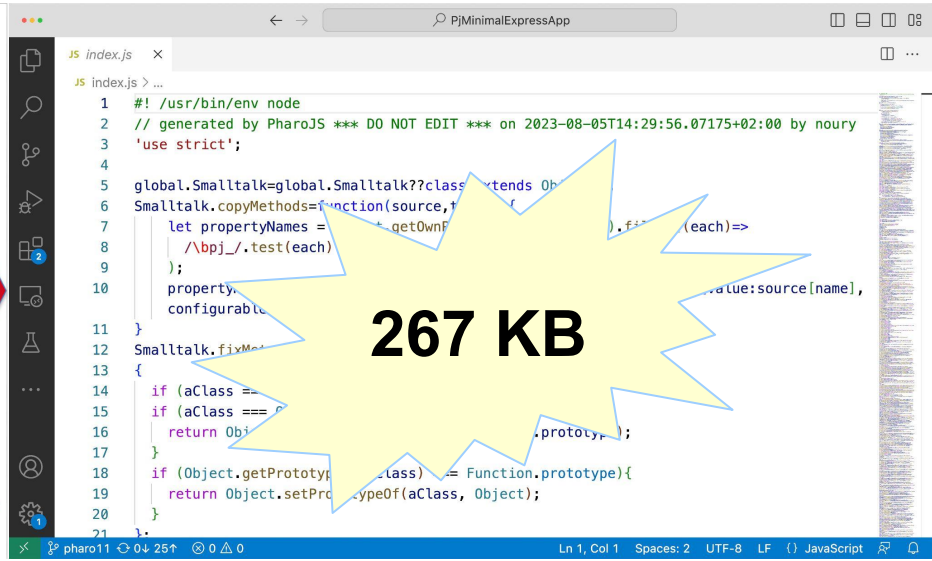

```
testDataAcces
    | display |
    display := app elementAt: #dataDisplaySpan.
    self assert: display textContent isEmpty.
    (app elementAt: #dataInput) value: '42'.
    self clickElementById: #postDataButton.
    self clickElementById: #getDataButton.
    self waitUntil: [ display textContent = '42' ]
```

Client JS Code Generated with PharJS

Pharo 100%

Development → Production

JS 100% Javascript



Summary

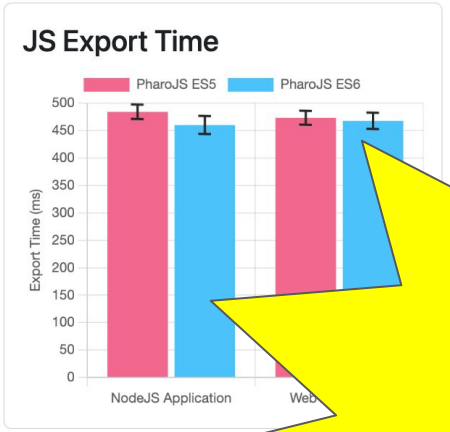
PharJS for both Server & Client Sides

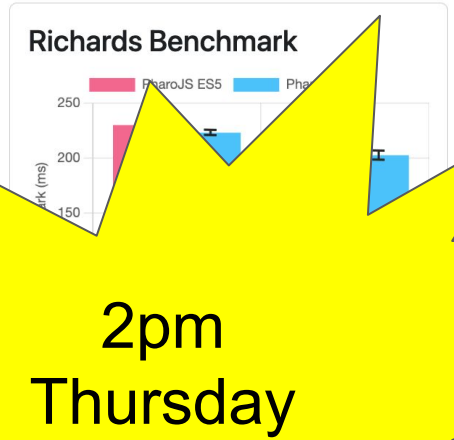
- Reuse JS Libraries & Run-Time
- Write 100% Pharo Code
- Test + Debug in Pharo with JS Objects
- Support Different Architectures & Workflows

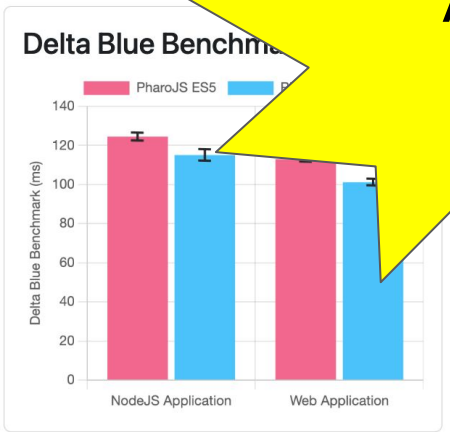
Other News of PharJS

- Current Stable = Pharo 11
- Continuous Integration
  - SmalltalkCI
  - GitHub Actions
- Improved Performance

IWST Talk on Performance of PharJS







2pm
Thursday
August 31st

Papa Parse

CSV Parser JS Lib

Chart.js

Charting JS Lib

MIT License

Kindly Supported by

EUROPEAN SMALLTALK USER GROUP
esug
www.esug.org

PharoJS.org

Develop in Pharo, Run on JavaScript

GitHub

Thanks to all the contributors!