Willow

The interaction tour

sponsored by mercap

Hello!

I am Gabriel Cotelli

Smalltalker since 2004

B.Comp. at FCEN - UBA

R&D at Mercap Software

@gcotelli

gcotelli@gmail.com

Agenda

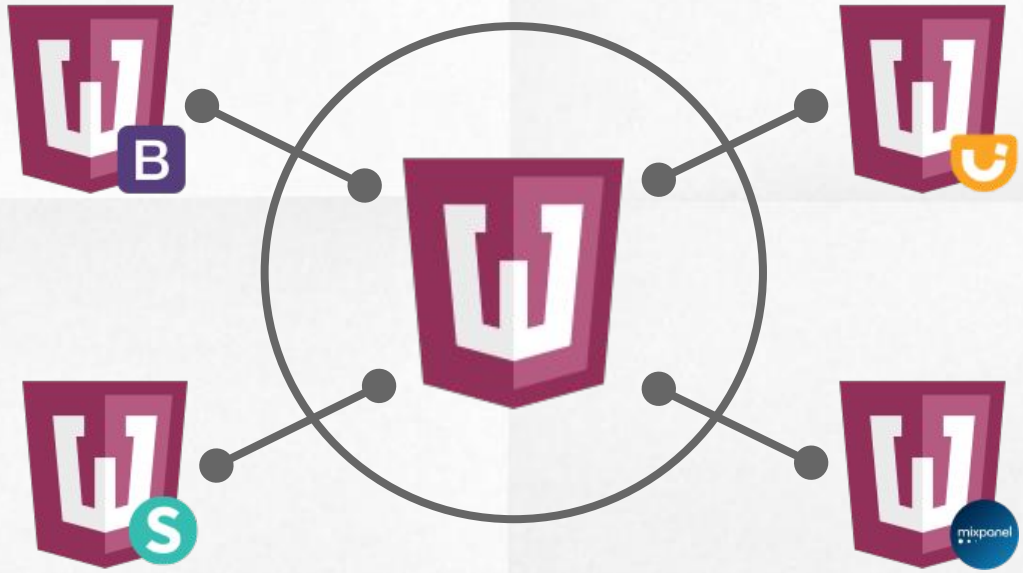
- Intro
- Components
- Interaction affordances
- Real life examples
- Complex interactions
- Conclusion

Intro

What is Willow?

Is a Web Interaction Library that eases the burden of creating AJAX-based web applications...

… conforming an ecosystem with specific integrations.



Why Willow?

?

Why Willow?

So that people can stop writing things like these:

```
html anchor
   id: 'idAjaxAnchor';
   onClick:
      (html jQuery
         script: [ :script |
            script
               << (JSStream on: 'this.onclick = function(){ return false; }').
            script
               <<
                  (html jQuery ajax
                     callback: [ flagDialog := false.
                        [ self setupVoceAttiva: each ]
                           on: Error
                           do: [ :ex | flagDialog := true ] ];
                     onSuccess:
                        (html jQuery ajax
                           script: [ :s |
                              flagDialog
                                 ifTrue: [ s
                                    << (s jQuery: #dialogId) dialog open ] ]))
   with: [ self renderItem: each on: html ]
```

Why Willow?

And start writing like these:

```
| anchor |

anchor := self componentSupplier
   asynchronicLinkLabeled: [ :html | self renderItem: each on: html ]
   applying: [  ].
anchor onTrigger
   determineBehaviorByEvaluating: [ :response |
      [ self setupVoceAttiva: each ]
         on: Error
         do: [ :ex | response onReturn open: self dialog ] ]
```

Features

Ready to use interactive components

Interactive components abstracts away a lot of the complexity of hand made AJAX calls.

Frontend integration

Component creation can be abstracted by using a component supplier, supporting:

- Plain HTML5
- Bootstrap
- Semantic UI
- JQuery UI

Seaside compatible

Built on top of Seaside rendering capabilities.

Seaside code can be mixed with Willow components.

Features

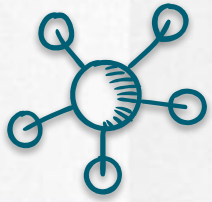
Ready to use interactive components

Interactive components abstracts away a lot of the complexity of hand made AJAX calls.

Frontend integration

Component creation can be abstracted by using a component supplier, supporting:

- Plain HTML5
- Bootstrap
- Semantic UI
- JQuery UI



Seaside compatible

Built on top of Seaside rendering capabilities.

Seaside code can be mixed with Willow components.

Features

Ready to use interactive components

Interactive components abstracts away a lot of the complexity of hand made AJAX calls.

Frontend integration

Component creation can be abstracted by using a component supplier, supporting:

- Plain HTML5
- Bootstrap
- Semantic UI
- JQuery UI

Seaside compatible

Built on top of Seaside rendering capabilities.

Seaside code can be mixed with Willow components.

Components

Fields

- Text Field
- Date Field
- Number Field

Selection

- Listbox
- Dropdown List
- Radio Groups
- Checkbox

Command Input

- Buttons
- Links
- Dropdown menu

Containers

- Lists
- Tables
- Grids
- Dialogs
- Panels

Misc

- Images
- Identified Views
- Labeled Views
- Tabs/Pills
- Navigation

Advanced

- Periodically rendered
- Delayed
- Typeahead

Components

Fields

- Text Field
- Date Field
- Number Field

Selection

- Listbox
- Dropdown List
- Radio Groups
- Checkbox

Command Input

- Buttons
- Links
- Dropdown menu

Containers

- Lists
- Tables
- Grids
- Dialogs
- Panels

Misc

- Images
- Identified Views
- Labeled Views
- Tabs/Pills
- Navigation

Advanced

- Periodically rendered
- Delayed
- Typeahead

Components

Fields

- Text Field
- Date Field
- Number Field

Selection

- Listbox
- Dropdown List
- Radio Groups
- Checkbox

Command Input

- Buttons
- Links
- Dropdown menu

Containers

- Lists
- Tables
- Grids
- Dialogs
- Panels

Misc

- Images
- Identified Views
- Labeled Views
- Tabs/Pills
- Navigation

Advanced

- Periodically rendered
- Delayed
- Typeahead

Components

Fields

- Text Field
- Date Field
- Number Field

Selection

- Listbox
- Dropdown List
- Radio Groups
- Checkbox

Command Input

- Buttons
- Links
- Dropdown menu

Containers

- Lists
- Tables
- Grids
- Dialogs
- Panels

Misc

- Images
- Identified Views
- Labeled Views
- Tabs/Pills
- Navigation

Advanced

- Periodically rendered
- Delayed
- Typeahead

Components

Fields

- Text Field
- Date Field
- Number Field

Selection

- Listbox
- Dropdown List
- Radio Groups
- Checkbox

Command Input

- Buttons
- Links
- Dropdown menu

Containers

- Lists
- Tables
- Grids
- Dialogs
- Panels

Misc

- Images
- Identified Views
- Labeled Views
- Tabs/Pills
- Navigation

Advanced

- Periodically rendered
- Delayed
- Typeahead

Components

Fields

- Text Field
- Date Field
- Number Field

Selection

- Listbox
- Dropdown List
- Radio Groups
- Checkbox

Command Input

- Buttons
- Links
- Dropdown menu

Containers

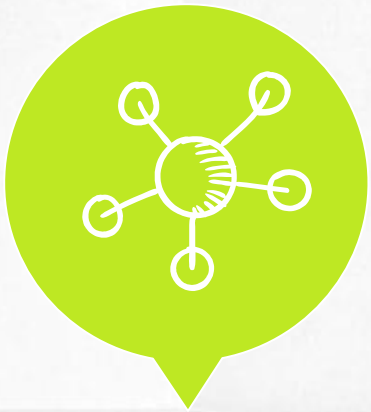
- Lists
- Tables
- Grids
- Dialogs
- Panels

Misc

- Images
- Identified Views
- Labeled Views
- Tabs/Pills
- Navigation

Advanced

- Periodically rendered
- Delayed
- Typeahead



Interaction affordances

Each interactive component supports the **#onTrigger** message

Client Execution

```
button onTrigger
    executeOnClient: [ :canvas |
        canvas javascript alert: 'Hi'
    ]
```

Client Execution

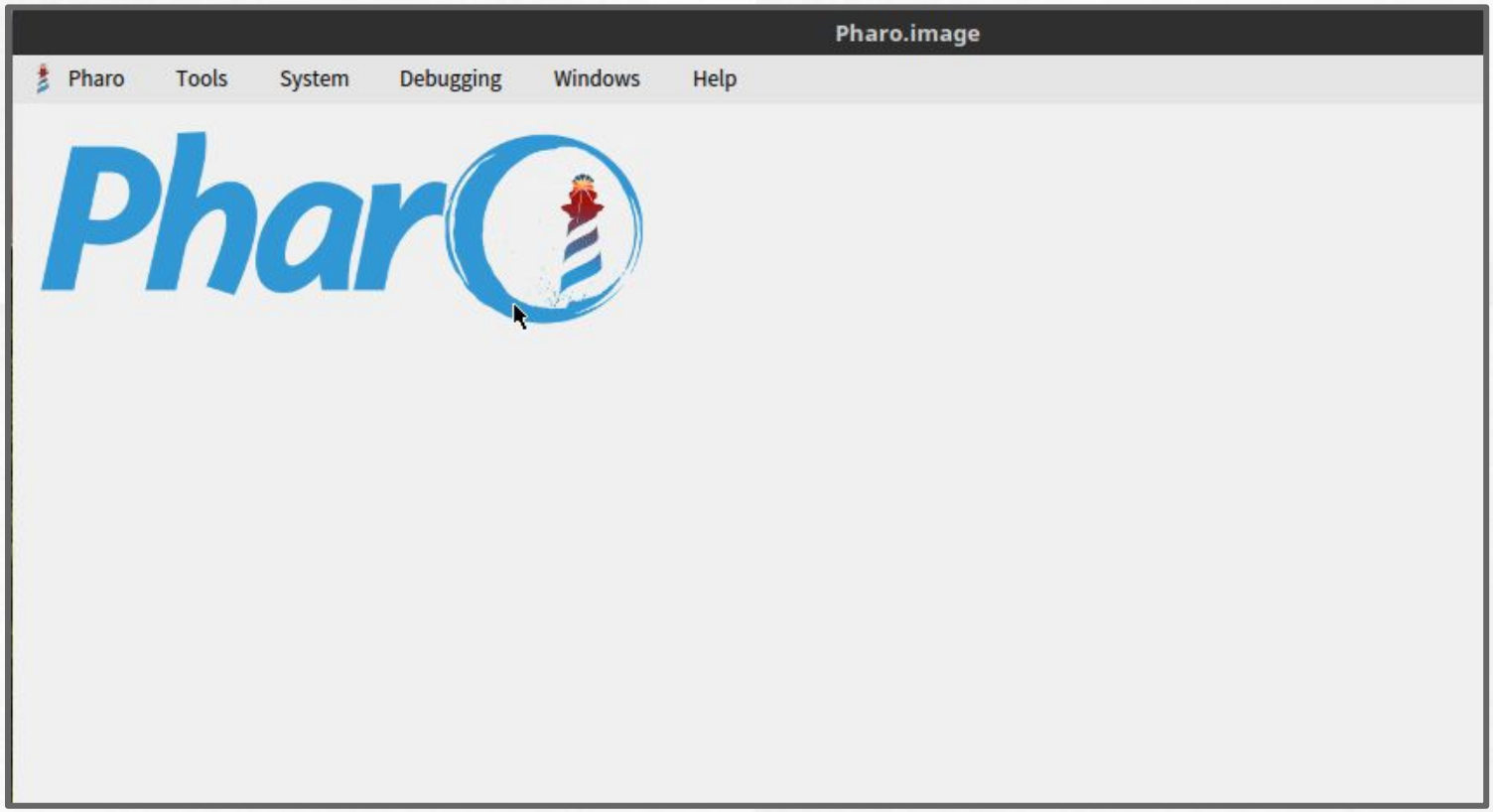
Client Execution

```
button onTrigger
        executeOnClient: [:canvas |
                canvas javascript alert: 'Hi'
        ]
```

Server Evaluation

```
button onTrigger
    evaluate: [
        'The button was clicked on the browser' inspect
    ]
```

Server Evaluation



Server Evaluation

Rendering

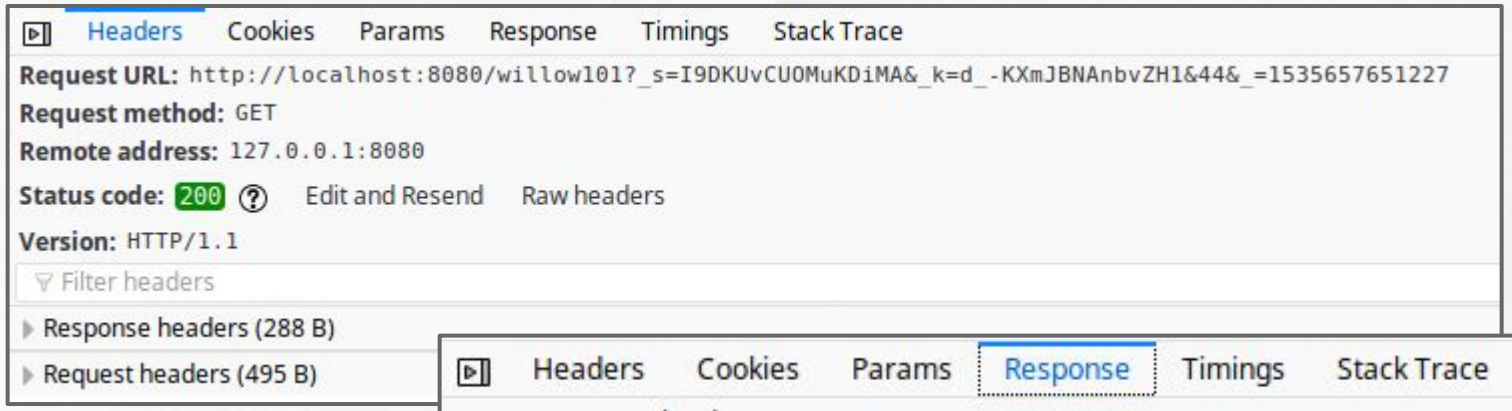
```
companionView := IdentifiedWebView
                forSpanNamed: 'Companion'
                containing: [ :canvas | canvas strong: self currentTime ].
button onTrigger render: companionView
```

Rendering

```
companionView := IdentifiedWebView
                forSpanNamed: 'Companion'
                containing: [ :canvas | canvas strong: self currentTime ].
button onTrigger render: companionView
```

Rendering


'button> event
click http://localhost:8080/willow101:1
jQuery
function(event) {
Willow.callServer({
"url": "/willow101",
"data": ["_s=I9DKUvCUOMuKDiMA", "_k=d_-KXmJBNAnbvZH1",
"21"].join("&")
})


Headers
Cookies
Params
Response
Timings
Stack Trace
Request URL: http://localhost:8080/willow101?_s=I9DKUvCUOMuKDiMA&_k=d_-KXmJBNAnbvZH1&44&_=1535657651227
Request method: GET
Remote address: 127.0.0.1:8080
Status code: 200
Edit and Resend
Raw headers
Version: HTTP/1.1
Filter headers
Response headers (288 B)
Request headers (495 B)
Headers
Cookies
Params
Response
Timings
Stack Trace
Response payload
1
$("#Companion-id46").html("16:58:33")

Serialization

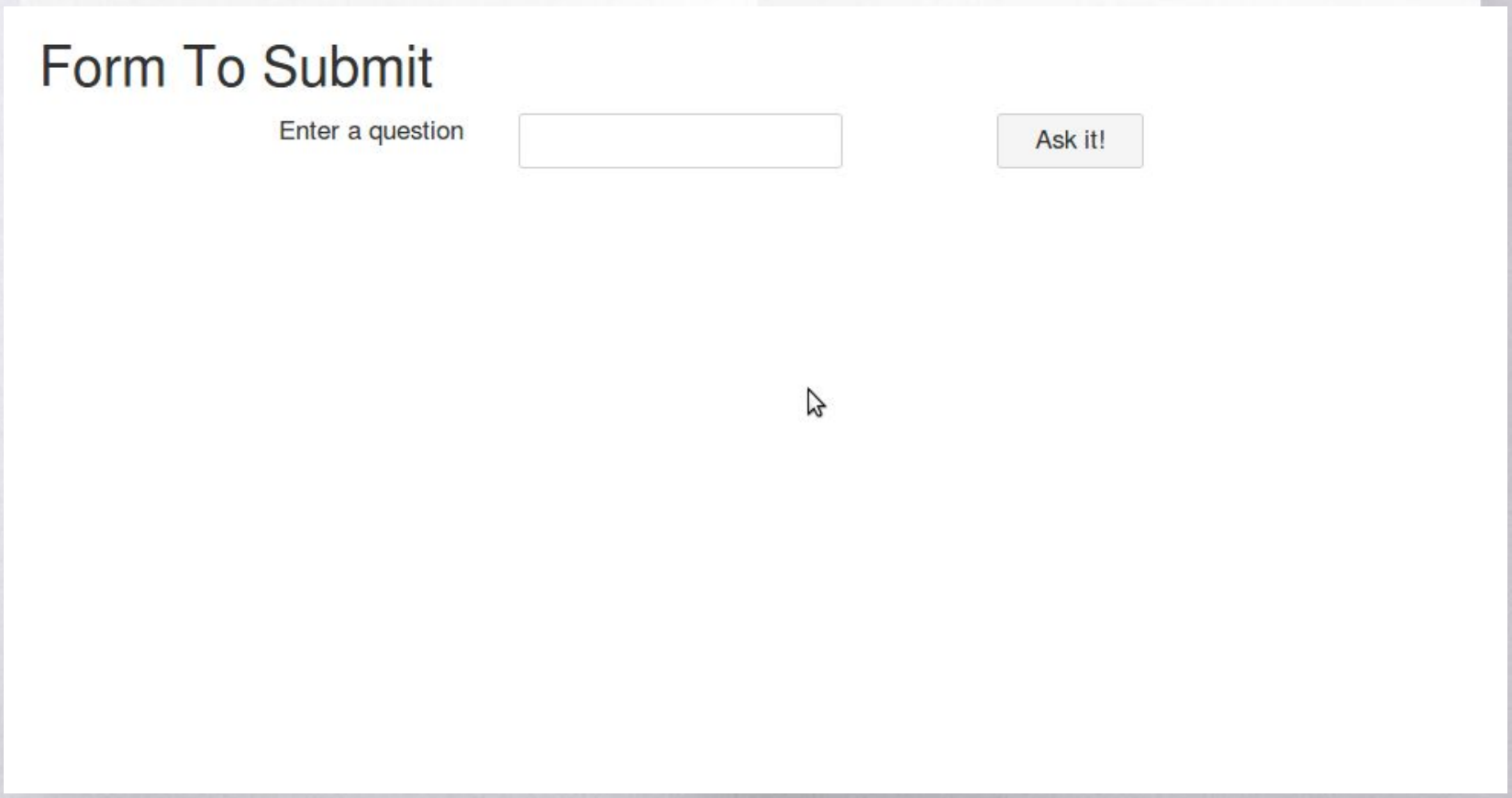
When working with form elements the contents must be serialized so that the server elements have the updated values. Here are some alternatives:

- serializeIt
- serializeWithHiddenInputs
- serializeForm:
- serializeContainerForm

Serialization

```
button onTrigger
        serializeContainerForm;
        disable;
        determineBehaviorByEvaluating: [ :response | | answer |
                answer := self askDeveloper: input contents.
                response onReturn
                        setValueTo: answer thenTriggerChangeOf: input ]
```

Serialization



Serialization

DOM Manipulation

Several affordances are available to manipulate DOM elements:

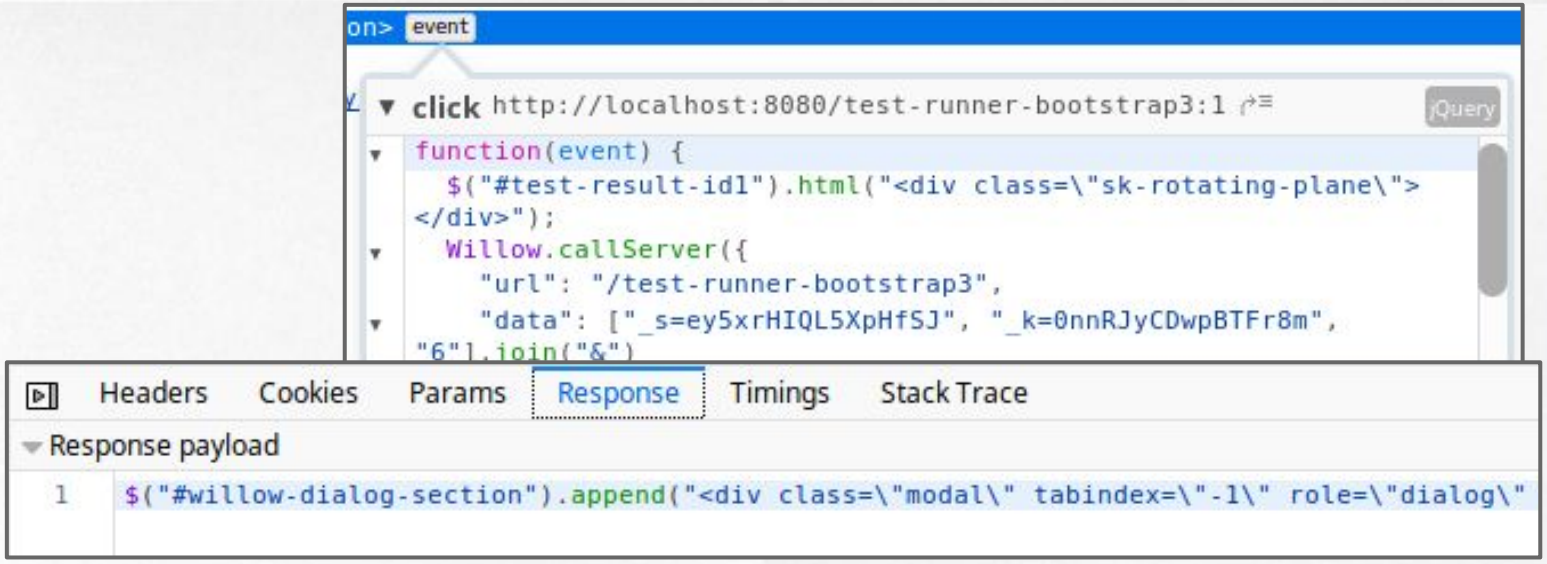
- disable/enable
- focus
- remove: identifiedView
- transform: identifiedView into: renderable
- setValueTo: val thenTriggerChangeOf: input
- show: renderable whileCallingServerToRender: identifiedView
- and more…

CSS Classes

- addCssClass:
- changeCssClass:to:
- removeCssClass:
- toggleCssClass:
- and some variations…

Dialogs

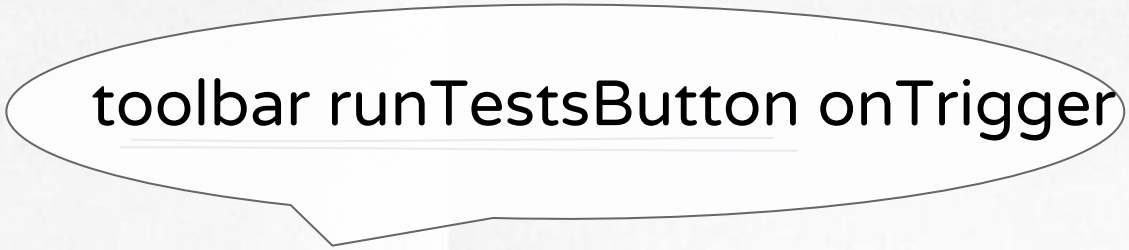
- closeLastDialog
- closeAllDialogs
- open: dialogView




Real life
examples

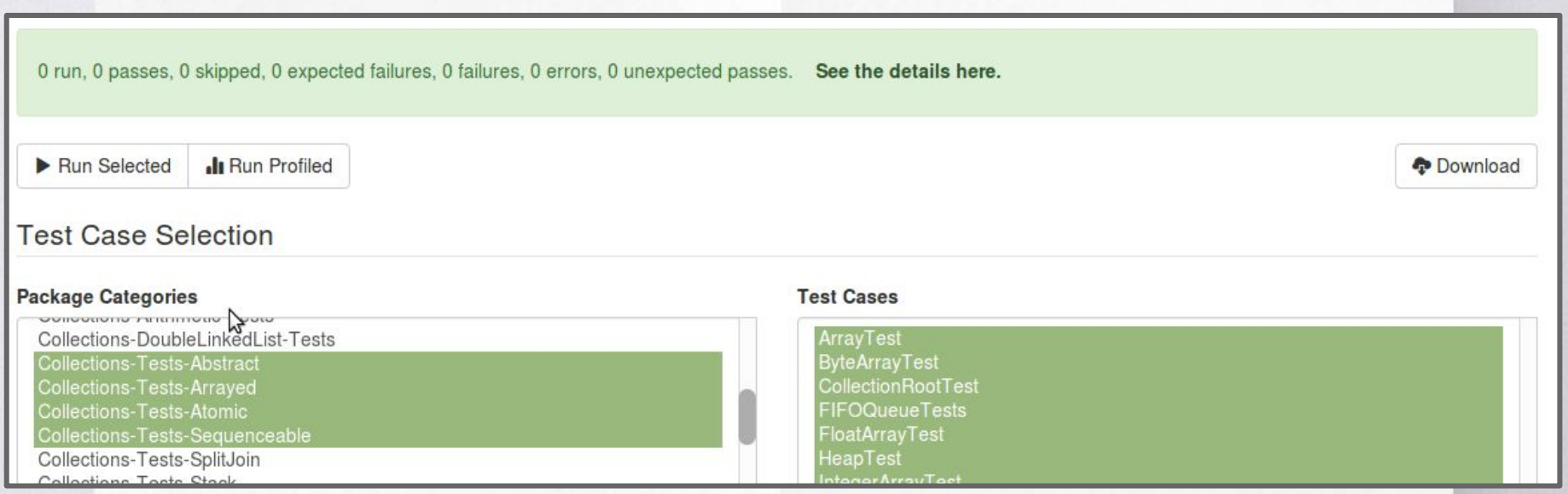
Combining Interactions

```
toolbar onRunSelectedTestsTrigger
    transform: testResult
    into: SpinKitTripleBounce new;
    evaluate: [ self session runTests ];
    render: testResult.
```




```
toolbar onRunSelectedTestsTrigger
    transform: testResult
    into: SpinKitTripleBounce new;
    evaluate: [ self session runTests ];
    render: testResult.
```

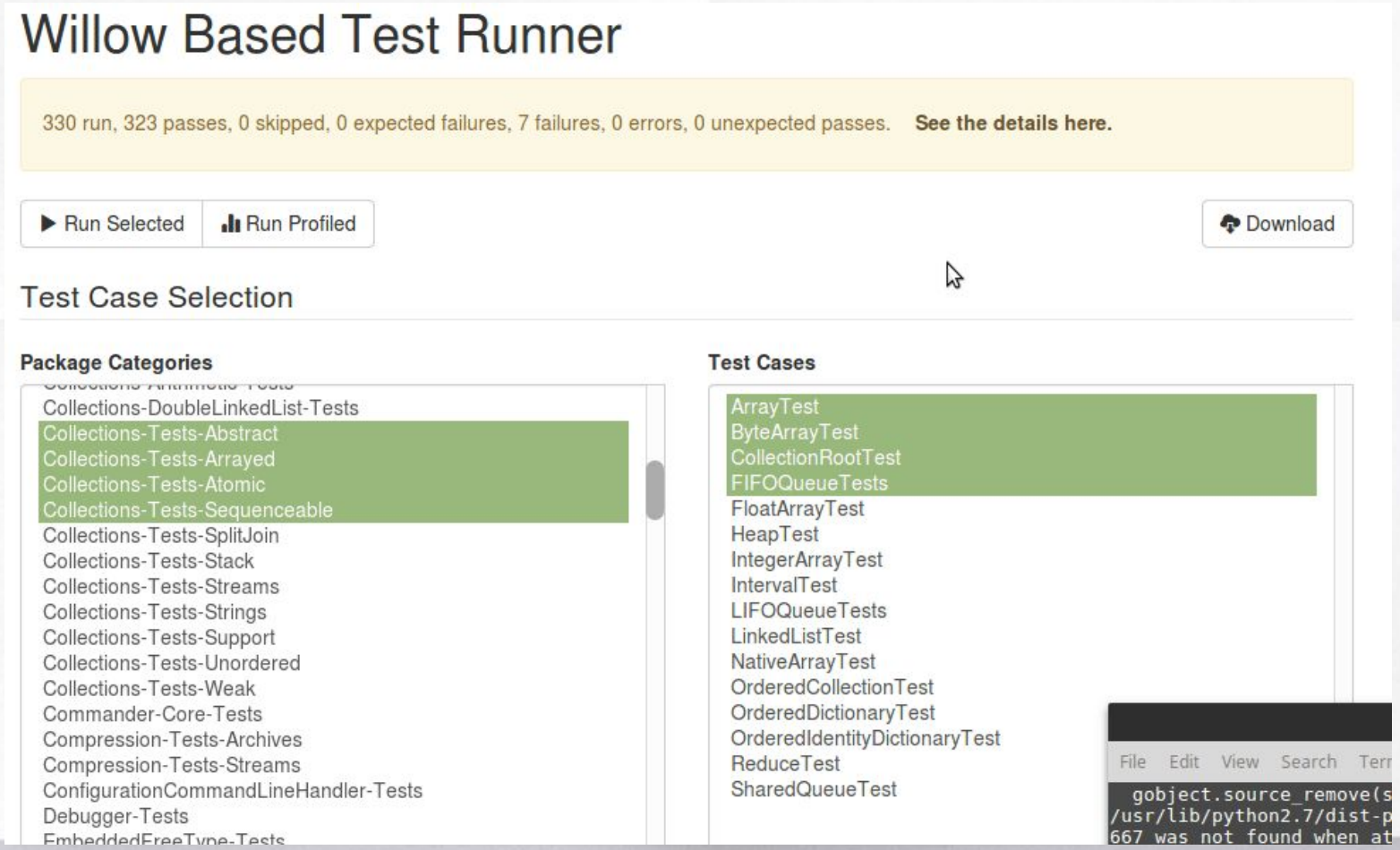
Combining Interactions



Dialog Opening

```
detailsLinkShowing: testResult
    | detailsAnchor |
    detailsAnchor := self componentSupplier
        asynchronicLinkActingAsButtonLabeled: 'See the details.'
        applying: [ :anchor | anchor addClass bootstrap alertLink ].
    detailsAnchor onTrigger open:
      (WPTestRunnerTestResultDetailsDialog for: testResult).
    ^ showDetailsAnchor
```

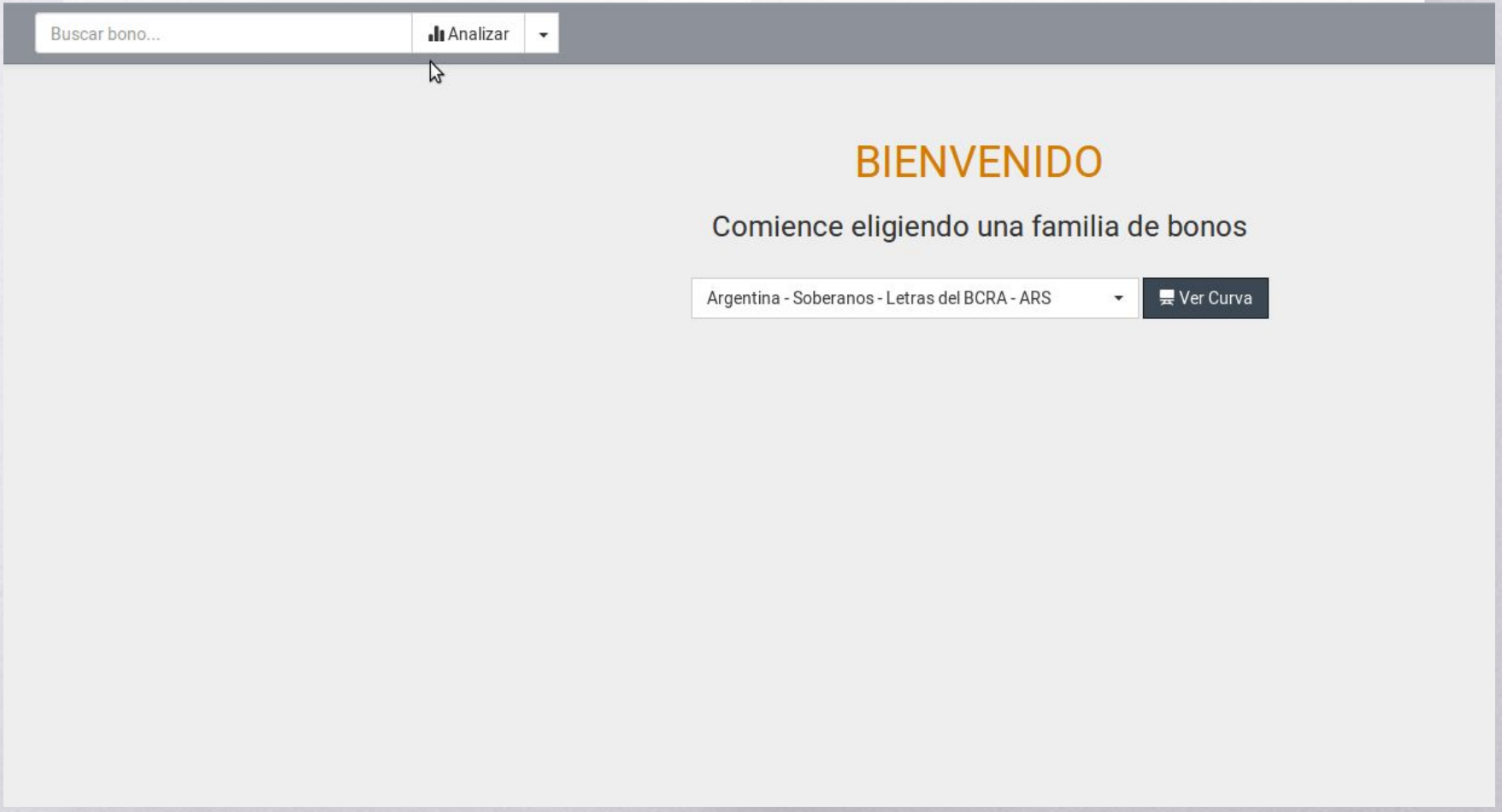
Dialog Opening



Conditional response

```
button onTrigger
    sendToMixpanel: [:mixpanel |
        mixpanel track: 'Clicked on "Search>Analyze"'];
    determineBehaviorByEvaluating: [:response |
        self applicationContext
            withBondUnderAnalysisDo: [:bond |
                response onReturn open:
                (BondInvestmentDialogWebView for: bond)]
            ifUnused: []].
```

Conditional response



Send To Mixpanel is just syntactic sugar

```
sendToMixpanel: aBlock

    ^self executeOnClient: [:canvas |
        | mixpanel |
        mixpanel := canvas mixpanel.
        aBlock cull: mixpanel cull: canvas.
        mixpanel]
```

Complex Interactions

(the crazy stuff)

Passing client parameters

```
| interpreter |
interpreter := WebInteractionInterpreter forInstantEvaluation.
userAgent := ''.
interpreter
    evaluate: [:clientUserAgent | userAgent := clientUserAgent]
    with: (JSStream on: 'navigator') access: 'userAgent'.
interpreter applyTo: aCanvas document on: aCanvas
```

Interaction with external JSObjects

```
interpreter := WebInteractionInterpreter
                    forEvaluationOf: #click: withAll: #().
interpreter determineBehaviorByEvaluating: [:response |
    self context setBondUnderAnalysisToBe: bond.
    response onReturn
        open: (BondInvestmentDialogWebView for: bond)].
events := HighchartsPlotOptionsScatterEvents new.
interpreter applyTo: events on: aCanvas.
^events
```

Subclass of JSObject

Interaction with external JSObjects

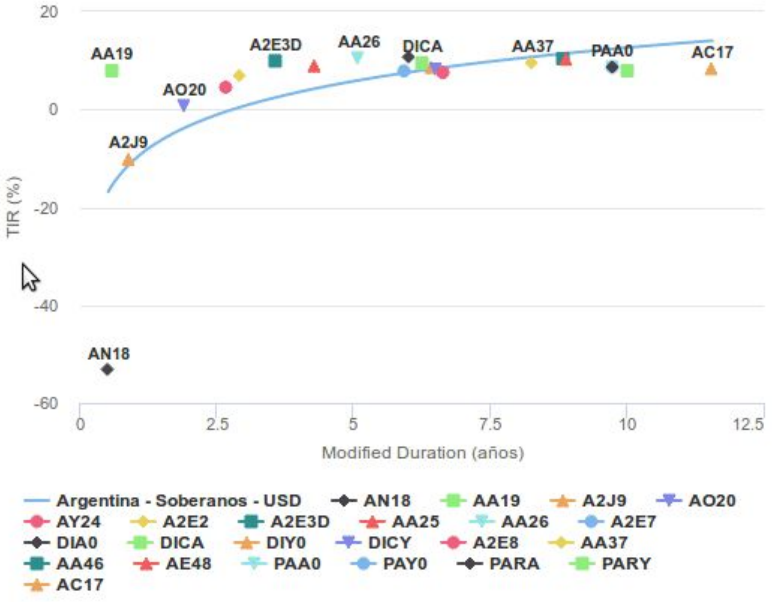
Buscar bono... | Analizar

Curvas / Argentina - Soberanos - USD

BONOS EN LA CURVA

Bono	TIR	MD	Paridad	VT	Cotización
A2E2	6.92 %	2.91	96.49 %	100.58	3,100.00 *
A2E3D	9.92 %	3.57	82.43 %	100.64	2,650.00 *
A2E7	7.85 %	5.95	94.97 %	100.71	3,055.00 *
A2E8	7.52 %	6.63	90.03 %	100.85	2,900.00 *
A2J9	-10.16 %	0.89	112.94 %	100.90	3,640.00 *
AA19	7.91 %	0.58	99.08 %	102.22	3,235.00 *
AA25	8.90 %	4.30	87.34 %	102.16	2,850.00 *
AA26	10.48 %	5.09	86.15 %	102.67	2,825.00 *
AA37	9.50 %	8.27	86.14 %	102.86	2,830.00 *
AA46	10.37 %	8.83	77.57 %	102.71	2,545.00 *

<< >>



Demo

Conclusions

- Simplified interactions by using a wide range of ready to use components
- Seaside compatible
- Advanced components available
- Complex interactions still possible

Acknowledgements

Special thanks to:

- Maxi Tabacman for starting this project in 2013
- Mercap Software for the financial support to be here today
- The Mercap development team and external collaborators for using and improving the library

Thanks!

Willow is MIT licensed and you can find it on /ba-st