Enhancing Dashboards with JavaScript!

Satoshi Kawasaki

Consultant, Splunk

splunk>

Disclaimer

During the course of this presentation, we may make forward looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC. The forward-looking statements made in the this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward looking statements we may make.

In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not, be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.

Who Am I?

Satoshi Kawasaki (hobbes3)

I'm a Ramblin' Wreck from Georgia Tech, and a hell of an engineer— (Aerospace)

but now I'm a consultant at Splunk after some dabbles in web development.

The Strategy

- Expectations
- Why JavaScript?
- Simple XML
- Tools + Tips
- Common Use Cases #1-#8
- References

Expectations

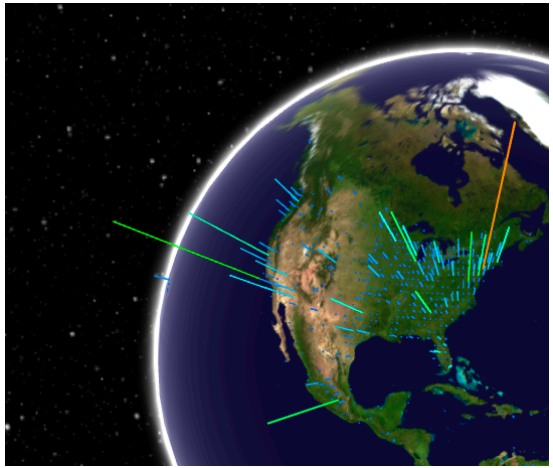
- I can’t teach you JavaScript (for free)
- You need to learn/explore on your own (or hire someone else)
- LMGTFY
- I don't support older Splunk versions (unless I'm paid)
- These slides (by itself) are terrible
- I will go FAST but you can always replay the recording at 0.5x
- This talk will probably only cover 1% of what SplunkJS and Javascript SDK can do

Why JavaScript?

- You’re not limited to Simple XML
- Client-side programming
- It's magic (to end users)!
- It makes bosses and executives happy!

Simple XML

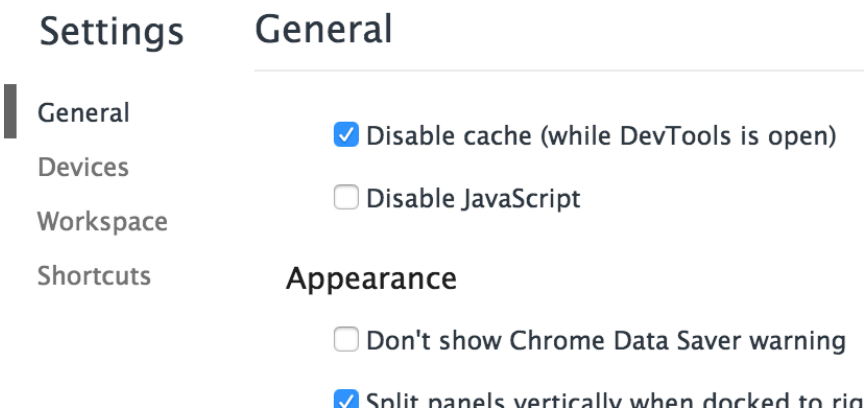
- You may not need JavaScript!
- Simple XML is getting more robust with every 6.x release
- Avoid JavaScript if possible (unless you want job security)
- You can even declare custom visualizations in only Simple XML!



```
<drilldown>
<selection>
<condition>
<change>
<chart depends="$foo$">
<set token="form.foo">
<unset token="cat">
<eval token="foo">
```

Splunk 6.x Dashboard Examples

Tools + Tip



- Chrome/Web Developer Tool
- Browser + Server caching
- Dev Splunk instance (restarts)
- web.conf

```
[settings]
cacheEntriesLimit = 0
minify_js = False
minify_css = False
```

Be the Explorer!

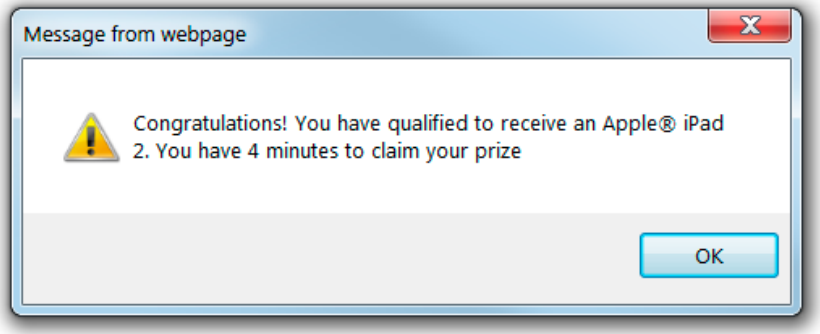
```
> splunkjs.mvc.Components.get("element1")
◂▾ child {_removed: false, id: "element1", name: "element
    ▸ $el: jQuery.fn.init[1]
      _checkExistingMarkup: false
    ▸ _debouncedOnEditModeChange: function ()
    ▸ _events: Object
    ▸ _listeningTo: Object
      _numConfigureCalls: 1
      _removed: false
      _tokenDepsMet: true
      cid: "view2198"
    ▸ contentLoadedDfd: Object
    ▸ el: div#element1.dashboard-element.html.splunk-view
      id: "element1"
    ▸ model: BaseModel.extend.constructor
      name: "element1"
    ▸ options: Object
    ▸ reportReady: Object
    ▾ settings: child
        _applyTokensByDefault: false
      ▸ _bindings: Object
        changing: false
```

And read the docs... especially example code!

Use Case #1: "Hello World"

```
<form script="their_app:their_script.js, foo.js">
```

```
alert("hello world!");
```

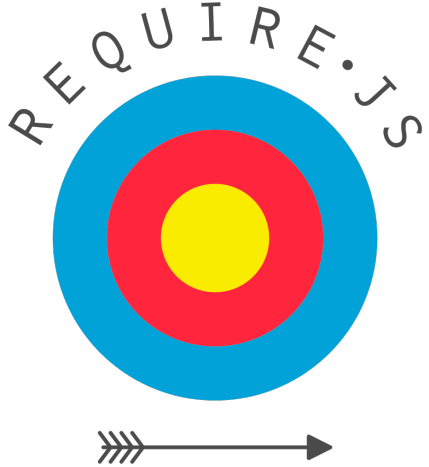


Put in `$SPLUNK_HOME/etc/apps/<my_app>/appserver/static/`

Requires a splunkweb restart to be read for the first time

Use Case #2: RequireJS

```
require([
    "jquery",
    "underscore",
    "splunkjs/mvc",
    "app/my_app/my_include",
    "splunkjs/mvc/simplexml/ready!"
], function(
    $,
    _,
    mvc,
    my_include
) {
    require(["splunkjs/ready!"], function() {
        // foo
    });
});
```



RequireJS is hard... (there, I said it)

Use Case #3: IDs

```
<chart id="my_chart">
<input type="text" token="foo" id="my_input">
<row id="my_row">
<search id="my_search">
```

```
mvc.Components.get("my_chart");
mvc.Components.get("my_search").settings.get("search");
$("#my_chart").slideDown();
```

Avoid this:

```
mvc.Components.get("element3");
$("#input23");
```

Use Case #4: Understanding $tokens$

- URL tokens
- form tokens
- default/unsubmitted tokens
- submitted tokens

.../app/my_app/my_dashboard?earliest=-24h%40h&latest=now&form.foo=bar

```
<input type="text" token="foo">
```

```
var unsubmitted_tokens = mvc.Components.get("default");
var submitted_tokens = mvc.Components.get("submitted");
```

Import "simple_xml_examples:showtokens.js" to understand the token madness!

Use Case #4 cont'd: Manipulating $tokens$

```
function submit_and_update_url() {
    submitted_tokens.set(unsubmitted_tokens.toJSON());
    mvc.Components.get("url").saveOnlyWithPrefix("form\\.", unsubmitted_tokens.toJSON(), {
        replaceState: false
    });
}
```

Setting and unsetting tokens:

```
unsubmitted_tokens.set("form.foo", "bar");
unsubmitted_tokens.unset("form.apple");
submit_and_update_url();
```

Listening to token changes:

```
submitted_tokens.on("change:foo", function() {
    // foo
});
```

WARNING!


Here be dragons!
Circulus Arcticus
TERRA
SAGUENA
CIA.
Norombega pars
OCEANUS
OCEANUS

Use Case #5: Searches and Jobs

Create a basic oneshot search:

```
var service = mvc.createService();

var search = "search index=_* | stats count by index sourcetype";
var search_params = {
    earliest_time: "-5m@m",
    latest_time: "now",
    count: 50
};

service.oneshotSearch(search, search_params, function(err, results) {
    var fields = results.fields;
    var rows = results.rows;
    var sourcetype = results.row[0][fields.indexOf("sourcetype")];
});
```

Soft limit of 100 rows; a hard limit of 50,000 rows!

If you need to retrieve a large data set, then use count+offset and "stitch" it back together (LMGTFY)

Use Case #5 Cont'd: Searches and Jobs

When a search completes:

```
var my_search = mvc.Components.get("my_search");

my_search.on("search:done", function(properties) {
    console.log("DONE!\nSearch job properties:", properties.content);
});
```

Accessing the results of a search:

```
var my_results = my_search.data("results", {count: 20});

my_results.on("data", function() {
    var rows = this.data().rows;
});
```

Use Case #6: Listeners

```
<option name="drilldown">all</option>
```

```
var my_timechart = mvc.Components.get("my_timechart");

my_timechart.on("click", function(e) {
    e.preventDefault();
});
```

Also there are:

- "selection"
- "click:chart"
- "click:legend"
- "click:marker"

```
> e
< ▼Object {field: "count", data: Object, event: Object}
    ▼data: Object
       click.name: "_time"
       click.name2: "count"
       click.value: "1441945500.000"
       click.value2: "1636"
       earliest: 1441945500
       latest: 1441945800
       row._span: 300
       row._time: "1441945500.000"
       row.count: "1636"
     ▶__proto__: Object
    ▶defaultPrevented: function ()
    ▶drilldown: function ()
    ▶event: Object
     field: "count"
    ▶preventDefault: function ()
```

Use Case #7: Render Interceptions

```
require([
    "jquery",
    "splunkjs/mvc",
    "splunkjs/mvc/tableview",
    "splunkjs/mvc/simplexml/ready!"
], function($, mvc, TableView) {
    var MyTableRenderer = TableView.BaseCellRenderer.extend({
        canRender: function(cell) { return cell.field === "foo"; },
        render: function($td, cell) {
            var field = cell.field;
            var value = cell.value;
            $td.text(value + "!");
        }
    });

    mvc.Components.get('my_table').getVisualization(function(tableView) {
        tableView.table.addCellRenderer(new MyTableRenderer());
        tableView.table.render();
    });
```

$SPLUNK_HOME/share/splunk/search_mrsparkle/exposed/js/splunkjs/mvc

Check more table examples at

Use Case #8: More JavaScript SDK

Get the current user:

```
service.currentUser(function(err, user) {
    console.log("Current user: ", user.properties().realname, " (" + user.name + ")");
});
```

- Can do a lot of (dangerous) tasks
- Manage configurations and objects (ie delete and modify objects)
- Log in
- Integrate Splunk results to your own application

References

- JavaScript SDK + Reference
- SplunkJS + Reference
- Simple XML
- Splunk 6.x Dashboard Examples
- Custom Visualizations
- Unofficial: My code snippets

Rachel Perkins
Chris Breshears

Genti Zaimi
Thomas Mann
Roy Moranz
Misty Gibbs

Itay Neeman
David Foster
Siegfried Puchbauer
Mathew Elting
Nick Filippi
Michael Porath
Sanford Owings
Vladimir Skyork
Alexander Johnson
Stephen Sorkin
Tom LaGatta

THANK YOU

splunk>