

Canary

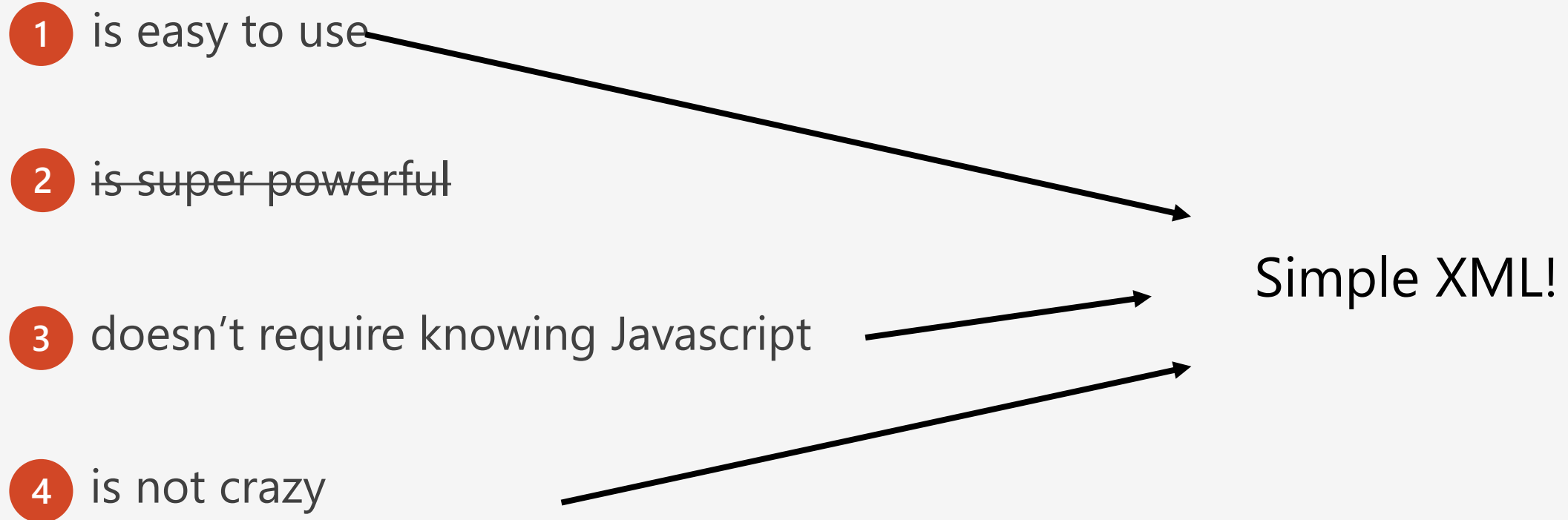
The craziest modular Splunk UI you've never seen



You need something that

- 1 is easy to use
- 2 is super powerful
- 3 doesn't require knowing Javascript
- 4 is not crazy

You need something that



You need something that

- 1 is easy to use
- 2 is super powerful
- 3 ~~doesn't require knowing Javascript~~
- 4 is not crazy

Simple XML
Extensions!

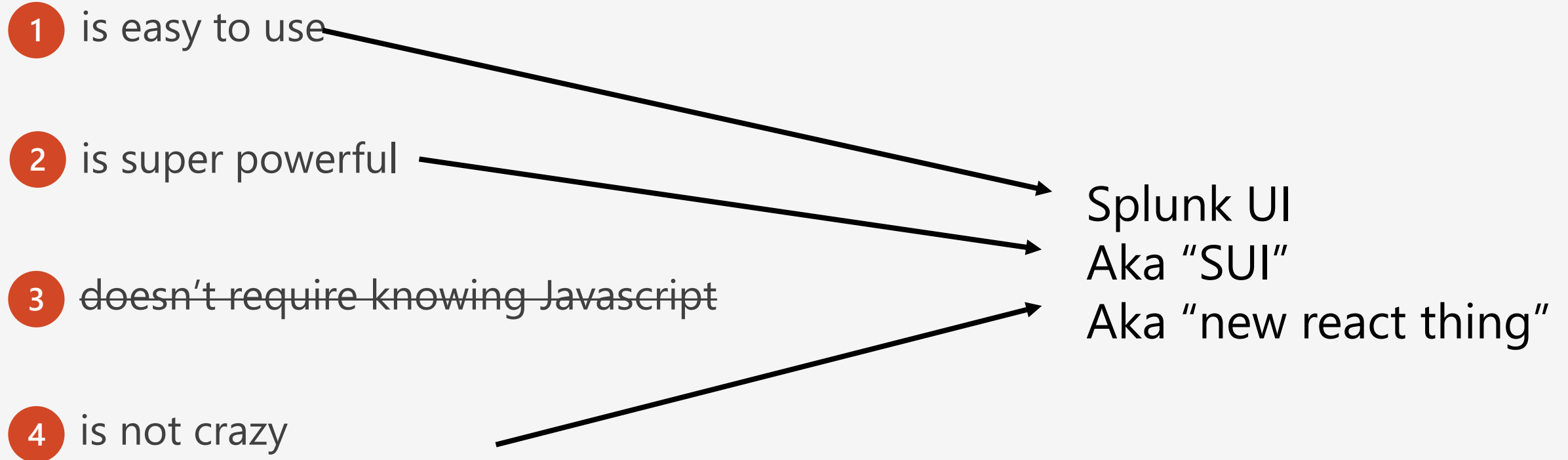
The diagram consists of four black arrows pointing from the list items on the left to the text 'Simple XML Extensions!' on the right. The first arrow points from '1 is easy to use' to the top of the text. The second arrow points from '2 is super powerful' to the middle of the text. The third arrow points from '3 doesn't require knowing Javascript' to the bottom of the text. The fourth arrow points from '4 is not crazy' to the bottom of the text.

You need something that

- 1 ~~is easy to use~~
- 2 ~~is super powerful~~
- 3 ~~doesn't require knowing Javascript~~
- 4 ~~is not crazy~~

Remember Django?

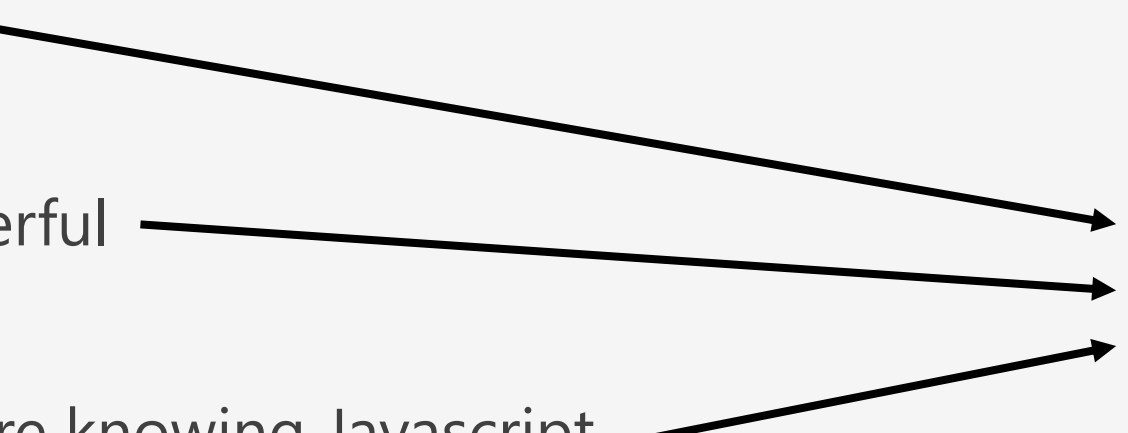
You need something that



You need something that

- 1 is easy to use
- 2 is super powerful
- 3 doesn't require knowing Javascript
- 4 is not crazy

You need something that

- 1 is easy to use
 - 2 is super powerful
 - 3 doesn't require knowing Javascript
 - 4 ~~is not crazy~~
- 
- Canary!

Wait... I was promised crazy. Where is the crazy?

There are some modules and patterns that themselves do surprising things.

1 **REST module.**

It's like plugging \$foo\$ tokens into a search, except instead of a Search they're going into the args for a POST to an arbitrary REST endpoint.

You know.. For kids!

Wait... I was promised crazy. Where is the crazy?

There are some modules and patterns that themselves do surprising things.

- 1 REST module.
- 2 **Multiplexer module.**

It's a little like the much bigger brother of "trellis".

For each row, make a new copy of <Arbitrary User Interface>

Wait... I was promised crazy. Where is the crazy?

There are some modules and patterns that themselves do surprising things.

- 1 REST module.
- 2 Multiplexer module.
- 3 **Layer.**

All downstream modules are now rendered into a popup layer.

Wait... I was promised crazy. Where is the crazy?

But it's really the size of the configuration space.
It's crazy how far you can get just by combining modules.

For example, if you stick a Layer under a Table, and then stick some Links under the Layer, And then a Redirector under each Link, you get...
...contextual action menus.

callingPartyNumber ⇅	originalCalledPartyNumber ⇅	finalCalledPartyNumber ⇅	cause_description ⇅	quality ⇅
4154414347	4153284435	4153284435	Normal call clearing	good acceptable
4154466278	4153284435	4153284435	Normal call clearing	acceptable good
4157043915	stop seeing only calls to/from 4153284435 exclude originalCalledPartyNumber="4153284435"		Normal call clearing	good acceptable
4157045968	view DN/extension details view call details		Normal call clearing	acceptable good
+4401494434411	4153284435	4153284435	Normal call clearing	acceptable good

Wait... I was promised crazy. Where is the crazy?

Let's introduce a weird notation for this.

Search + Table + Layer + Link + Redirector

= contextual action menus.

callingPartyNumber ↕	originalCalledPartyNumber ↕	finalCalledPartyNumber ↕	cause_description ↕	quality ↕
4154414347	4153284435	4153284435	Normal call clearing	good acceptable
4154466278	4153284435	4153284435	Normal call clearing	acceptable good
4157043915	stop seeing only calls to/from 4153284435		Normal call clearing	good acceptable
	exclude originalCalledPartyNumber="4153284435"			
4157045968	view DN/extension details		Normal call clearing	acceptable good
	view call details			
+4401494434411	4153284435	4153284435	Normal call clearing	acceptable good

What if we added more crazy?...

Just considering the more “normal” modules, the network effects of combining them into different patterns are a little overwhelming.

But when you fold in REST and Multiplexer it gets weird.

What if we added more crazy?...

Just considering the more “normal” modules, the network effects of combining them into different patterns are a little overwhelming.

But when you fold in REST and Multiplexer it gets weird.

Search + Table + Search + Pager + Multiplexer + PostProcess + Chart + Redirector

This gives you a table that when users click on it, loads a paged series of charts beneath the table that the user can page through.

FOR EXAMPLE: For each user, show me a timechart of their N most recent sessions.

AND for each of those little charts visualizing that one session, it can have a live drilldown to a detail view or a raw search or both.

What if we added more crazy?...

**Search (rest command) + Multiplexer + PostProcess + Multiplexer
+ Search + Multiplexer + Layer + Search + Table**

Yes we're not only multiplexing Multiplexers, but we're then multiplexing multiplexed Multiplexers.

We... just created an extremely odd little page that allows you to browse all EAI entities in Manager.

Let's... go drive it around.

Features you might enjoy

- **Back Button support**
even in complex pages (see URLLoader module)
- **Tabs**
think of them like odd Pulldown modules.
Often combined with a Switcher module.
- **CheckboxPulldown**
unlike some other frameworks, yes there is one.
- **SearchControls**
give your end users job controls, export, print buttons, Job Inspector.
- **Blue Links ! !**
The astonishing return of being able to render plain old blue links!
(Just Multiplex a Link module)

Next Steps, Questions

Can I use it? Is it free?

It's distributed under the "Sideview Free Internal Use License Agreement".

Basically it is free for "internal use", meaning within your company you can build views and apps.

If you want to build an app for other companies, other people, Then you need to talk to us for a different agreement.

What do I need to do, to build Canary views in our apps.

There is some hoopjumping to make your app redirect correctly back and forth from core Splunk ui pieces, to Canary pieces.

There is a tool coming in the next release that will do this busywork for you.

<https://sideviewapps.com/apps/canary/>

Where is it going?

It can post things to Splunk searches, and post things to arbitrary REST endpoints....

Can it post things to other systems outside of Splunk, AND render rows of key value pairs FROM those systems?

Yes! Ish!

Can it run totally independently from Splunk?

It will someday.