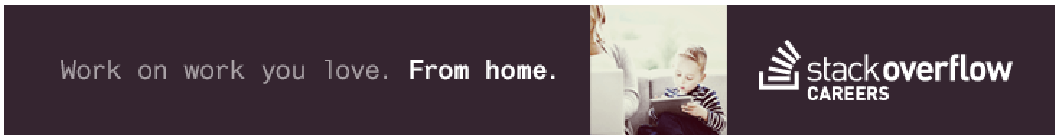
Reagent React Clojurescript Warning: Every element in a seq should have a unique :key



I have copied a two year old gist from here. It is now working with Figwheel and uses a more recent version of Reagent/React. I am looking for a generic way of isolating this warning message that comes to the Javascript console: `Warning: Every element in a seq should have a unique :key`. The idea is to put a `:key` with a generated unique value into all the components. Then the messages ought to disappear and I'll be in a position to see which components needed the unique `:key`. My problem is that even although a unique `:key` is being put into all of them, the warning message is still seen.

So - would someone be able to tell me which component I have missed or otherwise what I've done wrong? As you can see from the source (permalink) I have added `:key (gen-key)` to the two components: `[:polyline {:key (gen-key) ...` and `[:svg {:key (gen-key) ...` at lines 43 and 68 respectively.

Edit So this is the answer (permalink), in terms of code anyway. Just look for the placement of `^{:key (gen-key)}` at lines 44 and 60.

Note that the function `gen-key` was made for debugging. Natural keys to replace.

clojurescript reagent

edited Nov 3 at 0:07

asked Oct 31 at 0:53
Chris Murphy
465 4 11

2 Answers

From the example at the Reagent Project site

```
(defn lister [items]
  [:ul
   (for [item items]
     ^{:key item} [:li "Item " item])])

(defn lister-user []
  [:div
   "Here is a list:"
   [lister (range 3)]])
```

> Note: The ^{:key item} part above isn’t really necessary in this simple example, but attaching a unique key to every item in a dynamically generated list of components is good practice, and helps React to improve performance for large lists. The key can be given either (as in this example) as meta-data, or as a :key item in the first argument to a component (if it is a map). See React’s documentation for more info.

and the react documentation on dynamic children should point you in the right direction. At a high level, if you have some code which generates a number of similar components in some sort of loop, you need to prefix each component with the `^{:key val}`. However, as reagent needs things to be in vectors, you will typically need to wrap the output from the looping construct in some other vector, such as the `[:ul]` in the above example or a `[:div]` in a general case. I sometimes use a `(into [:div] (for ....))` type of construct to do this.

answered Nov 1 at 1:34
Tim X
993 4 10

The code is using `map` rather than `for` and `into`, but to the same effect. I have played around with :key either as metadata or as the first argument to the component. But as per the documentation *whichever* makes no odds. There are many x y points put in the `[:polyline` - that have no :key - perhaps that's the issue - I'm still trying to work it out... – Chris Murphy Nov 1 at 6:04

1 It definitely needs to be as metadata not the first element of the component and so must be of the form `^{:key val} [compnent]` . Note also that reagent doesn't deal with lazy-seq well, which is why you will often see things like `(into []...)` or `(doall ...)` used to ensure the seq is realised. Make sure your using latest reagent as error messages got a bit better. Seem to remember seeing a reactid referenced with the error last time I had it, so maybe you can use the browser inspector to look at page elements to get an idea which component is responsible – Tim X Nov 1 at 8:14

There is some excellent reagent documentation on the day8/re-fram wiki which might help. In particular, github.com/Day8/re-frame/wiki/Using-%5B%5D-instead-of-%28%29 might help with understanding where reagent will create react components, which isn't always obvious and may help track down where you need the metadata – Tim X Nov 1 at 8:40

That general case of `(into [:div] (for ....))` is very useful. I would add that if it is an `[:svg` component, you can contain with `[:g]` rather than `[:div]` . – Chris Murphy Nov 5 at 6:48

I believe you need to supply the :key as metadata instead, like e.g.

```
^{:key (gen-key)} [:polyline ...
```

I think that it only can be added as a map entry for components, like you do here:

```
[trending-app {:key (gen-key) :state-ref paths-ratom :comms ch}]
```

So the above works, but not e.g.

```
[:svg {:key (gen-key)
```

answered Oct 31 at 20:05
PeakCode
81 3

I tried putting `^{:key (gen-key)}` out the front for `[:polyline` and `[:svg` . No luck so far. Also with `[trending-app` there does not need to be anything so I'll fix that in the source. Can you give a documentation reference as part of your answer? – Chris Murphy Oct 31 at 22:40