ARTG 6900 Week 6

Event Architecture

Previous Weeks

The architecture of more complex visualization projects
- Reusable modules
- “Models” and “views”

This Week

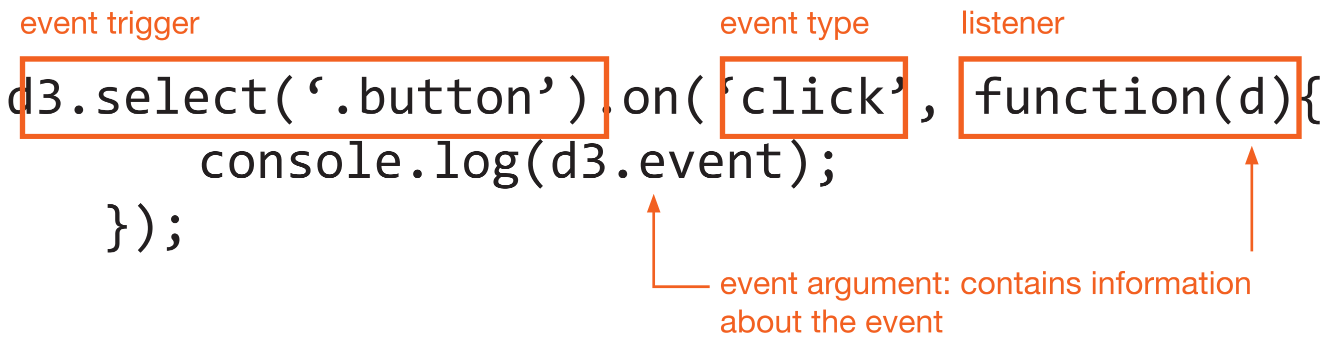
How do the various components in a complex visualization communicate?
- Event architecture with d3.dispatch

Begin to think about the overall architecture of your final project

d3.dispatch - motivation & basic API

Events and event listeners

We are used to seeing this event handling pattern.



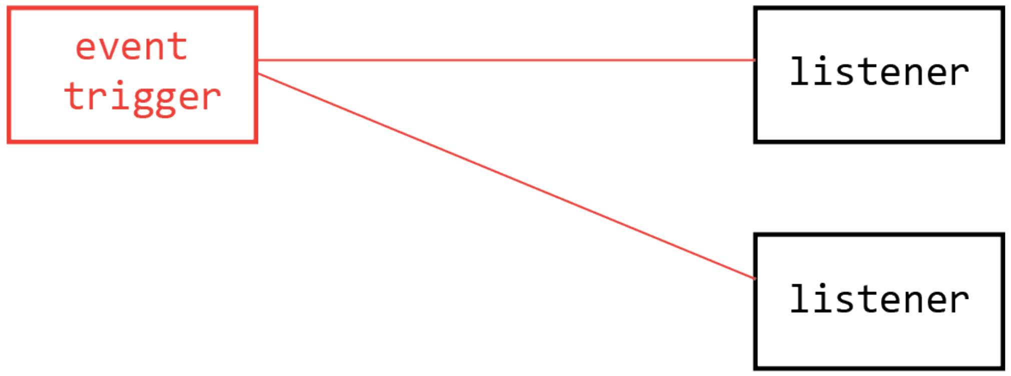
```
d3.select('.button').on('click', function(d){
        console.log(d3.event);
    });
```

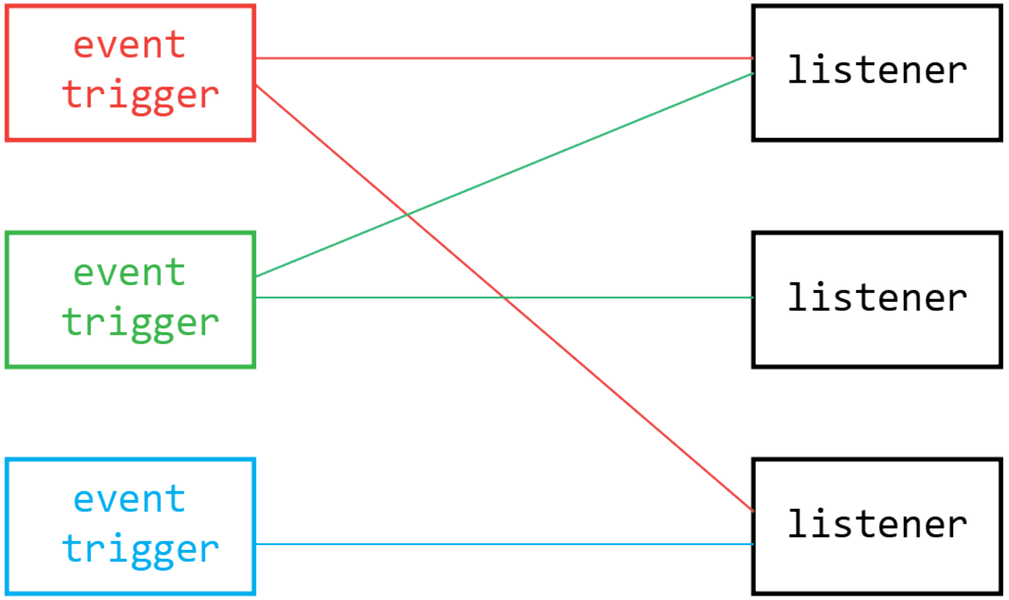
Events and event listeners

Multiple listeners can respond to the same event trigger

```
d3.select(‘.button’).on(‘click’, function(d){
      console.log(d3.event);
   })
   .on(‘click.foo’, listenerFunction2);
```

In order for multiple listeners to be registered for the same event, we must “namespace” the event individually for each listener


event trigger
listener
listener


event trigger
event trigger
event trigger
listener
listener
listener

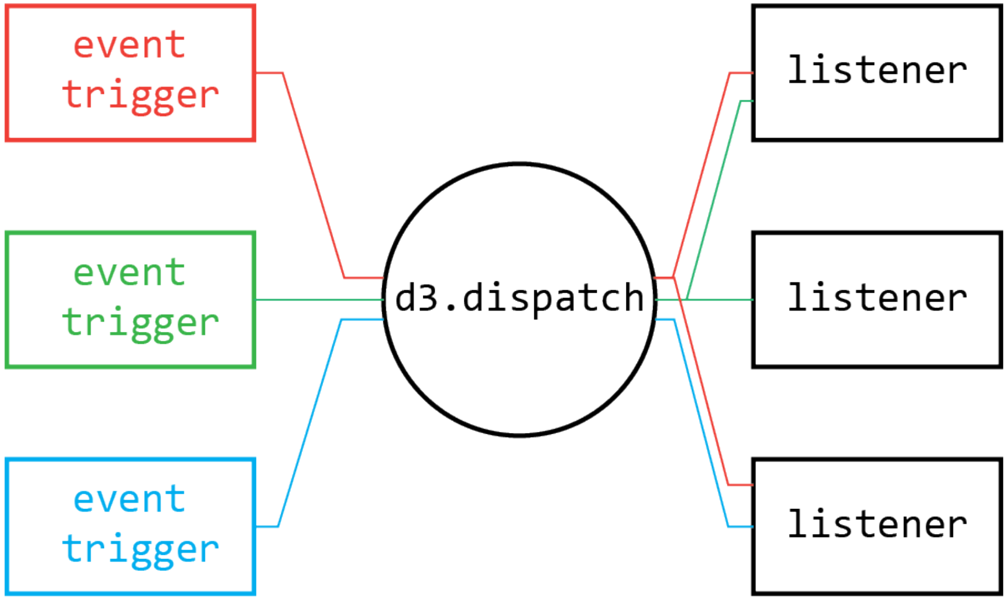
Limitations of This Approach

Event triggers and listeners are tightly coupled -- doesn’t play well with dynamic modules.

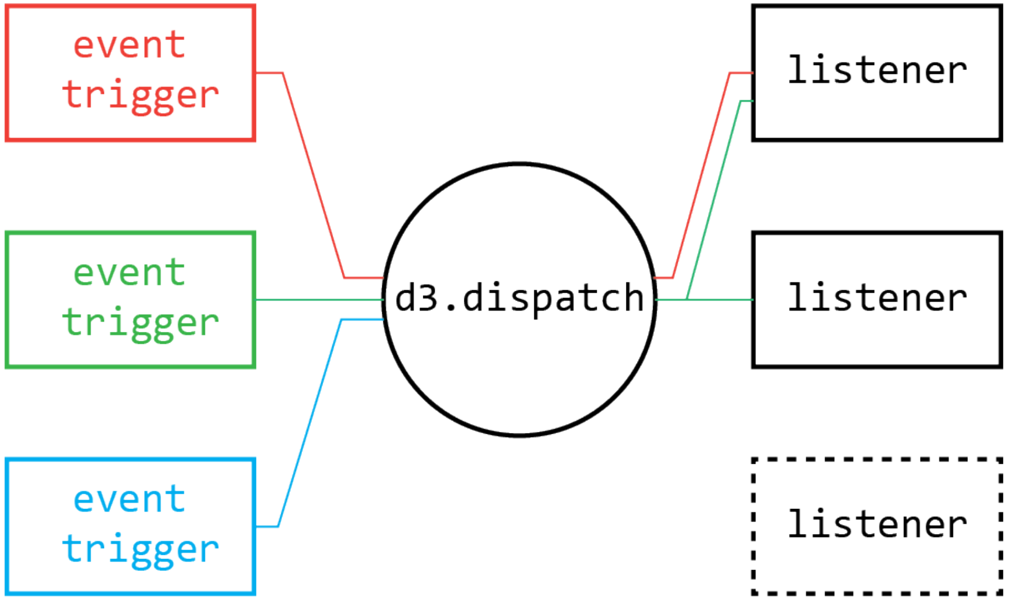
Hard to keep track of!

Only DOM events are supported.

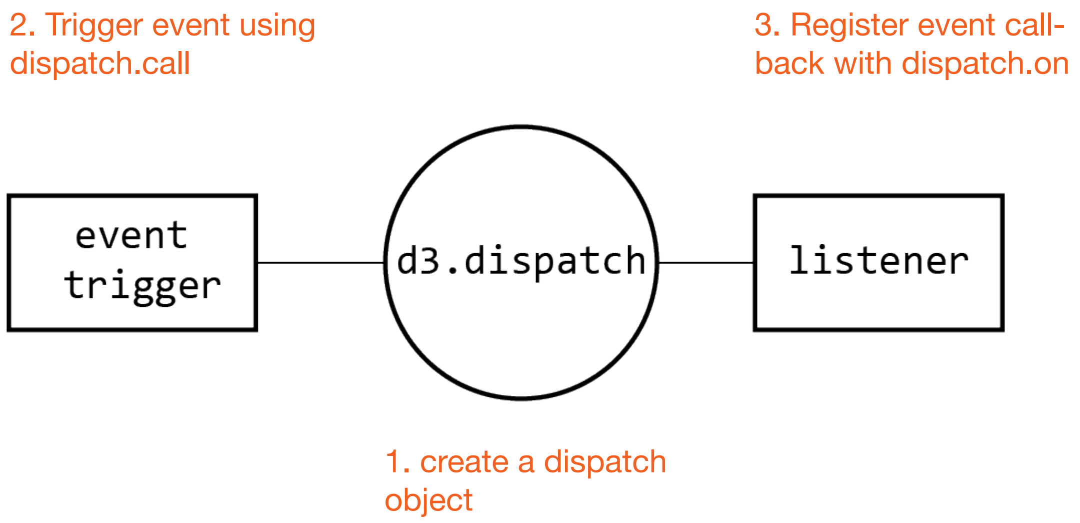
d3.dispatch: loosely coupled events



d3.dispatch: loosely coupled events



d3.dispatch: API details


2. Trigger event using dispatch.call
3. Register event call-back with dispatch.on
event trigger
d3.dispatch
listener
1. create a dispatch object

d3.dispatch: API details

Create a dispatch object

```
var dispatch = d3.dispatch(‘customEvent1’,
‘customEvent2’);
```

Register event callback

```
dispatch.on(‘customEvent1’, function(){
      //callback function
   });
```

Trigger event

```
dispatch.call(‘customEvent1’);
```

d3.dispatch: additional event arguments

Create a dispatch object

```
var dispatch = d3.dispatch('customEvent1',
'customEvent2');
```

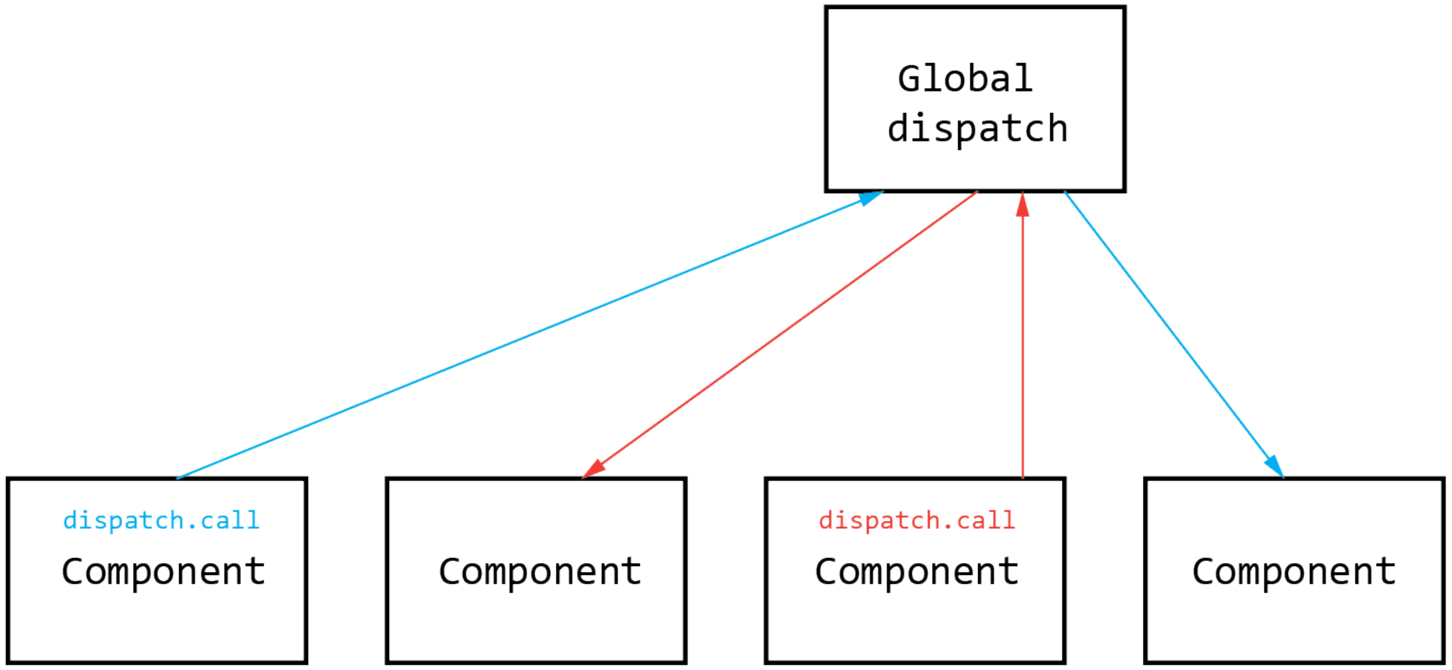
Register event callback

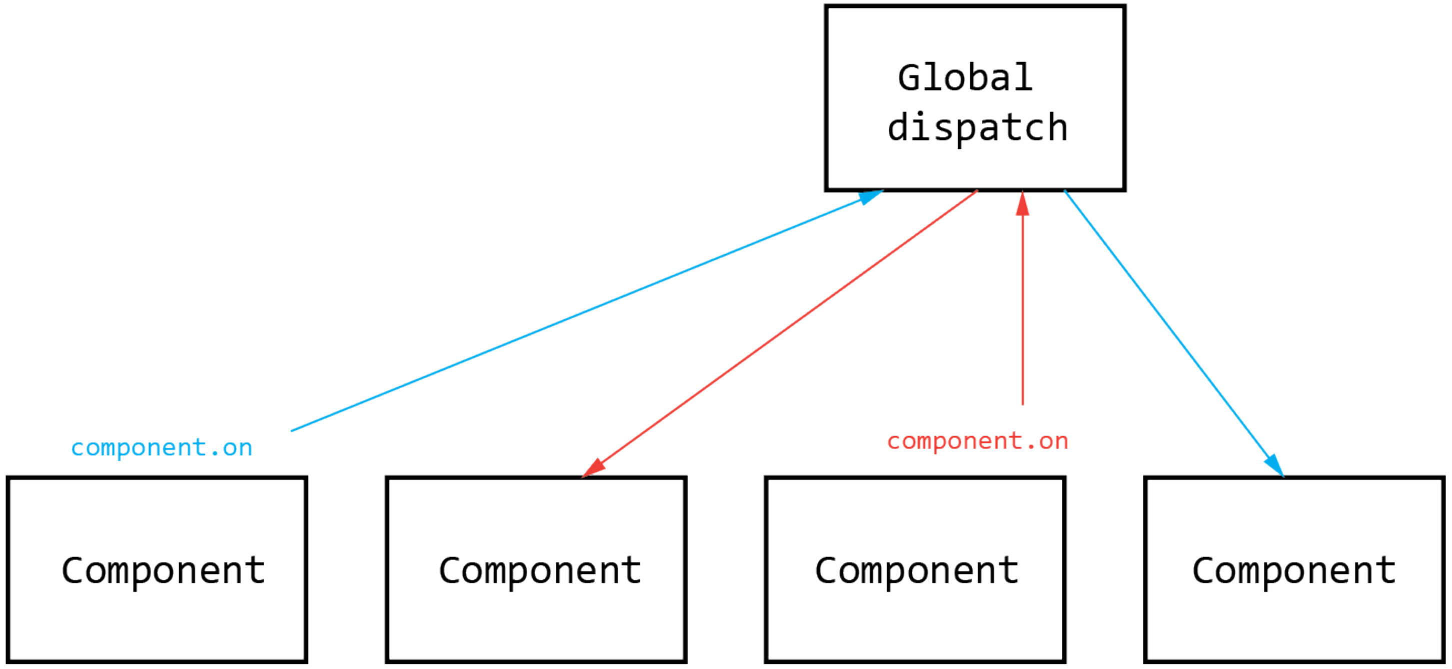
```
dispatch.on('customEvent1', function(arg1, arg2){
        //callback function
    });
```

Trigger event

```
dispatch.call('customEvent1', context, arg1, arg2);
```

d3.dispatch - putting this in practice


Global dispatch
dispatch.call
Component
Component
dispatch.call
Component
Component


Global dispatch
component.on
component.on
Component
Component
Component
Component

app.js

```
var timeseries = Timeseries();

timeseries.on('someEvent', listener);
```

Timeseries.js

```
function Timeseries(){
  var _dis = d3.dispatch('someEvent');

  var exports = function(div){
     ...

     _dis.call('someEvent', this, args);
  }

  exports.on = function(){
     _dis.on.apply(_dis,arguments);
  }

  return exports;
}
```

With “rebinding”, internal events can now be broadcast outside the module.

Modules, Event Architecture, and Next Steps

Drawing with <canvas>: Quick Warm-up