# Flux data fetching – REACTJS - transcript

Inside topic-list.jsx, our topic state starts as an empty array. This allows us to write a component that maps over ‘this.state.topics’ and not run into any errors if topics wasn’t defined.

For example, imagine if we started out with topic being equal to null, whenever we try to say ‘this.state.topics.map’, that would throw an error because we cannot map over null.

When the component is about to be rendered, it calls an action called getTopics.

If we open our Actions object, we see that we have a single action called getTopics. getTopics is really simple. It is going to call in any store that is listening this set of actions. If we call getTopics, we are going to run getTopics in that store as well.

Next, looking at topic-store.jsx, we are listening to all of the available actions via the property, listenables. ‘listenables’ is implemented automatically by Reflux for us. ‘listenables’ means any action, that is triggered by this Actions object, for any action that is triggered, this store will attempt to run a method with exact same name as that action that was triggered. In our case, triggering the action, getTopics will run the method, getTopics.

Next, running getTopics will fire off an AJAX request by using the api module that we wrote. When the AJAX request is complete, we will get some amount of json data. This is the data that we actually care about, so we save it ‘save.topics’. We then trigger the event called ‘change’, by calling triggerChange.

When we trigger the event ‘change’, we also pass in our list of topics. so we update our list of data.

Back inside topic-list.jsx, we have used the listenTo mixin to listen to any events that is coming off the TopicStore. Whenever an event is triggered by the store, we will automatically run the method, ‘onChange’. In the method, ‘ onChange’, we take the new list of topics and set it on our state as ‘this.state.topics’.

Finally, when we set the state of component, it automatically triggers a re-render of that component.