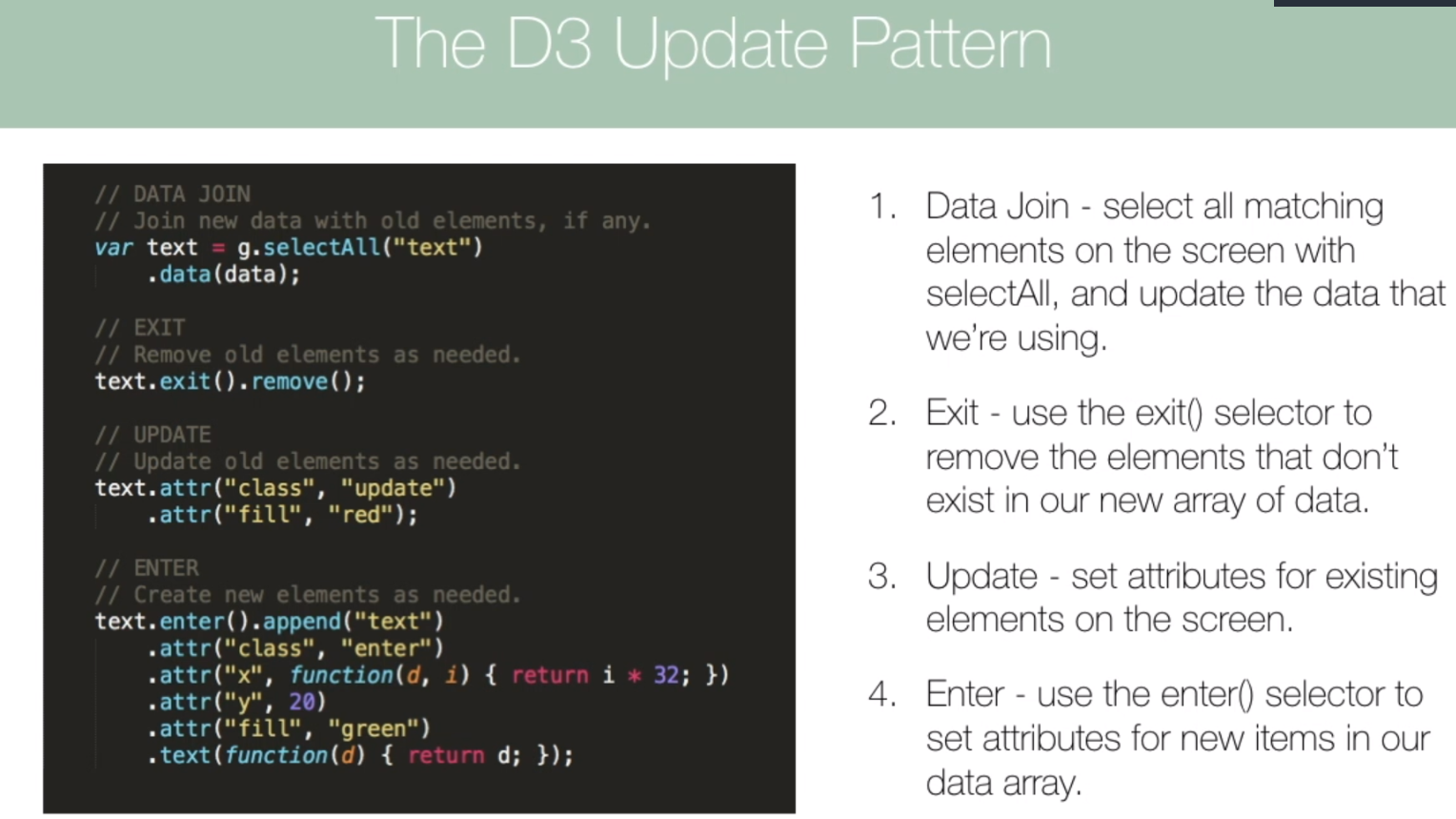
D3 general update pattern then and now

**Then** enter update exit



**Now** join

*[#](https://github.com/d3/d3-selection/blob/master/README.md" \l "selection_join) selection.****join****(enter[, update][, exit])*[*<>*](https://github.com/d3/d3-selection/blob/master/src/selection/join.js)

*Appends, removes and reorders elements as necessary to match the data that was previously bound by*[selection](https://github.com/d3/d3-selection/blob/master/README.md" \l "selection_data)*[.data](https://github.com/d3/d3-selection/blob/master/README.md" \l "selection_data), returning the*[*merged*](https://github.com/d3/d3-selection/blob/master/README.md#selection_merge)*enter and update selection. This method is a convenient alternative to the explicit*[*general update pattern*](https://bl.ocks.org/mbostock/3808218)*, replacing*[selection*.enter*](https://github.com/d3/d3-selection/blob/master/README.md#selection_enter)*,*[selection*.exit*](https://github.com/d3/d3-selection/blob/master/README.md#selection_exit)*,*[selection*.append*](https://github.com/d3/d3-selection/blob/master/README.md#selection_append)*,*[selection*.remove*](https://github.com/d3/d3-selection/blob/master/README.md#selection_remove)*, and*[selection](https://github.com/d3/d3-selection/blob/master/README.md" \l "selection_order)*[.order](https://github.com/d3/d3-selection/blob/master/README.md" \l "selection_order).*

[*https://observablehq.com/@d3/selection-join*](https://observablehq.com/@d3/selection-join)

***selection.join***

*D3’s*[*data join*](https://bost.ocks.org/mike/join/)*lets you specify exactly what happens to the DOM as data changes. This makes it fast—you can minimize DOM operations—and expressive—you can animate enter, update and exit separately. Yet power comes at a cost: the data join’s generality makes it hard to learn and easy to forget.*

*What if D3 offered a more convenient and memorable API for joining data without sacrificing this power? Enter d3-selection 1.4’s [selection.join](https://github.com/d3/d3-selection/blob/master/README.md" \l "selection_join).*

*If the joining selection isn’t empty—as on subsequent iterations of the loop above— selection.join*[*appends*](https://github.com/d3/d3-selection/blob/master/README.md#selection_append)[*entering*](https://github.com/d3/d3-selection/blob/master/README.md#selection_enter)*elements and*[*removes*](https://github.com/d3/d3-selection/blob/master/README.md#selection_remove)[*exiting*](https://github.com/d3/d3-selection/blob/master/README.md#selection_exit)*elements to match the data! Entering and updating elements are*[*merged*](https://github.com/d3/d3-selection/blob/master/README.md#selection_merge)*(and*[*ordered*](https://github.com/d3/d3-selection/blob/master/README.md#selection_order)*), allowing chained operations on the result.*

***This is more concise than the previous***[***general update pattern***](https://observablehq.com/@d3/general-update-pattern)***, but the best part is that you can still control what happens on enter, update and exit as desired!***

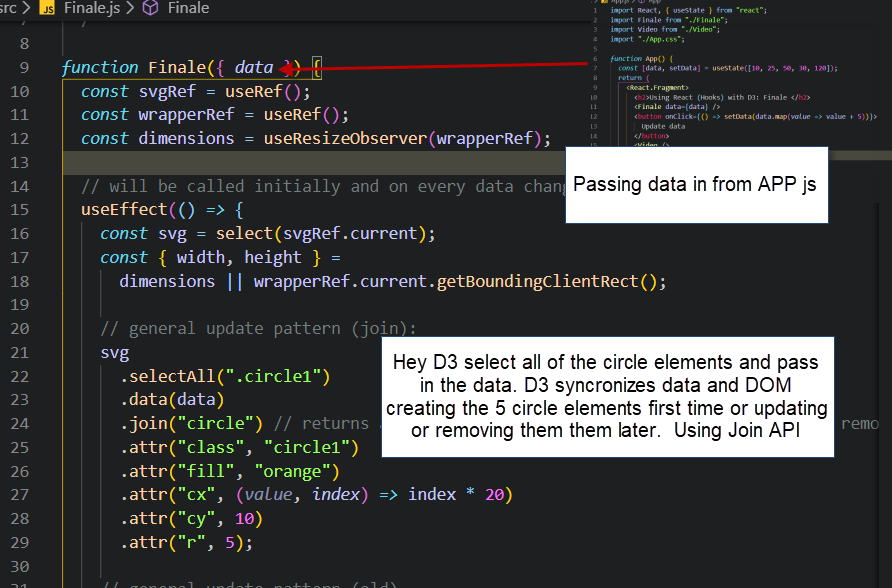
Put the stuff in the update function that changes as data changes like

The domain of scale (scaleBand() scaleLinear()

Append the axis to a group (g) once ,outside of the update function , at the top of the file , then call the axi generator in the update function ( axisBottom() axisLeft() )

**General Update Pattern (Join API )**

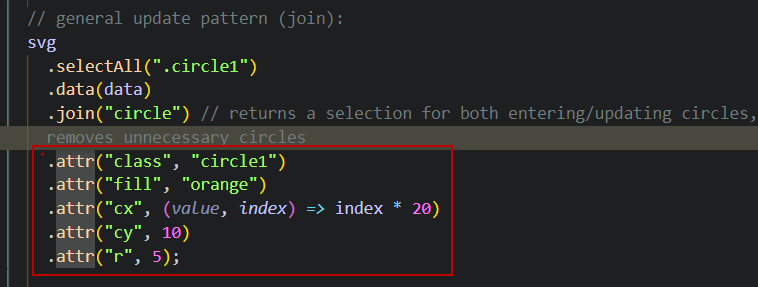
D3’s way of managing the DOM based on data. It handles stuff and react takes a step back. Provides the SVG to D3. Normally want to define data in App and pass into component as a prop.



Enter, Update and Exit are handled you the newer join API.

In Join can pass in callbacks if needed. ( for enter , update or exit) (did this in first tutorial)

Join returns new and existing elements in the SVG. Everyting after the “Join” gets applied to the new and existing elements in SVG



Right after the join the first attr adding class of circle1 is important. Otherwise could not update the circle elements we are looking for.

In the old method enter, append and merge needed to be used, along with exit and remove to remove

