**Data-Binding In Other Frameworks**

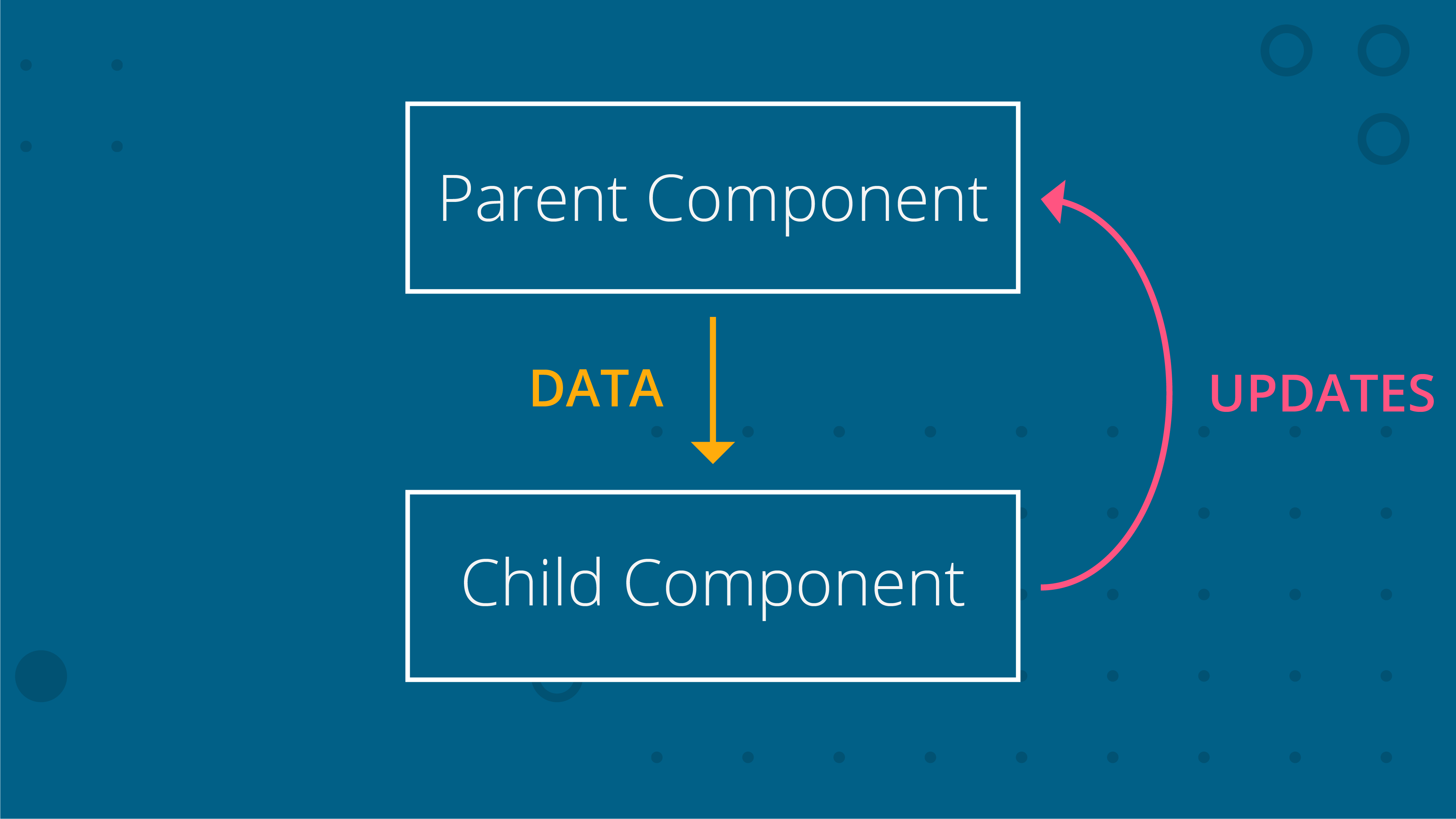
Front-end frameworks like [Angular](https://angular.io/) and [Ember](https://emberjs.com/) make use of two-way data bindings. In two-way data binding, the data is kept in sync throughout the app no matter where it is updated. If a model changes the data, then the data updates in the view. Alternatively, if the user changes the data in the view, then the data is updated in the model. Two-way data binding sounds really powerful, but it can make the application harder to reason about and know where the data is actually being updated.

**Further Research**

* [Angular's two-way data binding](https://angular.io/guide/template-syntax#two-way)
* [Ember's two-way data binding](https://guides.emberjs.com/v2.13.0/object-model/bindings/)

**React's Data-flow**

Data moves differently with React's unidirectional data flow. In React, the data flows from the parent component to a child component.



*[Data flows down from parent component to child component. Data updates are sent to the parent component where the parent performs the actual change.](https://classroom.udacity.com/nanodegrees/nd019/parts/57812373-5147-4152-a4f7-d2e164aba8a5/modules/7778e6bd-7868-47d0-8536-cfcbe4e29f19/lessons/69df9c23-5c48-4ed6-b909-49a4e62d688c/concepts/2ce529b7-7bcf-4516-8add-080c32a42eb1)*

In the image above, we have two components:

* a parent component
* a child component

The data lives in the parent component and is passed down to the child component. Even though the data lives in the parent component, both the parent and the child components can use the data. However, if the data must be updated, then only the parent component should perform the update. If the child component needs to make a change to the data, then it would send the updated data to the parent component where the change will actually be made. Once the change \_is\_ made in the parent component, the child component will be passed the data (that has just been updated!).

Now, this might seem like extra work, but having the data flow in one direction and having one place where the data is modified makes it much easier to understand how the application works.

**QUESTION 1 OF 2**

A FlightPlanner component stores the information for booking a flight. It also contains DatePicker and DestinationPicker as child components. Here's what the code might look like:

<FlightPlanner>

<DatePicker />

<DestinationPicker />

</FlightPlanner>

If this were a React application, which component should be in charge of making updates to the data? Check all that apply.

* FlightPlanner
* 

DatePicker

* 

DestinationPicker

SUBMIT

Now let's say that the FlightPlanner component has two child components: LocationPicker and DatePicker. LocationPicker itself is a parent component that has two child components: OriginPicker and DestinationPicker.

**QUESTION 2 OF 2**

If the following sample code were a React application, which of the following components should be in charge of making updates to data? Check all that apply.

<FlightPlanner>

<LocationPicker>

<OriginPicker />

<DestinationPicker />

</LocationPicker>

<DatePicker />

</FlightPlanner>

* FlightPlanner
* 

DatePicker

* LocationPicker
* 

OriginPicker

* 

DestinationPicker

SUBMIT

**Data Flow in React Recap**

In React, data flows in only one direction, from parent to child. If data is shared between sibling child components, then the data should be stored in the parent component and passed to both of the child components that need it.