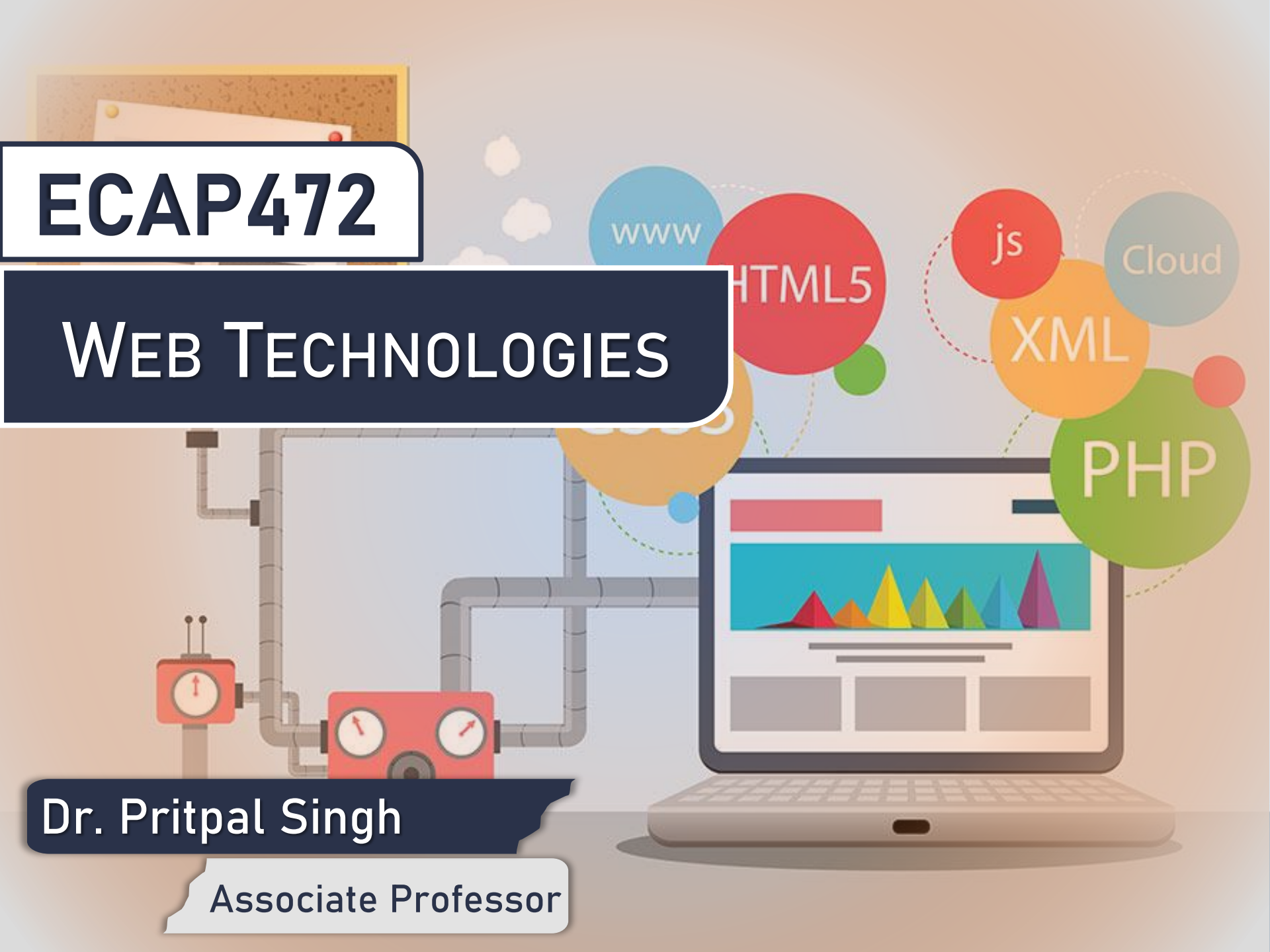
ECAP472

Web Technologies



Dr. Pritpal Singh

Associate Professor

Learning Outcomes

After this lecture, you will be able to

- Understand concept of React state management.

React Props

Props are arguments passed into React components.

Props are passed to components via HTML attributes.

Props stands for properties.

React Props

- Props stand for "Properties."
- They are read-only components. It is an object which stores the value of attributes of a tag and work similar to the HTML attributes.
- It gives a way to pass data from one component to other components.
- It is similar to function arguments. Props are passed to the component in the same way as arguments passed in a function.

React Props

- Props are immutable so we cannot modify the props from inside the component. Inside the components, we can add attributes called props. These attributes are available in the component as this.props and can be used to render dynamic data in our render method.
- When you need immutable data in the component, you have to add props to reactDom.render() method in the main.js file of your ReactJS project and used it inside the component in which you need

React Props

Props can be used to pass any kind of data such as:

String

Array

Integer

Boolean

Objects or,

Functions

Popular React JS Websites List

Facebook

React was created by a software engineer at Facebook. Since then, Facebook has maintained the framework even though it remains open-source. As such, it would be unusual if Facebook does not use it while others do. There is a lot of interactivity on the Facebook website.

Popular React JS Websites List

British Broadcasting Corporation

This is another one of the top websites built with React. The BBC has migrated its website since 2015 and it still uses it till now, although it moved to a new React-based application.Single Page Application built by the BBC World Service as a rendering platform.

State

State represents the value of a dynamic properties of a React component at a given instance. React provides a dynamic data store for each component. The internal data represents the state of a React component and can be accessed using this.state member variable of the component. Whenever the state of the component is changed, the component will re-render itself by calling the render() method along with the new state.

Example

A simple example to better understand the state management is to analyse a real-time clock component. The clock component primary job is to show the date and time of a location at the given instance. As the current time will change every second, the clock component should maintain the current date and time in it's state. As the state of the clock component changes every second, the clock's render() method will be called every second and the render() method show the current time using it's current state.

What is React State Management?

- React components have a built-in state object. The state is encapsulated data where you store assets that are persistent between component renderings.
- The state is just a fancy term for a JavaScript data structure. If a user changes state by interacting with your application, the UI may look completely different afterwards, because it's represented by this new state rather than the old state.

Why do you need React state management?

React applications are built using components and they manage their state internally and it works well for applications with few components, but when the application grows bigger, the complexity of managing states shared across components becomes difficult.

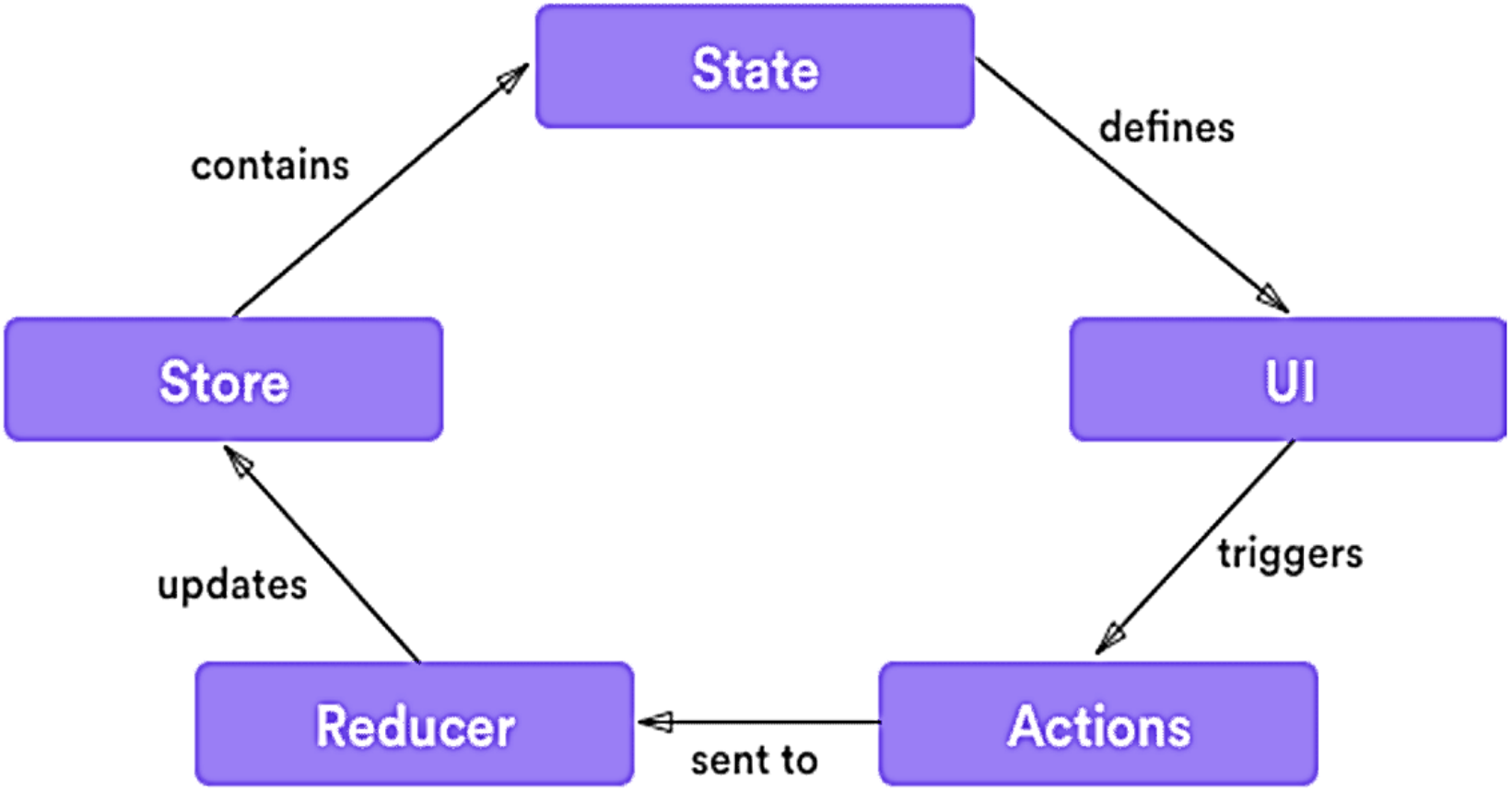
Example

- Here is a simple example of an e-commerce application, in which the status of multiple components will change when purchasing a product.
  - Add that product to the shopping list
  - Add product to customer history
  - trigger count of purchased products
  - If developers do not have scalability in mind, then it is really hard to find out what is happening when something goes wrong. This is why you need state management in your application.

Redux

Redux was created to resolve this particular issue. it provides a central store that holds all states of your application. Each component can access the stored state without sending it from one component to another.

simple view of how Redux works



Three building parts

There are three building parts:

actions,	store,	reducers

- Actions are payloads of information that send data from your application to your store.
- Actions are sent using store.dispatch().
- Actions are created via an action creator.

Three building parts

There are three building parts:

actions,	store,	reducers

- Actions are payloads of information that send data from your application to your store.
- Actions are sent using store.dispatch().
- Actions are created via an action creator.

Three building parts

There are three building parts:

actions,	store,	reducers

- Actions are payloads of information that send data from your application to your store.
- Actions are sent using store.dispatch().
- Actions are created via an action creator.

Three building parts

There are three building parts:

actions,	store,	reducers

- Actions are payloads of information that send data from your application to your store.
- Actions are sent using store.dispatch().
- Actions are created via an action creator.

Reducers in Redux

- Reducers specify how the application's state changes in response to actions sent to the store.
- The store holds the application state. You can access stored state, update the state, and register or unregister listeners via helper methods.
- Redux gives you code organization and debugging superpowers. This makes it easier to build more maintainable code, and much easier to track down the root cause when something goes wrong.

Reducers in Redux

- Reducers specify how the application's state changes in response to actions sent to the store.
- The store holds the application state. You can access stored state, update the state, and register or unregister listeners via helper methods.
- Redux gives you code organization and debugging superpowers. This makes it easier to build more maintainable code, and much easier to track down the root cause when something goes wrong.

Reducers in Redux

- Reducers specify how the application's state changes in response to actions sent to the store.
- The store holds the application state. You can access stored state, update the state, and register or unregister listeners via helper methods.
- Redux gives you code organization and debugging superpowers. This makes it easier to build more maintainable code, and much easier to track down the root cause when something goes wrong.

React Hook

These are functions that hook you into React state and features from function components. Hooks don't work inside classes and it allows you to use React features without writing a class.

Hooks are backwards-compatible, which means it doesn't keep any breaking changes. React provides some built-in Hooks like useState, UseEffect and useReducer etc. You can also make custom hooks.

React Hook Rules

Call hook at the top level only means that you need to call inside a loop, nested function, or conditions.

React function components are called hooks only.

Practical

That’s all for now…