* **Type Checking with PropTypes**:
  + PropTypes allow defining the **type of value** expected for each prop in a component.
  + Example: Ensuring maxRating must be a **number** and nothing else.
* **TypeScript vs PropTypes**:
  + For strict type checking, **TypeScript** is preferred over PropTypes.
  + PropTypes are still useful for **reusable components** but are less commonly used by developers today.
* **Using PropTypes**:
  + Import the PropTypes object from the prop-types package.
  + PropTypes are part of **Create-React-App** and don't require installation.
  + Define PropTypes using the **lowercase propTypes** property of the component.
* **Defining PropTypes for Props**:
  + Use the imported PropTypes object to specify the type of each prop:
    - maxRating: PropTypes.number
    - defaultRating: PropTypes.number
    - color: PropTypes.string
    - messages: PropTypes.array
    - className: PropTypes.string
    - onSetRating: PropTypes.func
  + Additional validators:
    - PropTypes.bool for boolean values.
    - PropTypes.object for objects.
* **Warnings with Invalid Props**:
  + Passing incorrect types (e.g., a string instead of a number) generates **warnings**.
  + Example warning: "Invalid prop of maxRating supplied, expected a number."
  + This helps developers **catch bugs** caused by accidental type mismatches.
* **Marking Props as Required**:
  + Use .isRequired to ensure a prop is mandatory.
  + Not necessary if **default values** are already defined for props.
* **Best Practices**:
  + Use **default values** instead of marking props as required.
  + PropTypes serve as both **type validation** and **documentation** for components.
* **Limitations**:
  + PropTypes are not as robust as TypeScript for large projects.
  + Recommended for **highly reusable components** if TypeScript is not used.

Code:

// Import React and PropTypes

import React from "react";

import PropTypes from "prop-types";

// Define the StarRating component

const StarRating = ({ maxRating, defaultRating, color, messages, className, onSetRating }) => {

// Component logic here (not the focus of this example)

return (

<div className={className} style={{ color }}>

<p>Max Rating: {maxRating}</p>

<p>Default Rating: {defaultRating}</p>

<p>Messages: {messages.join(", ")}</p>

<button onClick={onSetRating}>Set Rating</button>

</div>

);

};

// Define propTypes for the StarRating component

StarRating.propTypes = {

// maxRating must be a number

maxRating: PropTypes.number,

// defaultRating must be a number

defaultRating: PropTypes.number,

// color must be a string

color: PropTypes.string,

// messages must be an array

messages: PropTypes.array,

// className must be a string

className: PropTypes.string,

// onSetRating must be a function

onSetRating: PropTypes.func,

};

// Define defaultProps for the StarRating component

StarRating.defaultProps = {

maxRating: 5, // Default max rating is 5

defaultRating: 0, // Default rating is 0

color: "black", // Default color is black

messages: [], // Default is an empty array

className: "", // Default is an empty string

onSetRating: () => {}, // Default is an empty function

};

// Export the StarRating component

export default StarRating;