**استایل دهی در ری اکت**

**1 – استایل دهی درون خطی ( inline Styling )**

* به صورت ابجکت استایل ها رو می نویسیم و به المنت ها می دهیم : مثال :

const myStyle = {border:'1px solid red'}

<h1 style={myStyle}>شمارنده من</h1>

**2 – استایل دهی به صورت گلوبال در پوشه public**

* ایجاد پوشه css مثلا – ایجاد فایل app.css – لینک دهی در index.html

**3 – استفاده از ccs module**

چرا باید استفاده کنیم ؟به ما این قابلیت را می دهیم تا scope استایل های خود را تغییر دهیم . زمانی که فقط می خواهیم برای فقط کامپوننت خاص استایل بنویسیم .

دو روش داریم :

1 – باید پروژه رو eject کنیم . .... این روش دیگه توصیه نمی شود چرا که در CRA این امکان اضافه شده است .

2 – کافی هست یک فایل css بسازیم و نام کامپوننت را بنویسم . **app.module.css** همین رو بسازیم خود CRA می فهمه که باید این استایل رو به صورت scope اعمال کند 😊

**استایل دهی داینامیک در ری اکت**

<p style={count === 0 ? {color:'red'} : {color:'green'}}>{count}</p>

<p style={{color: count <= 0 ? 'red' : 'green'}}>{count}</p>

npm i classname

**آشنایی با ابزاری به ClassName 😊**

npm i classnames

زمانی که می خواهیم در صورت محیا بودن یک شرط کلاس های متعدد را رندر کنیم . یعنی بسته به true یا false بودن شرایط مختلف کلاس هایی را اضافه و یا کم کنیم .

var classNames = require('classnames');

class Button extends React.Component {

  // ...

  render () {

    var btnClass = classNames({

      btn: true,

      'btn-pressed': this.state.isPressed,

      'btn-over': !this.state.isPressed && this.state.isHovered

    });

    return <button className={btnClass}>{this.props.label}</button>;

  }

}

اگر ابزاری به نام classnames نبود کد هامون این شکلی می شد . .. . . . ☹

class Button extends React.Component {

  // ...

  render () {

    var btnClass = 'btn';

    if (this.state.isPressed) btnClass += ' btn-pressed';

    else if (this.state.isHovered) btnClass += ' btn-over';

    return <button className={btnClass}>{this.props.label}</button>;

  }

}