

Diagram illustrating the initial rendering process. A blue arrow labeled "root" points to the `render` function. The function is defined as follows:

```
function render(element, container) {
  let domElement = document.createElement(element.tag); //div
  if(['string', 'number'].includes(typeof element)) {
    container.appendChild(document.createTextNode(String(element)));
    return;
  }
  if(element.props.children) {
    element.props.children.forEach(child => {
      render(child, domElement);
    });
  }
  container.appendChild(domElement); //div
}

let root = document.getElementById('root');
render(ProductCard, root);
```

Handwritten notes include "JSX" and a blue arrow pointing to the `render` function. A blue arrow points from the `render` function to the `ProductCard` component, which is represented by the following JSX:

```
<div id="root"></div>
```

On the right, a JSON representation of the component tree is shown:

```
{
  "tag": "div",
  "props": {},
  "children": [
    {
      "tag": "h2",
      "props": {
        "children": ["iPhone"]
      },
      "type": "h2"
    },
    {
      "tag": "p",
      "props": {
        "children": ["Price: Rs. ", "90000"]
      },
      "type": "p"
    }
  ]
}
```

Diagram illustrating the rendering process with a red 'X' indicating an error. The `render` function is shown with a red 'X' next to the `render` function call. The JSX is `<div id="root"></div>`. The JSON representation of the component tree is shown with a red 'X' next to the `render` function call.

```
function render(element, container) {
  let domElement = document.createElement(element.tag); //div
  if(['string', 'number'].includes(typeof element)) {
    container.appendChild(document.createTextNode(String(element)));
    return;
  }
  if(element.props.children) {
    element.props.children.forEach(child => {
      render(child, domElement);
    });
  }
  container.appendChild(domElement); //div
}

let root = document.getElementById('root');
render(ProductCard, root);
```

Handwritten notes include `<div> </div>` and a red 'X' next to the `render` function call. A red arrow points from the `render` function to the `ProductCard` component, which is represented by the following JSX:

```
<div id="root"></div>
```

On the right, a JSON representation of the component tree is shown:

```
{
  "tag": "div",
  "props": {},
  "children": [
    {
      "tag": "h2",
      "props": {
        "children": ["iPhone"]
      },
      "type": "h2"
    },
    {
      "tag": "p",
      "props": {
        "children": ["Price: Rs. ", "90000"]
      },
      "type": "p"
    }
  ]
}
```

Diagram illustrating the rendering process with a red 'X' indicating an error. The `render` function is shown with a red 'X' next to the `render` function call. The JSX is `<div id="root"></div>`. The JSON representation of the component tree is shown with a red 'X' next to the `render` function call.

```
function render(element, container) {
  let domElement = document.createElement(element.tag); //div
  if(['string', 'number'].includes(typeof element)) {
    container.appendChild(document.createTextNode(String(element)));
    return;
  }
  if(element.props.children) {
    element.props.children.forEach(child => {
      render(child, domElement);
    });
  }
  container.appendChild(domElement); //div
}

let root = document.getElementById('root');
render(ProductCard, root);
```

Handwritten notes include `<div> </div>` and a red 'X' next to the `render` function call. A red arrow points from the `render` function to the `ProductCard` component, which is represented by the following JSX:

```
<div id="root"></div>
```

On the right, a JSON representation of the component tree is shown:

```
{
  "tag": "div",
  "props": {},
  "children": [
    {
      "tag": "h2",
      "props": {
        "children": ["iPhone"]
      },
      "type": "h2"
    },
    {
      "tag": "p",
      "props": {
        "children": ["Price: Rs. ", "90000"]
      },
      "type": "p"
    }
  ]
}
```

h2

```
function render(element, container) {
  let domElement = document.createElement(element.tag); //div
  if(['string', 'number'].includes(typeof element)) {
    container.appendChild(document.createTextNode(String(element)));
    return
  }

  if(element.props.children) {
    element.props.children.forEach(child => {
      render(child, domElement);
    });
    container.appendChild(domElement); //div
  }
}

let root = document.getElementById('root');
render(ProductCard, root);
```

▼ {tag: 'div', props: {}} ⓘ

▼ props:

▼ children: Array(2)

▼ 0:

▼ props:

► children: ['iPhone']

► [[Prototype]]: Object

tag: "h2"

► [[Prototype]]: Object

▼ 1:

▼ props:

► children: (2) ['Price: Rs. ', '90000']

► [[Prototype]]: Object

tag: "p"

► [[Prototype]]: Object

<div id="root"></div>

(1)