I’m LA


ESCROW.COM™


freelancer

<InANutshell topic="JSX">

JavaScript XML*

* Not the actual acronym, I just made it up.

```
const element = (
    <div>
        <h1>Hello, world!</h1>
        <h2>It is {new Date().toLocaleTimeString()}.</h2>
    </div>
);
```

TL;DR

Extension to ECMAScript

JSX → Virtual DOM → DOM

Syntactic Sugar

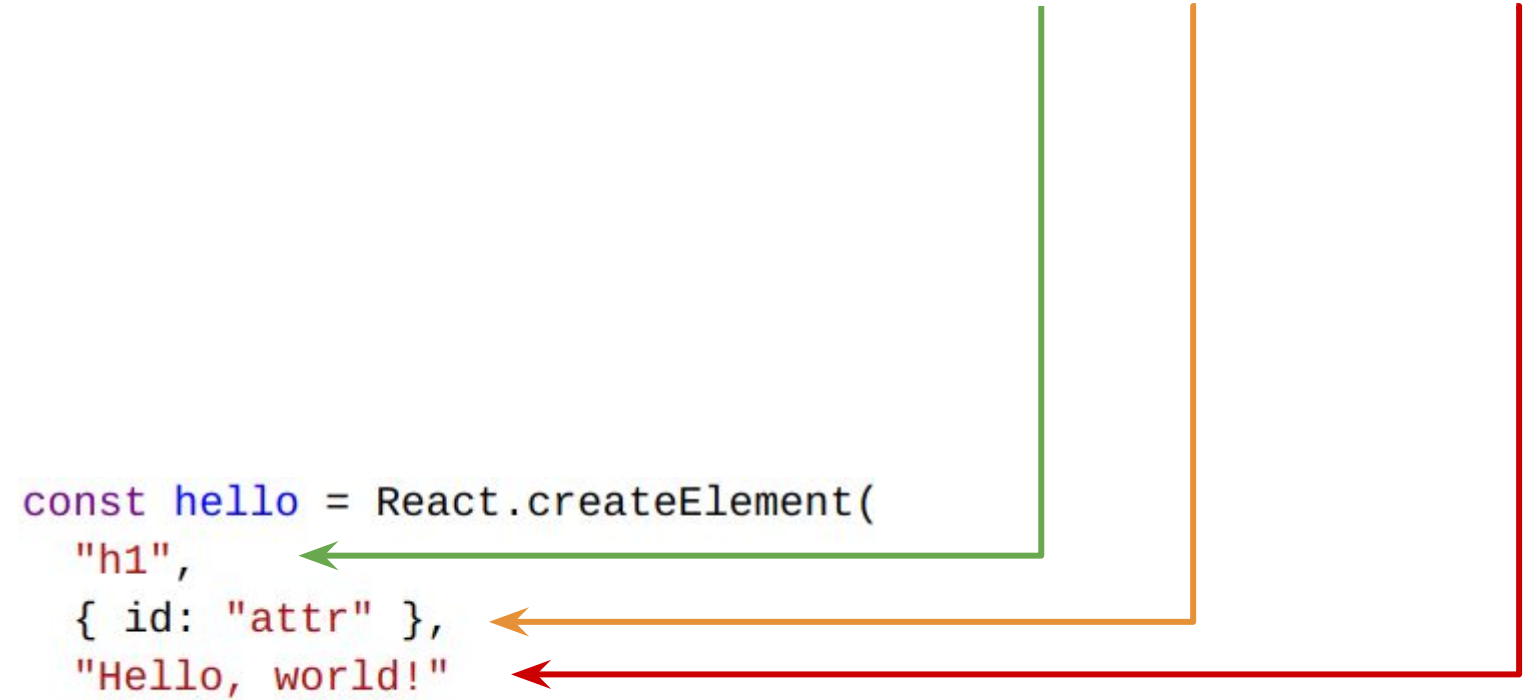
```jsx
const element = (
  <h1 className="greeting">
    Hello, world!
  </h1>
);
```

```jsx
const element = someFunction(
  "h1",
  { className: "greeting" },
  "Hello, World!"
);
```

Virtual DOM


JSX → Virtual DOM™ → DOM

```jsx
const hello = <h1 id="attr">Hello, world!</h1>;
```



```jsx
const hello = React.createElement(
  "h1",
  { id: "attr" },
  "Hello, world!"
);
```

pragma

`string`, defaults to `React.createElement`.

Replace the function used when compiling JSX expressions.

Note that the `@jsx React.DOM` pragma has been deprecated as of React v0.12

```jsx
/** @jsx fln */

const hello = (
    <h1 id="attr">
        Hello, world!
        <p>Hi!</p>
        What's up?
    </h1>
);
```

```jsx
/** @jsx fln */

const hello = fln(
   "h1",
   { id: "attr" },
   "Hello, world!",
   fln(
      "p",
      null,
      "Hi!"
   ),
   "What's up?"
);
```

```
/** @jsx fln */

const hello = (
    <h1 id="attr">
        Hello, world!
        <p>Hi!</p>
        What's up?
    </h1>
);

function fln(node, attributes, ...rest) {
  const children = rest.length ? rest : null
  return {
    node,
    attributes,
    children
  };
}
```

```jsx
/** @jsx fln */

const hello = (
    <h1 id="attr">
        Hello, world!
        <p>Hi!</p>
        What's up?
    </h1>
);
```

```json
{
  "node": "h1",
  "attributes": {
    "id": "attr"
  },
  "children": [
    "Hello, world!",
    {
      "node": "p",
      "attributes": null,
      "children": [
        "Hi!"
      ]
    },
    "What's up?"
  ]
}
```

```
{
  "type": "h1",
  "key": null,
  "ref": null,
  "props": {
    "id": "attr",
    "children": [
      "Hello, world!",
      {
        "type": "p",
        "key": null,
        "ref": null,
        "props": {
          "children": "Hi!"
        },
        "_owner": null,
        "_store": {}
      },
      "What's up?"
    ]
  },
  "_owner": null,
  "_store": {}
}
```


JSX → Virtual DOM™ → DOM

```
{
  "node": "h1",
  "attributes": {
    "id": "attr"
  },
  "children": [
    "Hello, world!",
    {
      "node": "p",
      "attributes": null,
      "children": [
        "Hi!"
      ]
    },
    "What's up?"
  ]
}
```

```
function render(virtualNode) {
  if (typeof virtualNode === 'string') {
    return document.createTextNode(virtualNode);
  }

  const element = document.createElement(virtualNode.node);

  Object.keys(virtualNode.attributes || {}).forEach(
    (attr) => element.setAttribute(attr, virtualNode.attributes[attr])
  );

  (virtualNode.children || []).forEach(
    (child) => element.appendChild(render(child))
  );

  return element;
}
```

```
const dom = render(hello);
document.body.appendChild(dom);
```


Babel + JSX
Result
Edit in JSFiddle
/** @jsx toVDOM */
const hello = (
<h1 id="attr">
Hello, world!
Hi!
What's up?
</h1>
);
function toVDOM(node, attributes, ...rest) {

JSX in the wild

	To Virtual DOM	**To DOM**
React	createElement	ReactDOM.render
Inferno	createVNode	render
Mithril	m	
Vue	h	
Preact		

blog.laroberto.com

</InANutshell>

Questions?

Thanks!