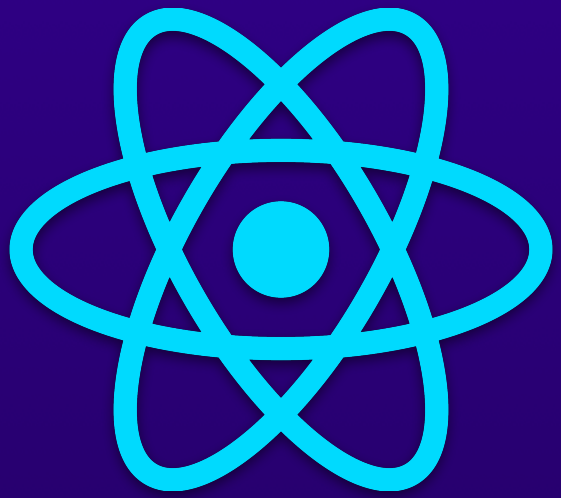
ACADEMIND

Component-Driven User Interfaces

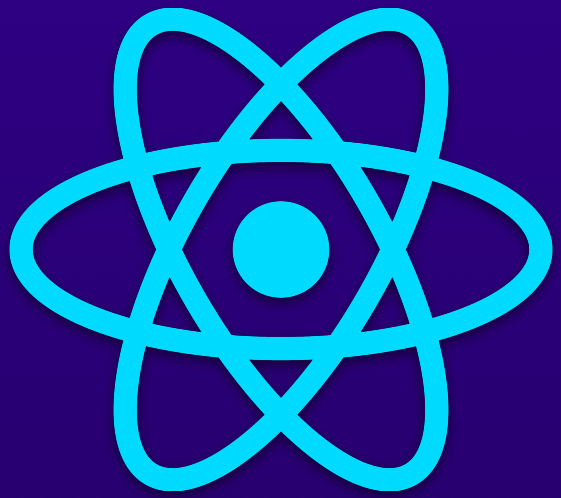
Building Interactive & Scalable UIs

- React Core Syntax & JSX
- Working with Components
- Working with Data

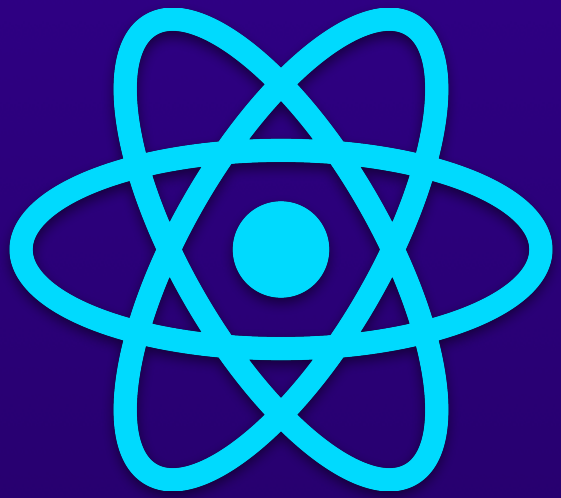


React is a JavaScript library for building user interfaces

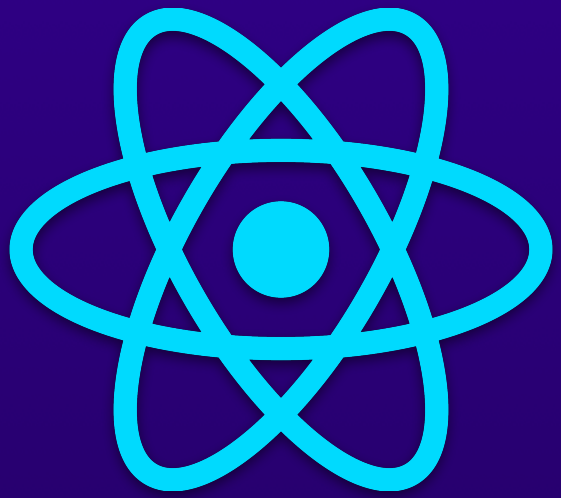
HTML, CSS & JavaScript are about building user interfaces as well



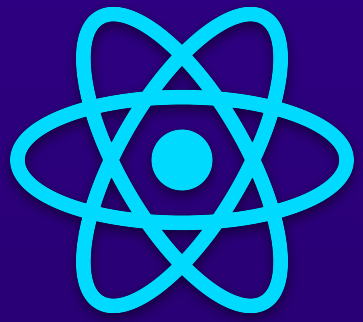
React makes building complex, interactive and reactive user interfaces simpler



React is all about Components



What is a Component?



React is all about Components

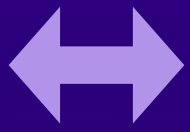
Because all user interfaces in the end are made up of components

ACADEMIND

Why Components?

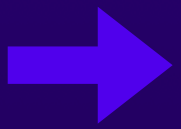
Reusability

Don't repeat yourself

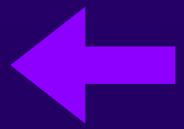


Separation of Concerns

Don't do too many things in one and the same place (function)

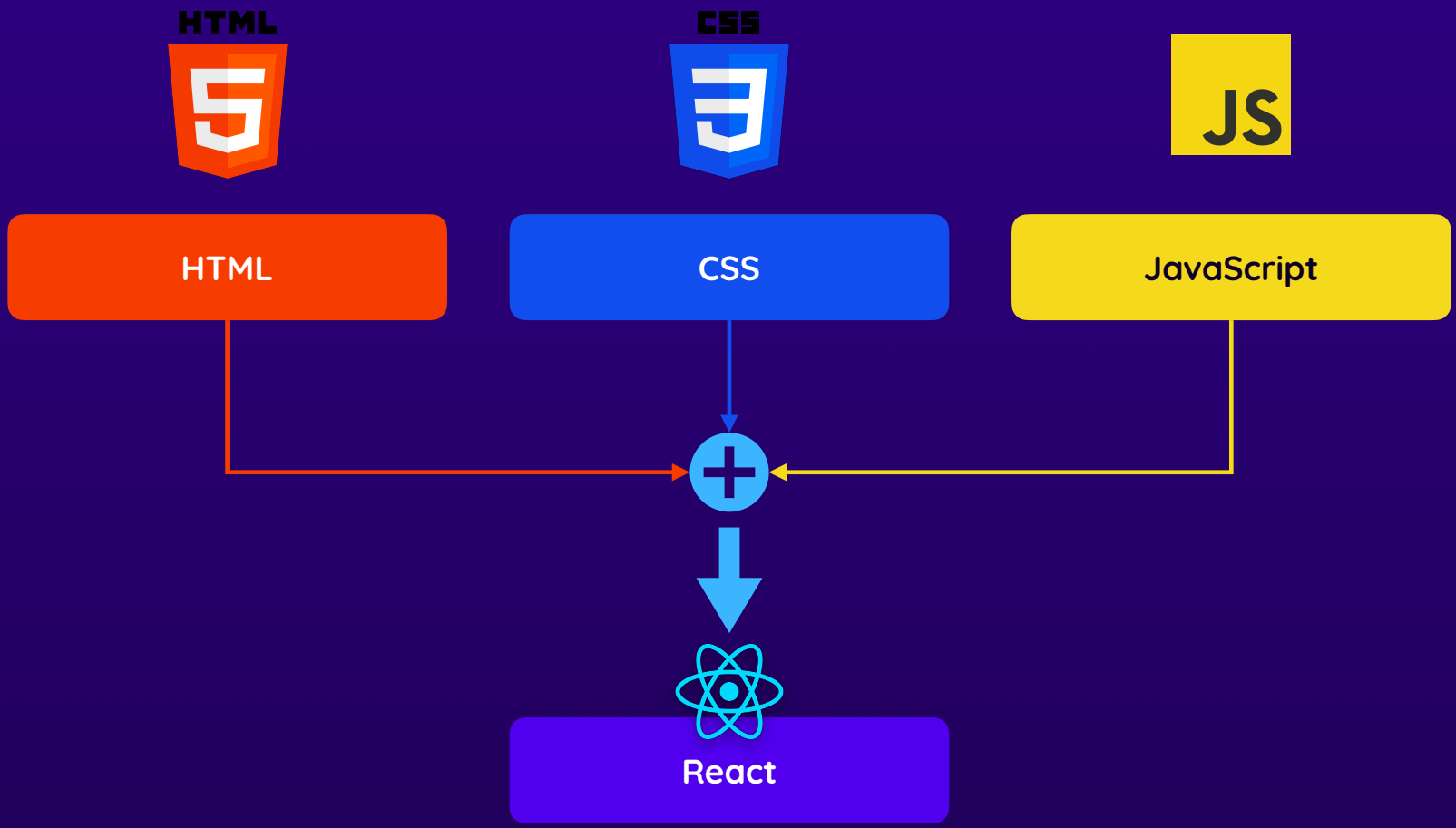


Split big chunks of code into multiple smaller functions



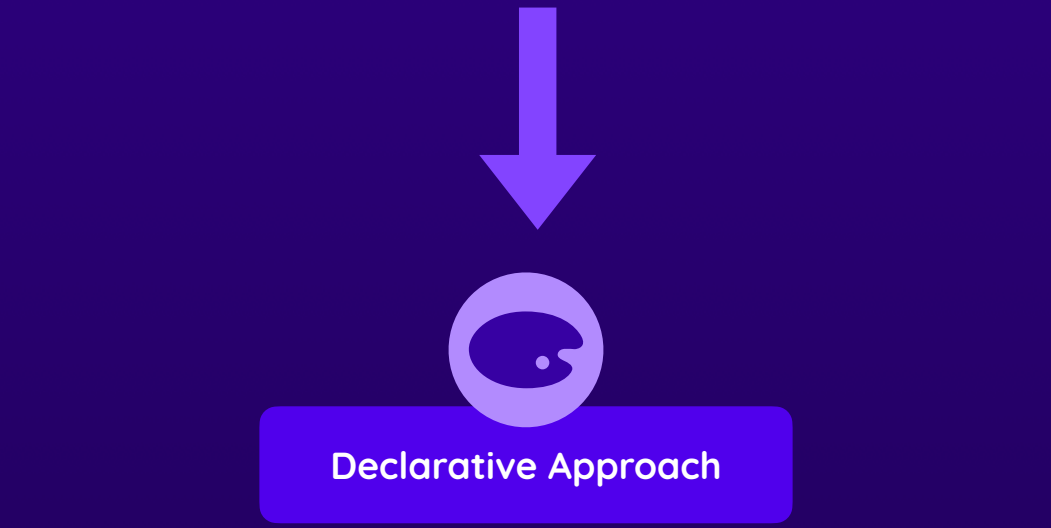
ACADEMIND

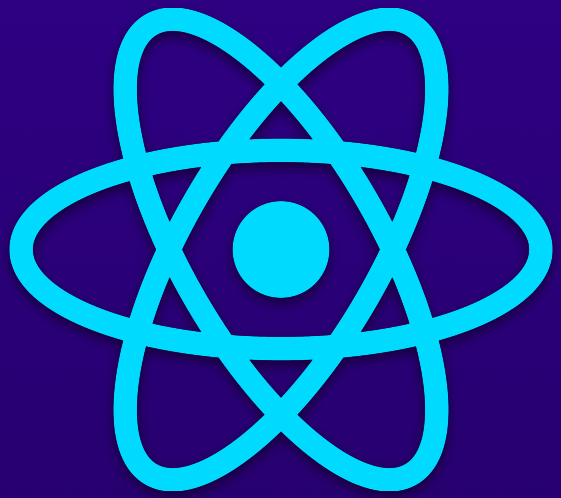
How Is A Component Built?



React & Components

React allows you to create **re-usable** and **reactive** components consisting of **HTML and JavaScript** (and CSS)





Build your own, custom HTML Elements

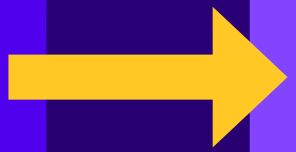
JSX = “HTML in JavaScript”

Understanding JSX

```
<p title="Intro text">

React.js is a library for
building user interfaces.

</p>
```

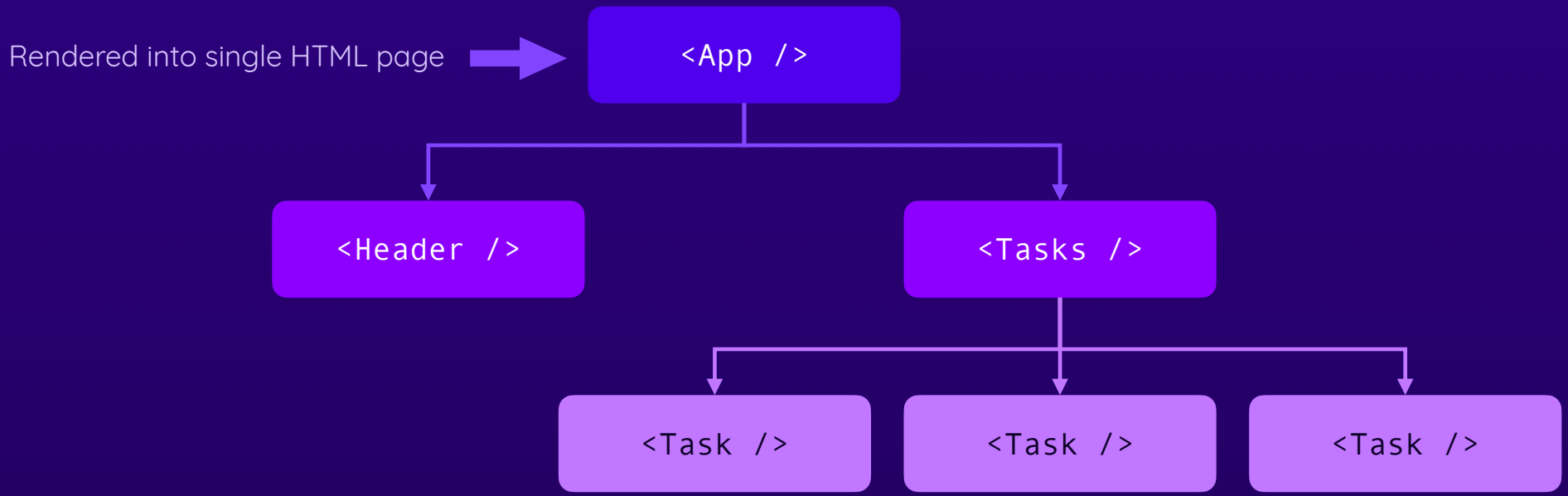


```
React.createElement(
  'p',
  {title: 'Intro text'},
  'React.js is a library for
  building user interfaces.'
);
```

"Syntactic sugar", **does not run** in the browser like this!

Real JavaScript code, would run in the browser like this. But not that nice to use.

You Build A Component Tree



Props are the “attributes” of your “custom HTML elements” (Components)

Passing Data via “Props”