React Js Online Training

sahosoft

Online Learning Platform

02

By: Chandan Kumar

JAVASCRIPT LIBRARY

JAVASCRIPT LIBRARY

- From the definition above, you can see how JavaScript plays a critical role in website and web application development. But there are times when you need JavaScript to perform repetitive functions—things like stock animation effects or autocomplete search bar features. Re-coding these functions every time they occur becomes a "reinventing the wheel" situation. Annoying. This is where JavaScript libraries come in.

- JavaScript libraries are collections of pre-written JavaScript code that can be used for common JS tasks, allowing you to bypass the time intensive (and unnecessary) process of coding by hand. If there's a run-of-the-mill JavaScript function that you keep needing to code (and that other developers before you have needed for their own projects) there's probably a JS library to ease your pain.

- Libraries are code written by someone else that is used to help solve common problems.

JAVASCRIPT LIBRARY

For example, let’s say you have a program where you plan on working with strings. You decide to keep your code DRY (don’t repeat yourself) and write some reusable functions like these:

```
function getWords(str) {
  const words = str.split(' ');
  return words;
}
function createSentence(words) {
  const sentence = words.join(' ');
  return sentence;
}
```

Congratulations. You’ve created a library.

- There isn’t anything magic about library. libraries are reusable code written by someone else. Their purpose is to help you solve common problems in easier ways.
- There are a lot of different JS libraries out there and React JS is one of them—but what makes it unique? What is React JS used for, and why should you learn React JS? And is it better than other JavaScript libraries?

Library VS Framework

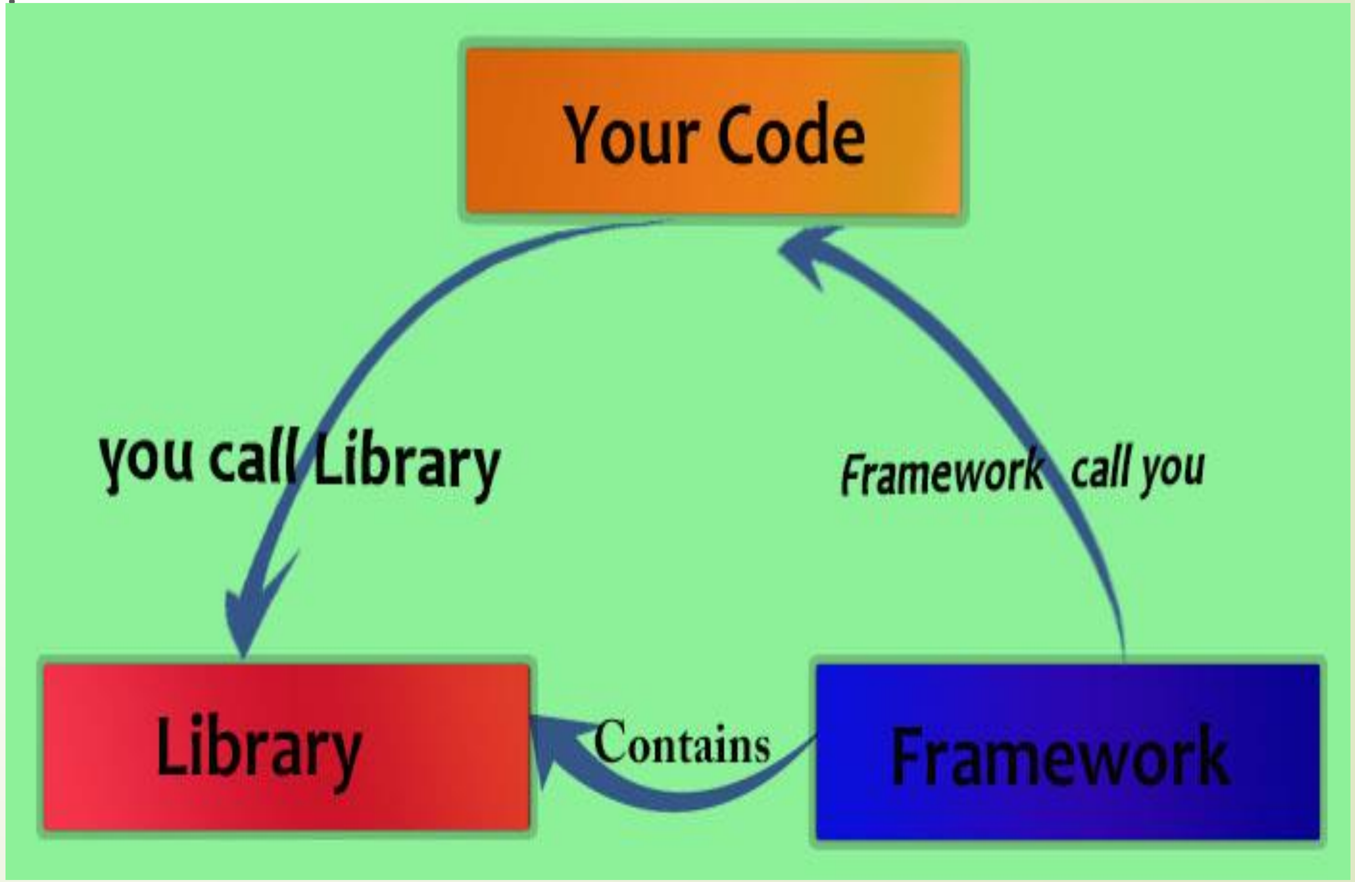
02

Library VS Framework

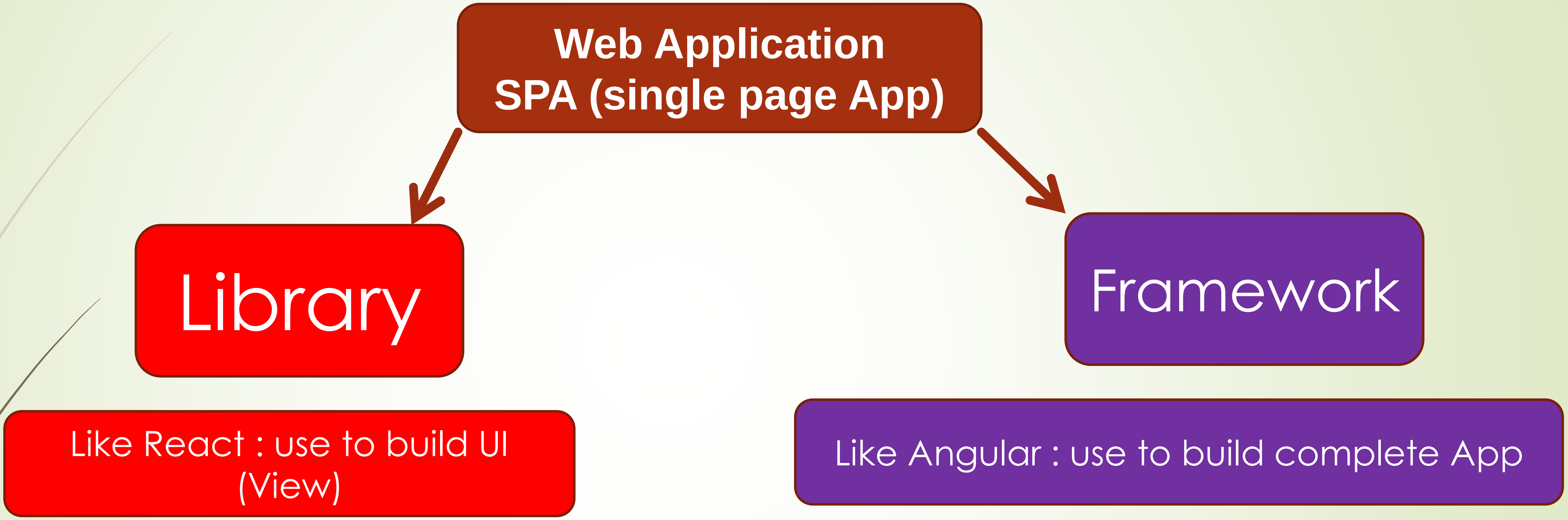
Both frameworks and libraries are code written by someone else that is used to help solve common problems.

The key difference between a library and a framework is "Inversion of Control".

When you call a method from a library, you are in control. But with a framework, the control is inverted: the framework calls you.



Library VS Framework



Web Application
SPA (single page App)

Library

Like React : use to build UI (View)

Framework

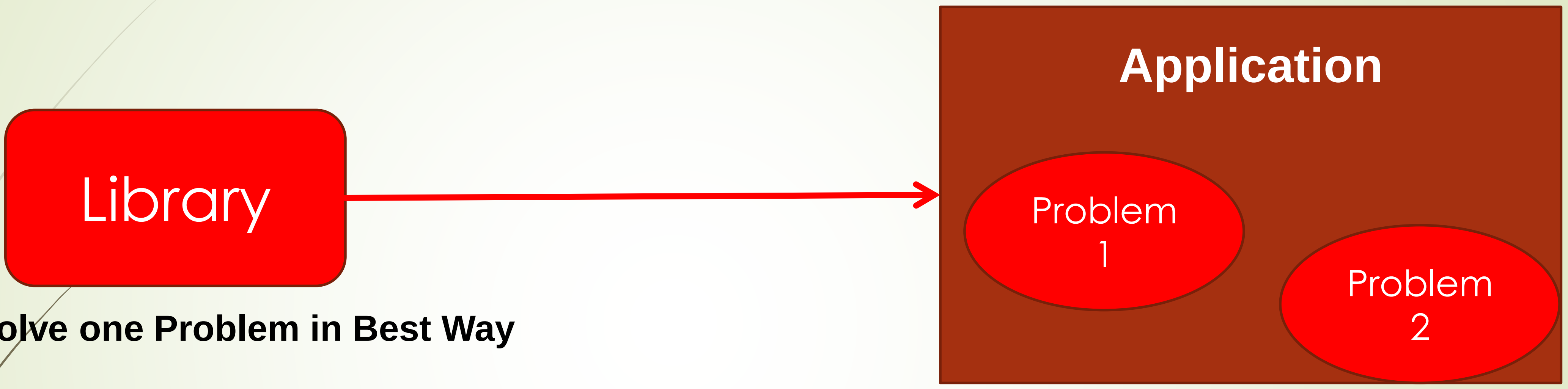
Like Angular : use to build complete App

Library VS Framework

React
View Library

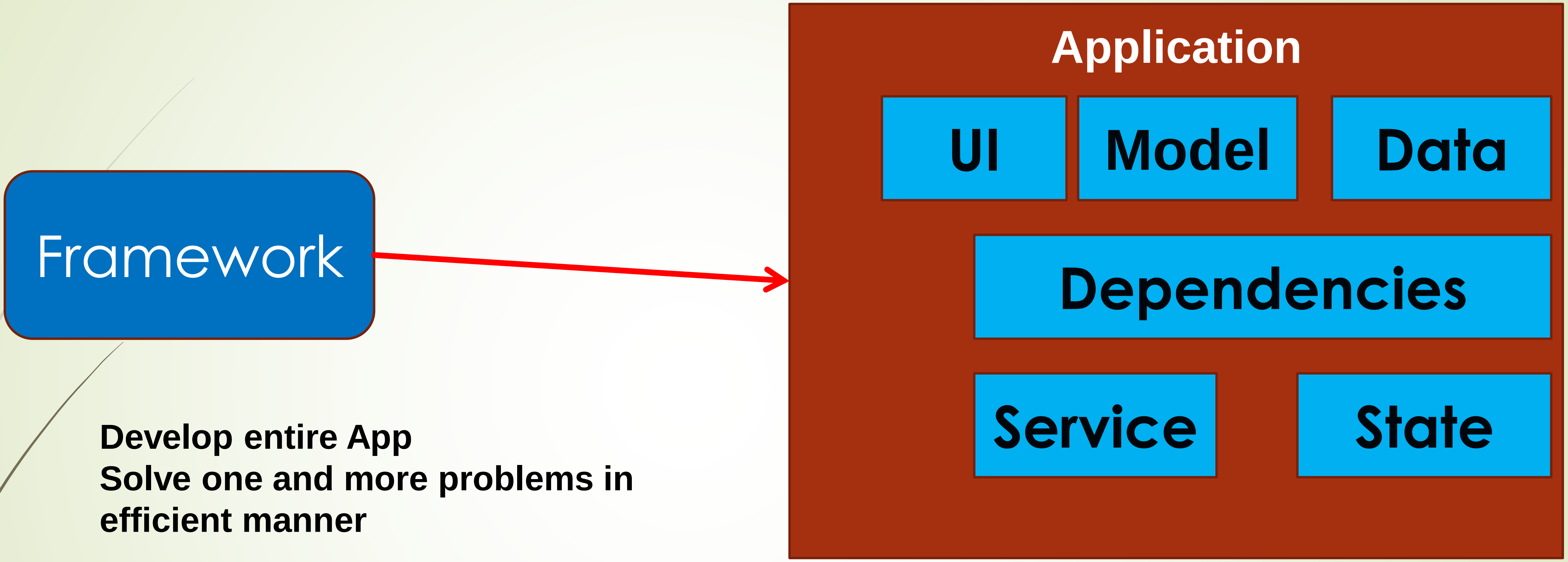
Angular
Framework

Library



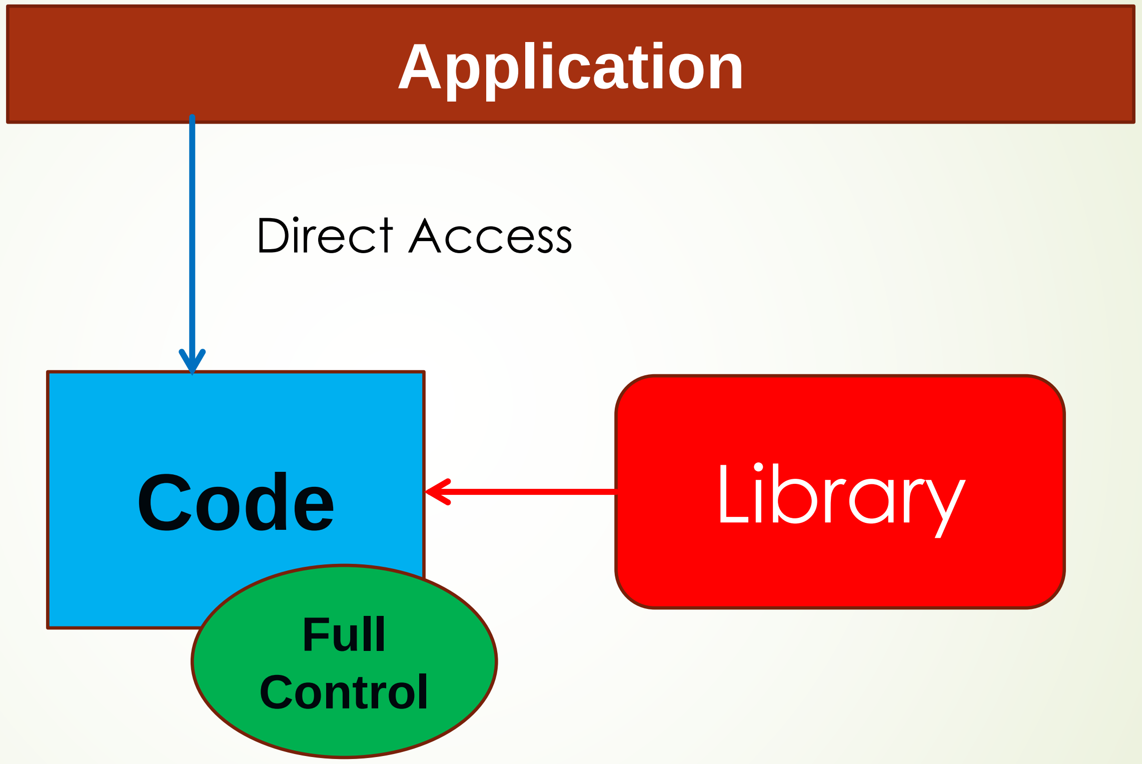
Solve one Problem in Best Way

Framework

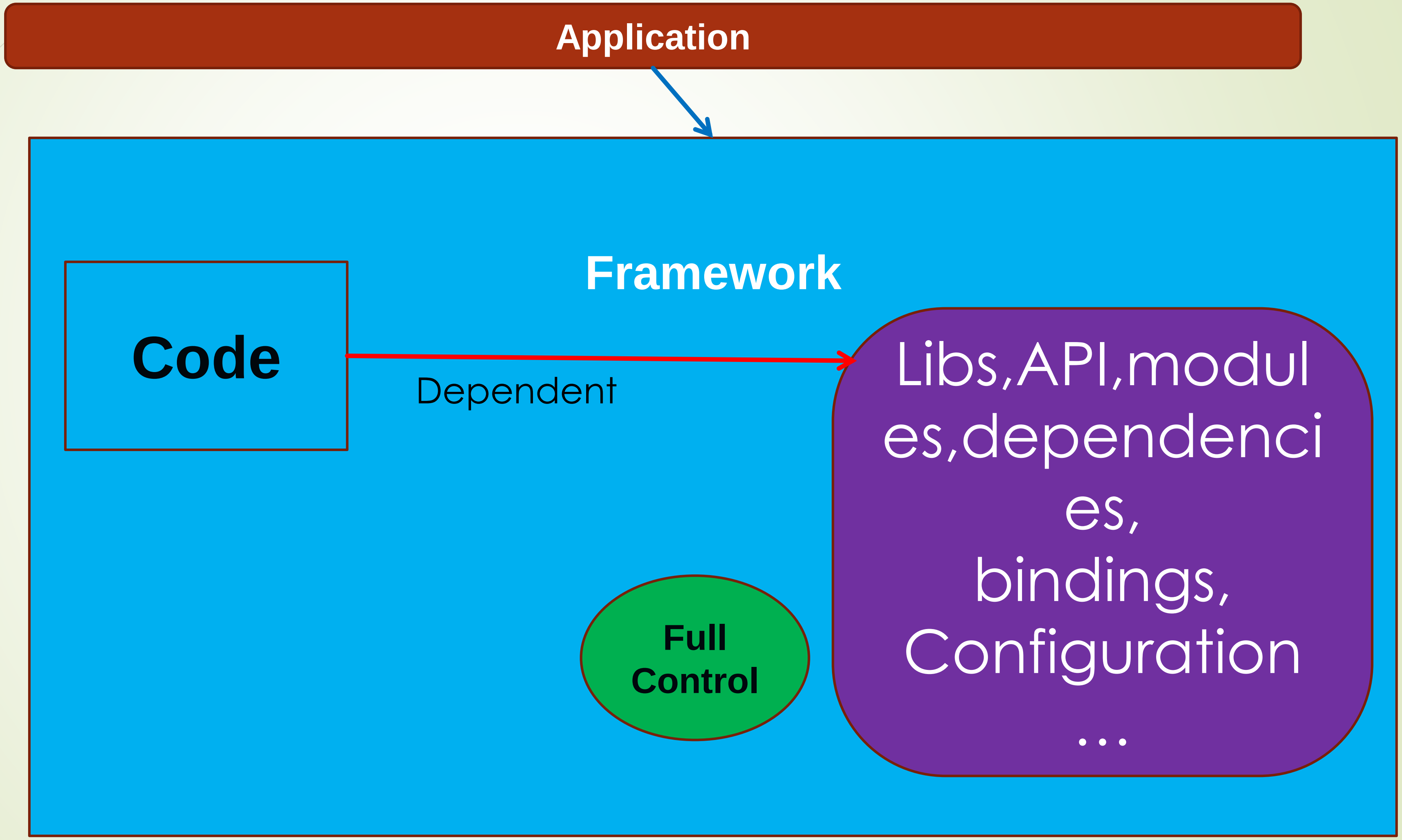


Develop entire App
Solve one and more problems in efficient manner

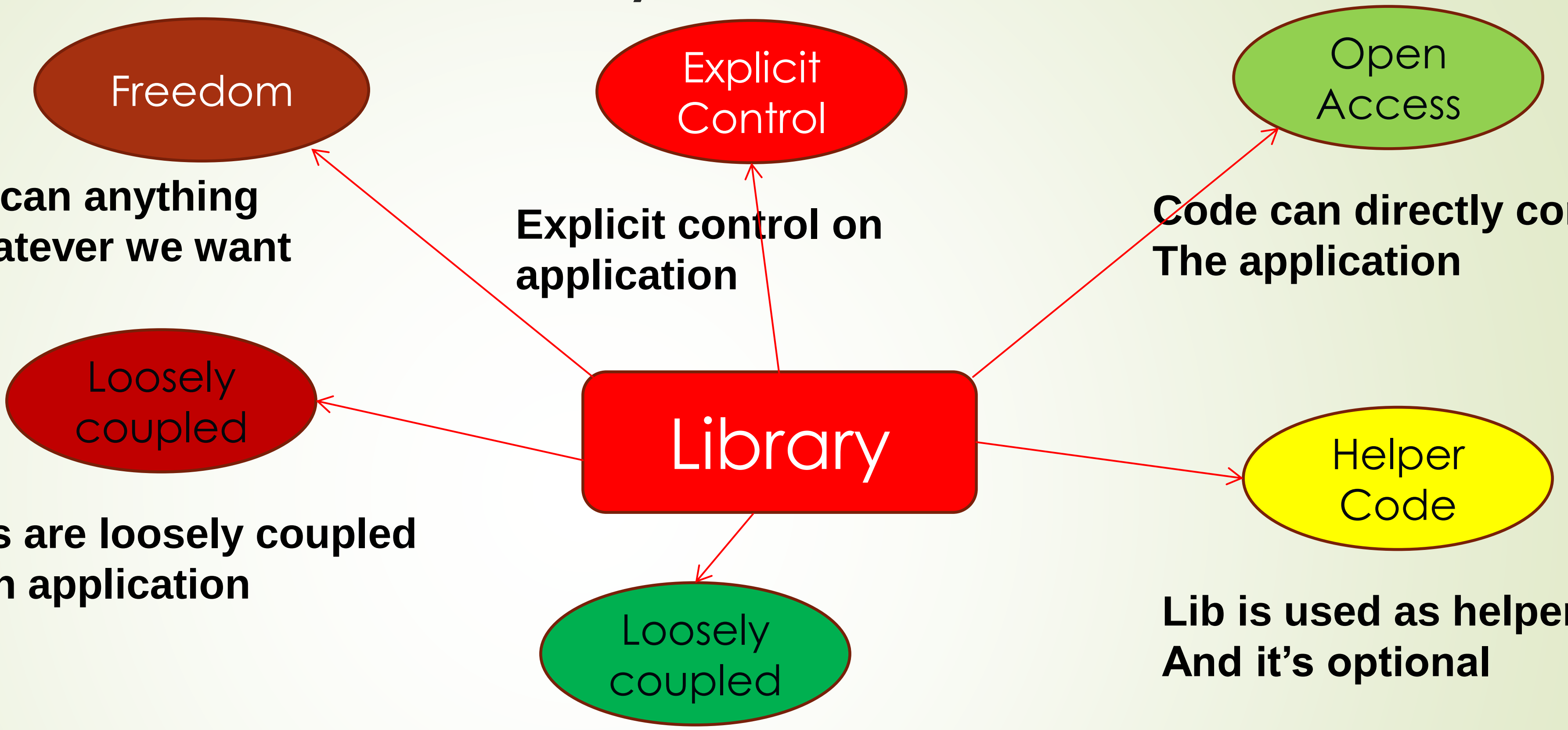
Library



Framework


Application
Framework
Code
Dependent
Libs,API,modules,dependencies,
bindings,
Configuration
...
Full Control

Library



We can anything
Whatever we want

Explicit control on
application

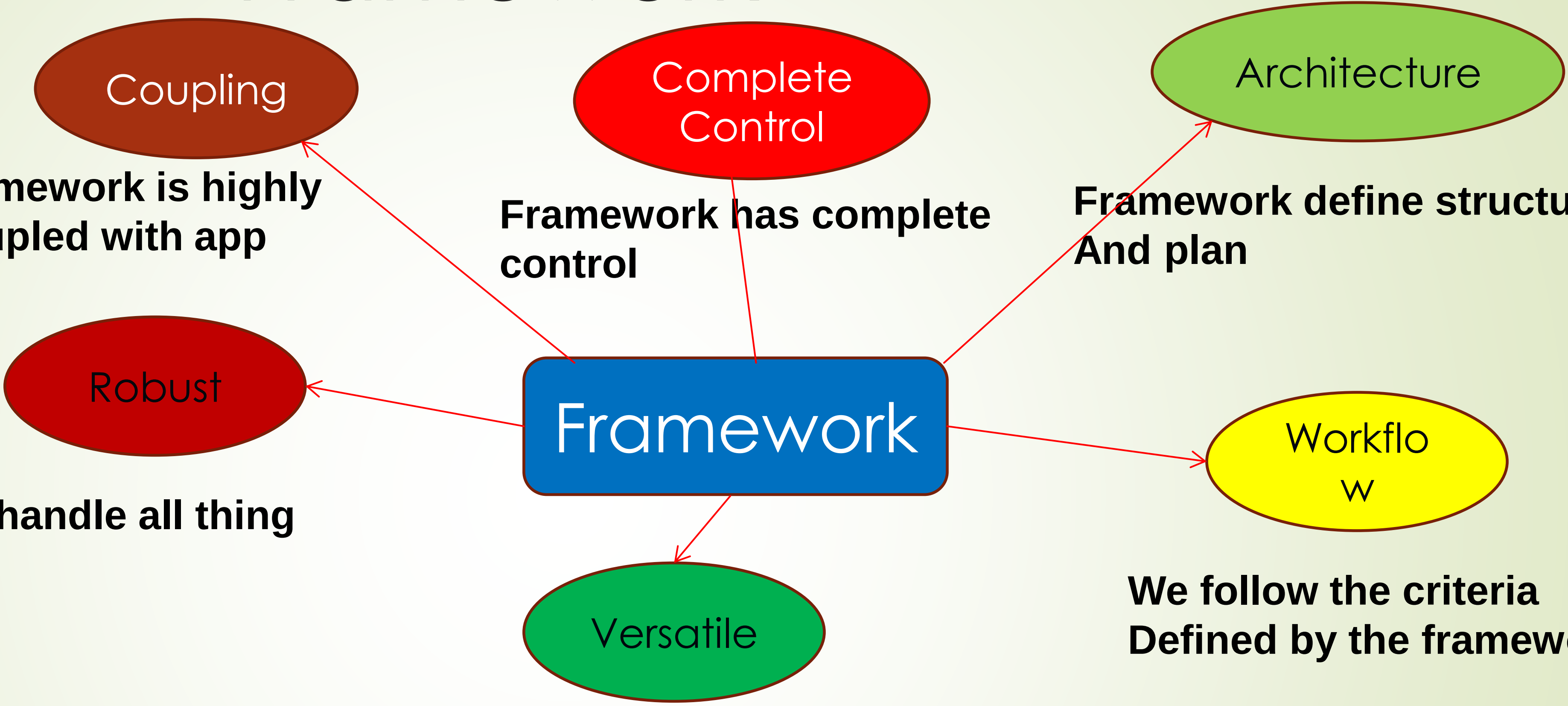
Code can directly control
The application

Libs are loosely coupled
With application

Lib is used as helper code
And it's optional

Libs are loosely coupled
With application

Framework



Framework is highly Coupled with app

Framework has complete control

Framework define structure And plan

It's handle all thing

We follow the criteria Defined by the framework

It's work more work than library

Library

Library

✓ There is less to learn
✓ Provide more freedom
✓ Workflow is highly flexible and less strict.
✓ It's provide functionality to integrate itself With your code

Framework

✓ There is more to learn
✓ Provide limited freedom
✓ Workflow is highly concrete and strict.
✓ Application is driven by framework, so we need to follow the guidelines of framework.

Library VS Framework

React
(tool)

Angular
(toolbox)

React VS Angular

02

React VS Angular

History of React vs. Angular

- Angular is a JavaScript framework written in TypeScript. It was developed and is maintained by Google, and is described as a “Super heroic JavaScript MVW(Model-View-Whatever) Framework” on Angular’s webpage. Angular 2 originally released in September 2016, is a complete rewrite of AngularJS (released in October 2010).
- React is a JavaScript library developed and maintained by Facebook. It was released in March 2013 and is described as “a JavaScript library for building user interfaces”.

React VS Angular

Framework vs. Library

- Angular and React have many similarities and many differences. One of them is that Angular is a full-fledged MVW framework and React is merely a JavaScript Library (just the view).
- Angular is considered a framework because it offers strong opinions as to how your application should be structured. It also has much more functionality "out-of-the-box". You don't need to decide which routing libraries to use or other such considerations – you can just start coding. However, a drawback is that you have less flexibility – you must use what Angular provides.
- Angular provides the following "out of the box":

React VS Angular

Architecture of React vs. Angular

- Templates, based on an extended version of HTML
- XSS protection
- Dependency injection
- Ajax requests by @angular/HTTP
- Routing, provided by @angular/router
- Component CSS encapsulation
- Utilities for unit-testing components.
- @angular/forms for building forms

React VS Angular

Architecture of React vs. Angular

- React, on the other hand, gives you much more freedom. It only provides the "view" in M**V**C – you need to solve the M and C on your own. Due to this, you can choose any of your own libraries as you see fit. You will end up using many independent, fast-moving libraries. Because of this, you will need to take care of the corresponding updates and migrations by yourself. In addition, each React project is different and requires a decision requiring its folder hierarchy and architecture. Things can go wrong much more easily due to this.
- React provides the following "out of the box":

  Instead of classic templates, it has JSX, an XML-like language built on top of JavaScript

  XSS protection

  No dependency injection

  Fetch for Ajax requests

  Utilities for unit-testing components
- Some popular libraries to add functionality are:

  React-router for routing

  Redux or MobX for state management

  Enzyme for additional testing utilities

React VS Angular

Architecture of React vs. Angular

- **Regular DOM vs. Virtual Dom**

React's use of a **virtual DOM** is one of its features that makes it so blazingly fast. You've probably heard of it. It was React's "killer feature" when it was first released. Let me give you an example scenario:

Let's say that you want to update a user's age within a block of HTML tags. A **virtual DOM** only looks at the differences between the previous and current HTML and changes the part that is required to be updated. Git employs a similar method, which distinguishes the changes in a file.

Conversely, Angular opted to use a regular DOM. This will update the entire tree structure of HTML tags until it reaches the user's age.

So why does this matter? The example above is trivial and probably won't make any difference in a real app. However, if we're dealing with hundreds of data requests on the same page (and the HTML block is replaced for every page request) it drastically affects the performance, in addition to the user's experience.

React VS Angular

Architecture of React vs. Angular

- Templates – JSX or HTML

React decided to combine UI templates and inline JavaScript logic, which no company had ever done before. The result is called "**JSX**".

Although it may have sounded like a bad idea, Facebook's gamble paid off big-time.

React uses something called a component, which contains both the markup AND logic in the same file. It also uses an XML-like language that allows you to write markup directly in your JavaScript code. JSX is a big advantage for development, because you have everything in one place, and code completion and compile-time checks work better.

Angular uses templates that are enhanced HTML with Angular directives ("ng-if" or "ng-for").

React only requires knowledge of JavaScript, but with Angular, you must learn its specific syntax like directives, pipes, databinding etc.

React VS Angular

Architecture of React vs. Angular

- Components

Both **React** and **Angular** are **component-based**. A component receives an input, and after some internal logic returns a rendered UI template (a sign-in form or a table for example) as output. Components should be **easy to reuse** within other components or even in other projects.

For example, you could have a sign-in component consisting of two text inputs (user & password) and a "Login" button. This component may have various properties and underlying logic, but it should be generalized so that you can reuse the component with different data on another page or in another app.

Components are meant to be self-contained "chunks" of your app that you can reuse in different situations. They are meant to encapsulate logic.

React VS Angular

Architecture of React vs. Angular

- State Management

There are **states** everywhere in an application. Data morphing over time involves complexity. The UI is described by the component at a given point in time. Then, the framework re-renders the entire UI of the component when data changes. This ensures that the data is always up to date.

To handle state in React, Redux is often used as the solution. In Angular, you may not need **Redux**. But, if your application becomes large enough, chances are that you will.

- Data Binding

A large difference between React and Angular is **one-way vs. two-way binding**. Angular uses **two-way** binding.

For example, if you change the UI element (a user input) in **Angular**, then the corresponding model state changes as well. Additionally, if you change the model state, then the UI element changes – hence, **two-way data binding**.

However, **React** only has **one-way binding**. First, the model state is updated, and then it renders the change in the UI element. However, if you change the UI element, the model state DOES NOT change. You must figure that out for yourself. Some common ways are through **callbacks** or **state management libraries**.

React VS Angular

Architecture of React vs. Angular

- Mobile Solutions of React vs. Angular

Angular and **React** both offer solutions to create mobile applications.

Ionic is a framework for developing hybrid mobile applications. It uses a Cordova container that is incorporated with Angular. Ionic provides a robust UI component library that is easy to set up and develop hybrid mobile applications with. However, the resulting app on a device is simply a web app inside of a native web view container. Because of this, the apps can be slow and laggy

React Native, on the other hand, is a platform developed by Facebook for creating truly native mobile applications using React. The syntax is slightly different, but there are much more similarities than differences. Unlike Ionic, which is simply a glorified web app, React Native produces a truly native UI. It also allows you to create your own components and bind them to native code written in Objective-C, Java, or Swift.

React VS Angular

Architecture of React vs. Angular

- Testing in React vs. Angular

Jest is used by **Facebook** to tests its **React code**. It is included in every React project and requires zero configuration to use. It also includes a powerful mocking library. Many times Jest is used in combination with Enzyme (a JavaScript testing utility used at Airbnb).

Jasmine is a testing framework that can be used in **Angular**. Eric Elliott says that Jasmine "results in millions of ways to write tests and assertions, needing to carefully read each one to understand what it's doing".

React VS Angular

React

Architecture of React vs. Angular

- Learning Curve of React vs. Angular

An important decision you must make in choosing a new technology is its learning curve. The answer depends on your previous experience and familiarity with the related concepts. However, we can still try to assess the number of new things you'll need to learn before you get started:

React:

The first thing you'll learn in React is JSX. It may seem awkward to write at first, but it doesn't add much complexity. You'll also need to learn how to write components, manage internal state, and use props for configuration. You don't need to learn any new logical structures or loops since all of this is plain JavaScript.

Once you're done learning the basics, you'll need to learn a routing library (since React doesn't come with one). Next comes state management with Redux or MobX. Once you've learned the basics, a routing library, and state management library, you're ready to start building apps!

React VS Angular

Architecture of React vs. Angular

- Learning Curve of React vs. Angular

Angular:

Angular has many topics to learn, starting from basic ones such as directives, modules, decorators, components, services, dependency injection, pipes, templates etc. After that, there are more advanced topics such as change detection, AoT compilation, and Rx.js.

The entry barrier for Angular is clearly higher than for React. The sheer number of new concepts is confusing to newcomers. And even after you've started, the experience might be a bit rough since you need to keep in mind things like Rx.js subscription management and change detection performance.

It may seem like React has a lower barrier for entry. However, that doesn't mean that React is "better". I encourage you to try both React and Angular to see which one you personally prefer.

React VS Angular

Architecture of React vs. Angular

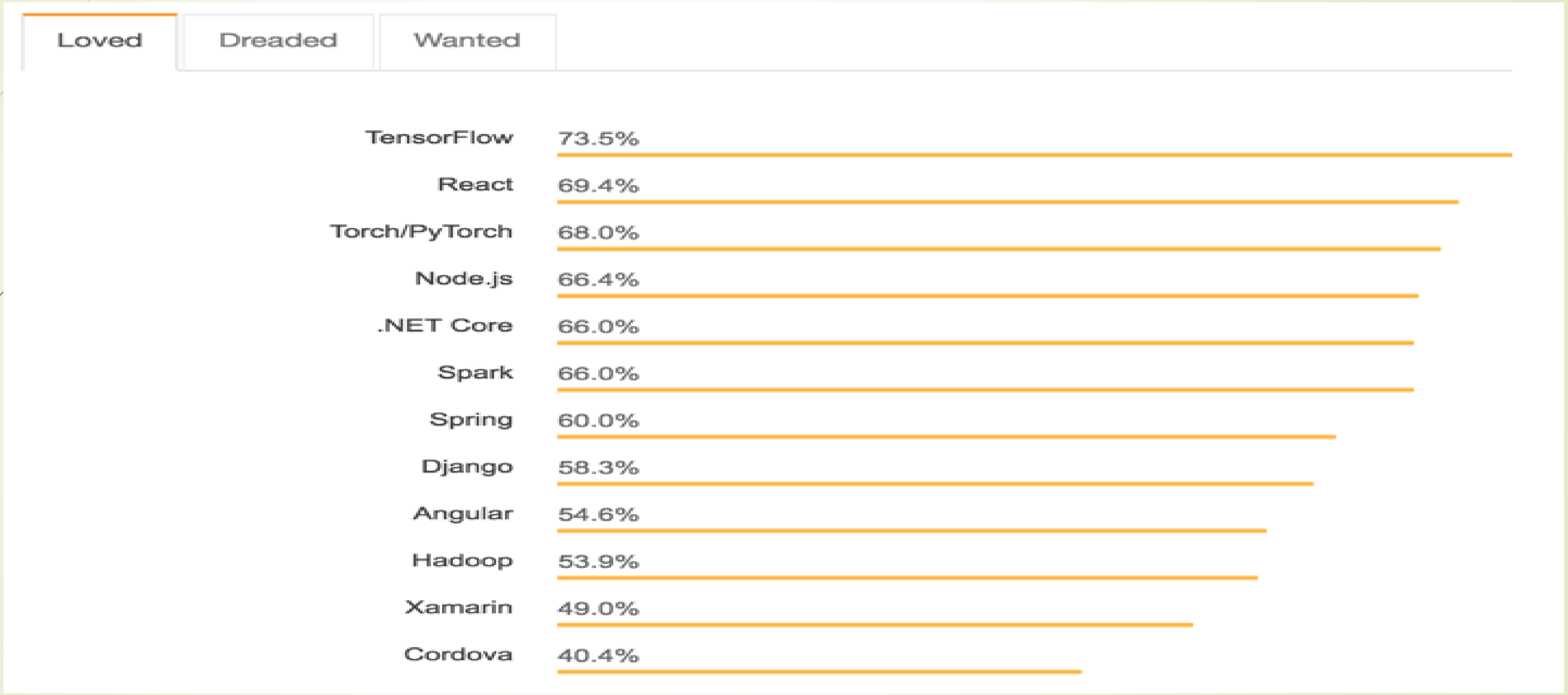
- Popularity & Growth Trajectory   React vs. Angular

Before you choose a framework/library to work with, it's important to look at its popularity for the sole purpose of job prospects. The more popular a technology is, in most cases, the more jobs you can find.

- Most Loved, Dreaded, and Wanted Frameworks, Libraries, and Tools

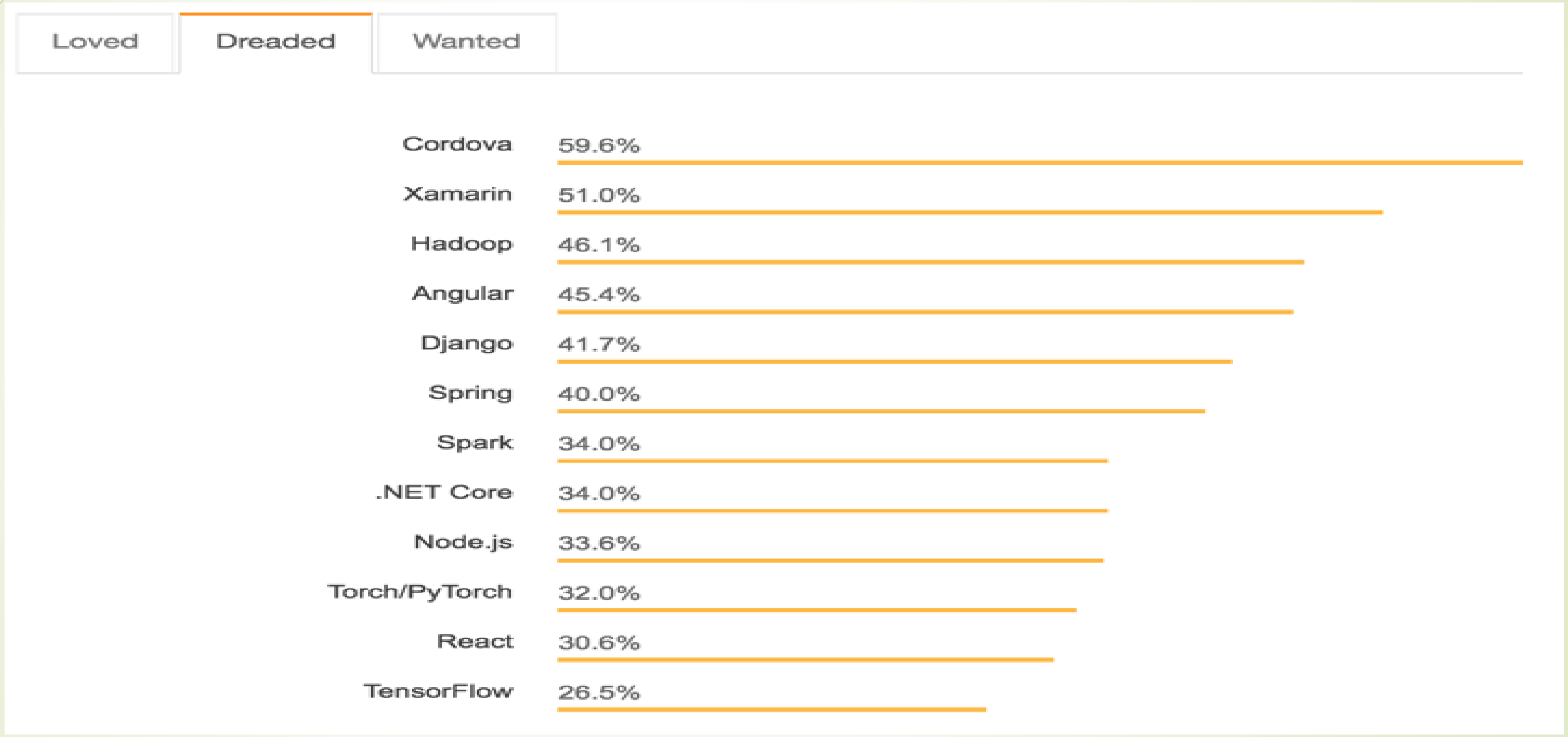
Another important statistic you should look at is the percentage of developers that love, dread, and want to learn a specific technology. A few statistics from Stack Overflow's 2019 Developer Survey can be found below. React is the 2nd most loved technology. Angular is ranked far lower, in 9th place.

React VS Angular



TensorFlow	73.5%
React	69.4%
Torch/PyTorch	68.0%
Node.js	66.4%
.NET Core	66.0%
Spark	66.0%
Spring	60.0%
Django	58.3%
Angular	54.6%
Hadoop	53.9%
Xamarin	49.0%
Cordova	40.4%

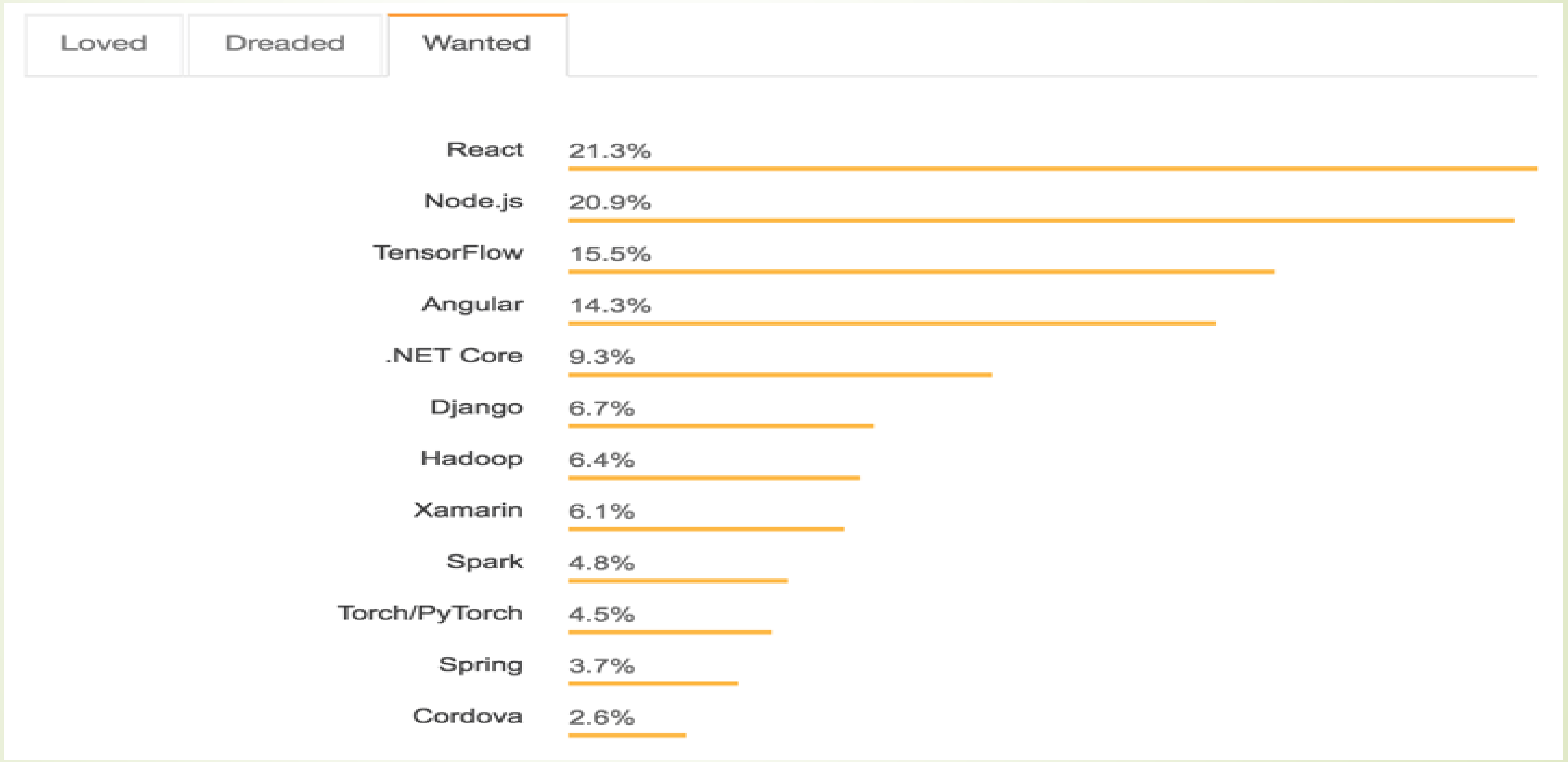
React VS Angular



Loved	Dreaded	Wanted

Cordova	59.6%
Xamarin	51.0%
Hadoop	46.1%
Angular	45.4%
Django	41.7%
Spring	40.0%
Spark	34.0%
.NET Core	34.0%
Node.js	33.6%
Torch/PyTorch	32.0%
React	30.6%
TensorFlow	26.5%

React VS Angular



React	21.3%
Node.js	20.9%
TensorFlow	15.5%
Angular	14.3%
.NET Core	9.3%
Django	6.7%
Hadoop	6.4%
Xamarin	6.1%
Spark	4.8%
Torch/PyTorch	4.5%
Spring	3.7%
Cordova	2.6%

React VS Angular

Companies Using

HUGE companies are utilizing both React and Angular. I’m talking some of the biggest in the world. Here is just a small sample:

React
- Facebook
- Airbnb
- Uber
- Netflix
- Instagram
- WhatsApp
- Dropbox

Angular
- Google
- Nike
- Forbes
- Upwork
- General Motors
- HBO
- Sony

Why React ? | Features of React

02

Why React ? | Features of React

There are so many open-source platforms for making the front-end web application development easier, like Angular. Let us take a quick look on the benefits of React over other competitive libraries or frameworks.

- Simple

React JS is easy to learn and implement (only if you have a basic understanding of JavaScript). It has a well-defined lifecycle. Everything is component based here. It is simple in comparison to other frameworks.

- Easy to learn

In technology, anything easy to learn and understand is always easy to implement. Any developer can easily understand the syntax and components of React. You should have a basic knowledge of HTML, CSS, and JavaScript.

Anyone with a basic previous knowledge in programming can easily understand React while Angular and Ember are referred to as 'Domain-specific Language', implying that it is difficult to learn them. To react, you just need basic knowledge of CSS and HTML.

- Native Approach

React can be used to create mobile applications (React Native). And React is known for its reusability. You can extensively use the code and that's the plus feature of React. You can create mobile applications on iOS, Android and also the web applications.

- Data Binding

React uses one-way data binding. Also, it is easier to debug ReactJS components of large ReactJS apps. It has one directional flow of data and it uses flux architecture.

Why React ? | Features of React

- Performance

React uses Virtual DOM, thereby creating web applications faster. Virtual DOM compares the components' previous states and updates only the items in the Real DOM that were changed, instead of updating all of the components again, as conventional web applications do.

ReactJS is fast and secured. Also, it has not much of the performance issues. It does not offer any built-in container for dependency injection.

You can use Browserify, Require JS, EcmaScript 6 modules which we can use via Babel, ReactJS-di to inject dependencies automatically.

- Easy creation of dynamic applications

React makes it easier to create dynamic web applications because it requires less coding and offers more functionality, as opposed to JavaScript, where coding often gets complex very quickly.

- Reusable components

Components are the building blocks of any React application, and a single app usually consists of multiple components. These components have their logic and controls, and they can be reused throughout the application, which in turn dramatically reduces the application's development time.

- Testability

ReactJS applications are easily testable. You can easily set triggers, events, functions and debug the code. You can easily debug the code.

Why React ? | Features of React

- Dedicated tools for easy debugging:

Facebook has released a Chrome extension that can be used to debug React applications. This makes the process of debugging React web applications faster and easier.

- It has SEO Friendly features:

Whenever there is a common search engine failure to read JavaScript at that time ReactJS runs on a server, renders and returns virtual DOM to browser through a regular webpage. uses of ReactJs has this ability to deal with failures efficiently.

- A helpful developer toolset:

Another important uses of React JS is a user-friendly development platform. The two major tools provided are React to Developer Tools and Redux Developer Tools. Both these tools can be installed as Chrome extensions.

React developer tool is used for inspecting react components that lie in a hierarchy and are used for introspecting current properties and states. It also helps to view component hierarchies, discover child and parent components as well.

When using Redux you can observe various dispatched actions, current store states and check changes on stores. The different dispatch actions or modify stores with different changes can also be reflected and viewed instantly. A user can also record and go back to any previous states of the application for debugging purposes.

Why React ? | Features of React

- It is backed by a strong community:

Initially, React library was created for internal use and later shared with the entire world. Currently, it is supported by Facebook and Instagram engineering teams, plus external experts. For example, React GitHub repository numbers over 1377 contributors, while users can ask their questions on Stack Overflow, Discussion forum, Reactiflux Chart, Freenode IRC, social media platforms and many others.

Disadvantages of React JS Limitations of React

02

Disadvantages of React JS Limitations of React

- Limitations of React

Most of the code is written in JSX, i.e., Html and CSS are part of JavaScript, it can be quite confusing as most other frameworks prefer keeping Html separate from the JavaScript code.

The file size of React JS is large.

React only focuses on the View section of an application so, if you are developing a large application then you have to include some other libraries to handle the other parts of applications.

Fewer features compared to any monolithic framework such as AngularJS. For example, in Angular routing is inbuilt but in React you have to include some external files to perform the routing.

React uses JSX , so Babel is used to transpile the JSX into plane JavaScript and Webpack & npm is used to build and run the application. If you are not familiar with these technologies and you get any error into code then it will be difficult for you to handle and resolve the error.

Disadvantages of React JS Limitations of React

- Limitations of React

Extra SEO Problem: There are issues that Google and different search engines can’t index or poorly index dynamic sites with client-side rendering. These issues haven’t been absolutely established and there are exposure materials around. Google itself confirmed that their crawlers are capable of reading dynamic content. So, we tend to aren’t gonna say that your ReactJS app won’t be indexed by Google. It’s 2018 in spite of everything.

HTML in JavaScript!’ – JSX as a barrier:ReactJS uses JSX. It’s a syntax extension, that permits commixture HTML with JavaScript. JSX has its own advantages (for instance, protective code from injections), however, some members of the event community think about JSX to be a significant disadvantage. Developers and designers complain regarding JSX’s quality and resultant steep learning curve.

React

DOM VS Virtual DOM

sahosoft
Online Learning Platform

DOM VS Virtual DOM

DOM

Just to get things straight - DOM stands for Document Object Model and is an abstraction of a structured text. For web developers, this text is an HTML code, and the DOM is simply called HTML DOM. Elements of HTML become nodes in the DOM.

So, while HTML is a text, the DOM is an in-memory representation of this text.

The HTML DOM provides an interface (API) to traverse and modify the nodes. It contains methods like **getElementById** or **removeChild**. We usually use JavaScript language to work with the DOM.

So, whenever we want to dynamicly change the content of the web page, we modify the DOM:

Javascript

- var item = document.getElementById("myLI");
- item.parentNode.removeChild(item);

document is an abstraction of the root node, while **getElementById, parentNode and removeChild** are methods from HTML DOM API.

DOM VS Virtual DOM

DOM Issues

The DOM in simple words represents the UI of your application. Every time there is a change in the state of your application UI, the DOM gets updated to represent that change. Now the catch is frequently manipulating the DOM affects performance, making it slow.

The HTML DOM is always tree-structured - which is allowed by the structure of HTML document. This is cool because we can traverse trees fairly easily. Unfortunately, easily doesn't mean quickly here.

The DOM trees are huge nowadays. Since we are more and more pushed towards dynamic web apps (Single Page Applications - SPAs), we need to modify the DOM tree incessantly and a lot. And this is a real performance and development pain.

DOM VS Virtual DOM

Virtual DOM

First of all - the Virtual DOM was not invented by React, but React uses it and provides it for free.

That's where the concept of virtual DOM comes in and performs significantly better than the real DOM. The virtual DOM is only a virtual representation of the DOM. Every time the state of our application changes, the virtual DOM gets updated instead of the real DOM.

Well, you may ask" Isn't the virtual DOM doing the same thing as the real DOM, this sounds like double work? How can this be faster than just updating the real DOM?"

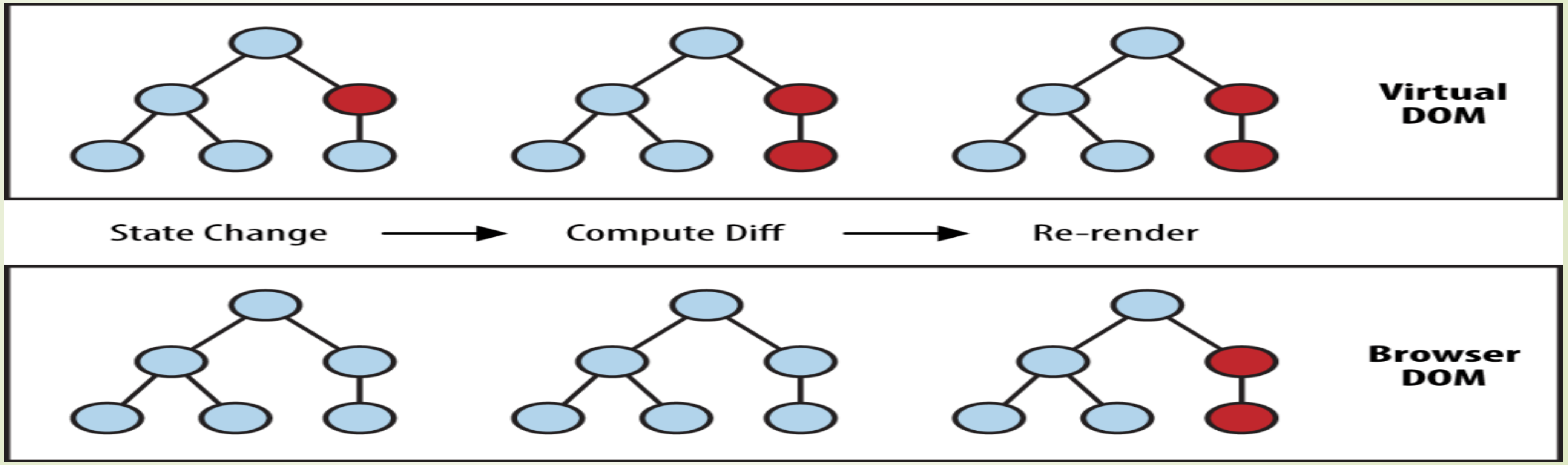
The answer is virtual DOM is much faster and efficient, here is why.

DOM VS Virtual DOM

How is Virtual DOM faster?

When new elements are added to the UI, a virtual DOM, which is represented as a tree is created. Each element is a node on this tree. If the state of any of these elements changes, a new virtual DOM tree is created. This tree is then compared or “diffed” with the previous virtual DOM tree.

Once this is done, the virtual DOM calculates the best possible method to make these changes to the real DOM. This ensures that there are minimal operations on the real DOM. Hence, reducing the performance cost of updating the real DOM.



DOM VS Virtual DOM

How does React use Virtual DOM

Now that you have a fair understanding of what a Virtual DOM is, and how it can help with performance of your app, let's look into how React leverages the virtual DOM.

In React every UI piece is a component, and each component has a state. React follows the observable pattern and listens for state changes. When the state of a component changes, react updates the virtual DOM tree. Once the virtual DOM has been updated, react then compares the current version of the virtual DOM with the previous version of the virtual DOM. This process is called "diffing".

Once React knows which virtual DOM objects have changed, then React updates only those objects, in the real DOM. This makes the performance far better when compared to manipulating the real DOM directly. This makes React standout as a high-performance JavaScript library.

In simple words, you tell React what state you want the UI to be in, and it makes sure that the DOM matches that state. The great benefit here is that as a developer, you would not need to know how the attribute manipulation, event handling or the manual DOM updates happen behind the scenes.

DOM VS Virtual DOM

How does React use Virtual DOM

All of these details are abstracted away from React developers. All you need to do is update the states of your component as and when needed and React takes care of the rest. This ensures a superior developer experience when using React.

Batch Update

React follows a batch update mechanism to update the real DOM. Hence, leading to increased performance. This means that updates to the real DOM are sent in batches, instead of sending updates for every single change in state.

The repainting of the UI is the most expensive part, and React efficiently ensures that the real DOM receives only batched updates to repaint the UI.