

# Introduction to Vue.js

**Bok, Jong Soon**  
**[javaexpert@nate.com](mailto:javaexpert@nate.com)**  
**<https://github.com/swacademy/Vue.js>**

# What is JavaScript?

- Formerly *LiveScript*.
- Netscape's simple, cross-platform, World-Wide-Web scripting language.
- Is intimately tied to the World-Wide-Web.
- Currently runs in only three environments.
  - As a server-side scripting language.
  - As an embedded language in server-parsed HTML.
  - As an embedded language run in Web browsers.

## What is JavaScript? (Cont.)

- Is a lightweight programming language.
- Is programming code that can be inserted into HTML pages.
- JavaScript code can be executed by all modern web browsers.
- Is easy to learn.

# JavaScript History

- Was originally developed in Netscape, by Jared Russell.
- Developed under the name *Mocha* Project(LiveConnect is Server-Side tech.)
- Hires Brendan Eich at MicroUnity Systems Engineering
  - Role to new programming language design & implementation.
- *LiveScript* was the official name for the language in beta releases of Netscape Navigator 2.0 in September 1995.

## JavaScript History (Cont.)

- Release Java Language in 1995 at SUN Microsystems.
- Renamed JavaScript on December 4, 1995 at Netscape browser version 2.0B3.
- MS releases IE & VBScript against Netscape → Jscript
- Jscript was included in IE 3.0, released in August 1996.
- Occurs cross-browser compatibility problems.
- Netscape toward to standardization.
- Report on JavaScript Specification to ECMA Conference on November, 1996.
- Releases ECMA-262 in 1997.

# ECMAScript(or ES)

- Is a scripting-language specification standardized by Ecma International in ECMA-262 and ISO/IEC 16262.
- Was created to standardize JavaScript, so as to foster multiple independent implementations.
- <http://ecma-international.org/>
- [ES5](#), 3 December 2009
- [ES6](#), June 2015
- [ES7](#), June 2016
- [ES8](#), June 2017
- [ES9](#), June 2018
- [ES10](#), June 2019

# Conformance

- In 2010, Ecma International started developing a standards test for Ecma 262 ECMAScript 5<sup>th</sup> Edition Specification.
- Test262 is an ECMAScript conformance test suite.

Scripting engine	Reference application(s)	Conformance			
		ES5	ES6	ES7	Newer(2016+)
Chakra	MS Edge 18	100%	96%	100%	48%
SpiderMonkey	Firefox 67	100%	98%	100%	83%
Chrome V8	Chrome 75, Opera 62	100%	98%	100%	98%
JavaScriptCore(Nitro)	Safari 12.1	99%	99%	100%	87%

# SPA

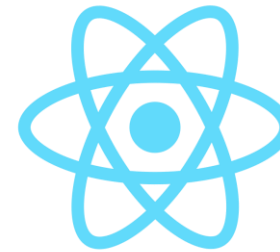
- Single Page Application
- Is a web application or web site that **interacts** with the user by **dynamically rewriting** the current page rather than loading entire new pages from a server.



# SPA (Cont.)

## ■ JavaScript frameworks

- [AngularJS](#)
- [Ember.js](#)
- [ExtJS](#)
- [Knockout.js](#)
- [Meteor.js](#)
- [React](#)
- [Vue.js](#)



# SPA (Cont.)

- Ajax
  - XMLHttpRequest, jQuery
- WebSockets
- Server-send events(SSE)
- Browser plugins
  - Silverlight, Flash, Java applets
- Data transport
  - XML, JSON

## SPA (Cont.)

- Web Browser 및 Browser Rendering Engine 내부에서 동작하는 Web Application.
- Web Browser에서 최초 접근한 URL을 기점으로 하여 다양한 화면으로의 이동을 제공하지만 기본적으로 최초의 HTML 안에서 User Interface가 완결됨.
- Page 내의 User Interface 변화에 따라 URL이 순차적으로 변화하며, Browser History를 통해 앞의 Page로 거슬러 올라갈 수 있음.
- Page에서 필요로 하는 Data는 Server로부터 API 등의 형태로 필요할 때마다 단편적으로 제공됨.

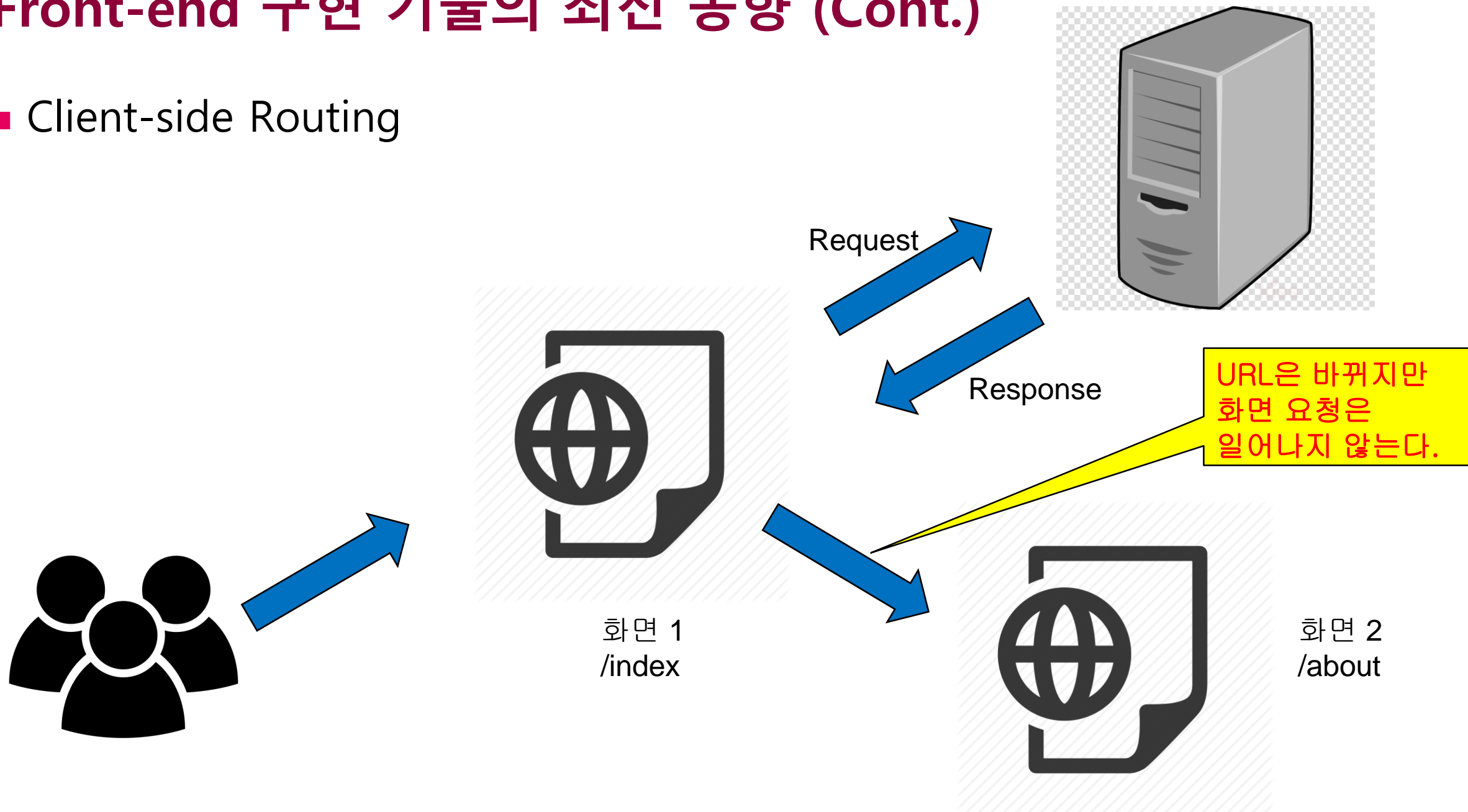
# Front-end 구현 기술의 최신 동향

## ■ Client-side Routing

- 일반적으로 Routing은 사용자의 Request가 들어온 URL에 대해 적절한 Response(HTML or JSON)을 돌려주기 위해 어떤 처리와 연결할 것인지를 결정하는 과정이다.
- 최근에는 Client 즉 Front-end에서도 Routing이라는 용어가 사용된다.
- SPA는 하나 이상의 화면 상태로 구성된다.
- 그러나 일반적인 Web Application과 달리 URL 단위로 Server에서 HTML Response를 받아오지 않는다.

# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ Client-side Routing



# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ Client-side Routing

- 이 경우 Browser의 이전 Button이나 Bookmark를 사용할 수 없고
- 중간 상태를 공유할 수 없기 때문에 User가 곤란해 진다.
- 따라서, Client 쪽에서도 URL마다 화면을 만들어두거나 HTML5의 History API로 이전 Button을 사용할 수 있도록 해야 한다.
- 이런 내용들이 Server-side Routing과 구분하기 위해서 Client-side Routing이라고 한다.

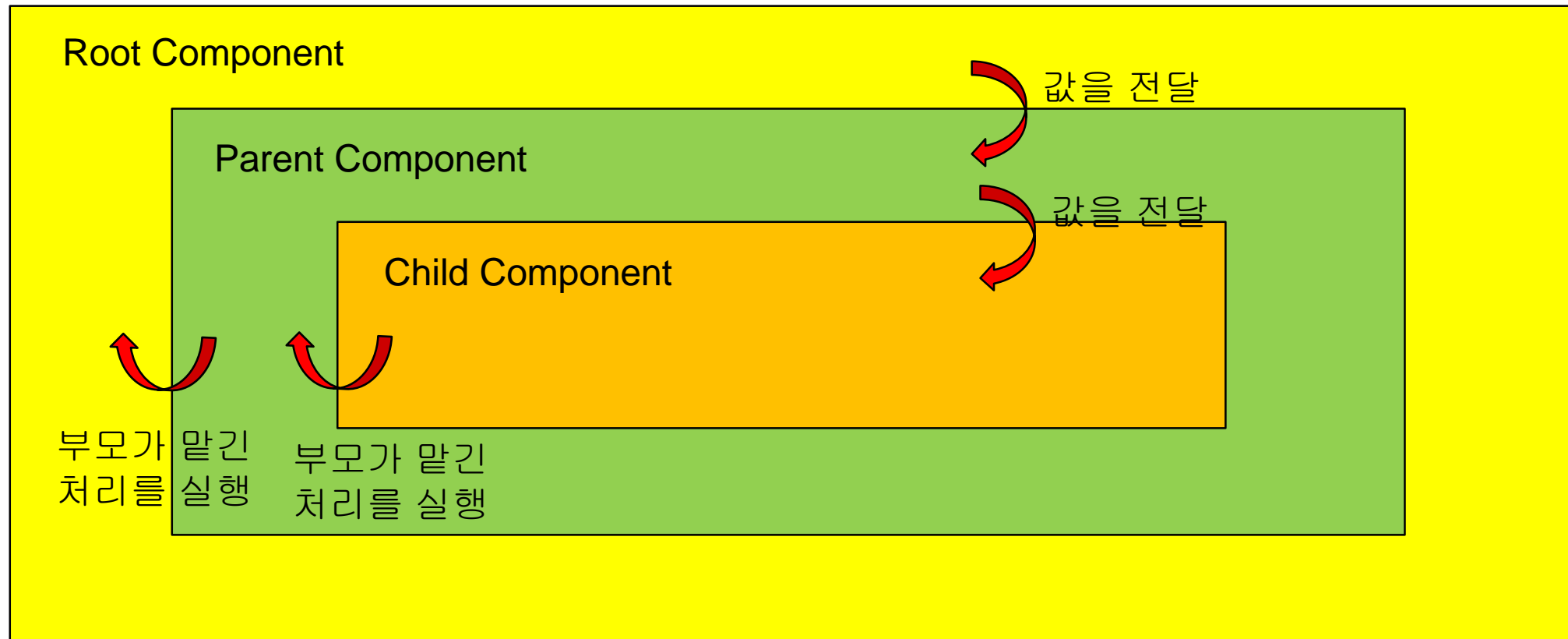
# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ Component-oriented

- Component와 Component간의 상호 작용 형태로 프로그램을 작성
- 만일, Radio Button이 Check되면 Send Button이 Click할 수 있는 상태가 되는 동작은 두 Component 사이의 상호작용이 있어야 가능하다.
- Component가 주체가 되어 Component 사이의 관계로 Application 전체를 구성하는 것
- Component-Oriented에서는 Application의 기본 정보를 포함하는 page를 통째로 하나의 Component(Root Component)로 정의한다.
- 이 Root Component 안에 다른 Component를 포함시키는 형태로 Component간의 부모 관계 및 형제 관계를 형성하고 Component 사이의 상호작용을 정의하여 Application을 제작한다.

# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ Component-oriented





# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ Component-oriented

- 구현시 Component 각각이 단독으로 필요한 기능을 수행할 수 있도록 Style과 Script를 포함하는 독립된 존재 즉 Capsulation이 필요하다.
- Vue.js에서는 Component에 필요한 CSS, Template, JavaScript를 File 하나에 기술한다.
- CSS는 Component 내부에만 적용되도록 범위가 제한된다.
- 이렇게 구현된 이유는 Framework 설계에 Component-Oriented가 반영되어 있기 때문이다.

# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ SSR과 Pre-rendering

- SSR(Server-Side Rendering)
- 일반적인 Web에서는 Browser는 Server에서 받아오는 HTML은 이미 Rendering이 끝난 HTML이다.
- 이와 달리 SPA에서는 비어 있는 HTML을 응답으로 받는다.

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

- 위의 code에서는 #app 안에 SPA의 Contents가 삽입되어 HTML이 동적으로 생성된다.
- 그래서 Client-side에서 Rendering이 이루어진다고 표현하는 것이다.
- 하지만, 현실적으로 Client-side에서 Rendering 할 때 문제점은 없는가?
- Page 표시 속도와 Crawler가 대응하지 못한다는 문제가 발생한다.

# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ SSR과 Pre-rendering

### ● Page 속도를 생각해보자.

- 빈 HTML이 먼저 표시되고 그 후에 Contents가 Rendering되기 때문에 전체 HTML이 바로 표시되는 경우에 비해 속도가 느려진다.
- 만일, Contents가 API를 경유하여 전달된다면, 더더욱 속도가 느려 질 것이다.

### ● 두번째, Crawler에서의 2가지 문제다.

- SEO(Search Engine Optimization) → Page가 검색 결과에 잘 포함될 것인가?
- SNS에 대한 문제 → 정상적으로 Web page가 공유될 것인가?
- wget이나 curl같은 명령행 도구로 URL에 접근하는 것과 마찬가지로 JavaScript가 실행되기 전에 page가 수집되는 문제 발생
- 이유는 Server로부터 들어오는 HTML은 Contents가 들어갈 자리가 비어있는 div 요소가 있기 때문이다.

### ● SPA의 약점은 기계가독성이 낮다는 점.

### ● 그래서 Crawler의 유형과 상관없이 정상적으로 Contents를 제공하려면 일반적인 Web site와 동등한 정보를 HTTP Response에 넣어서 보낼 수 있는 방법이 필요.

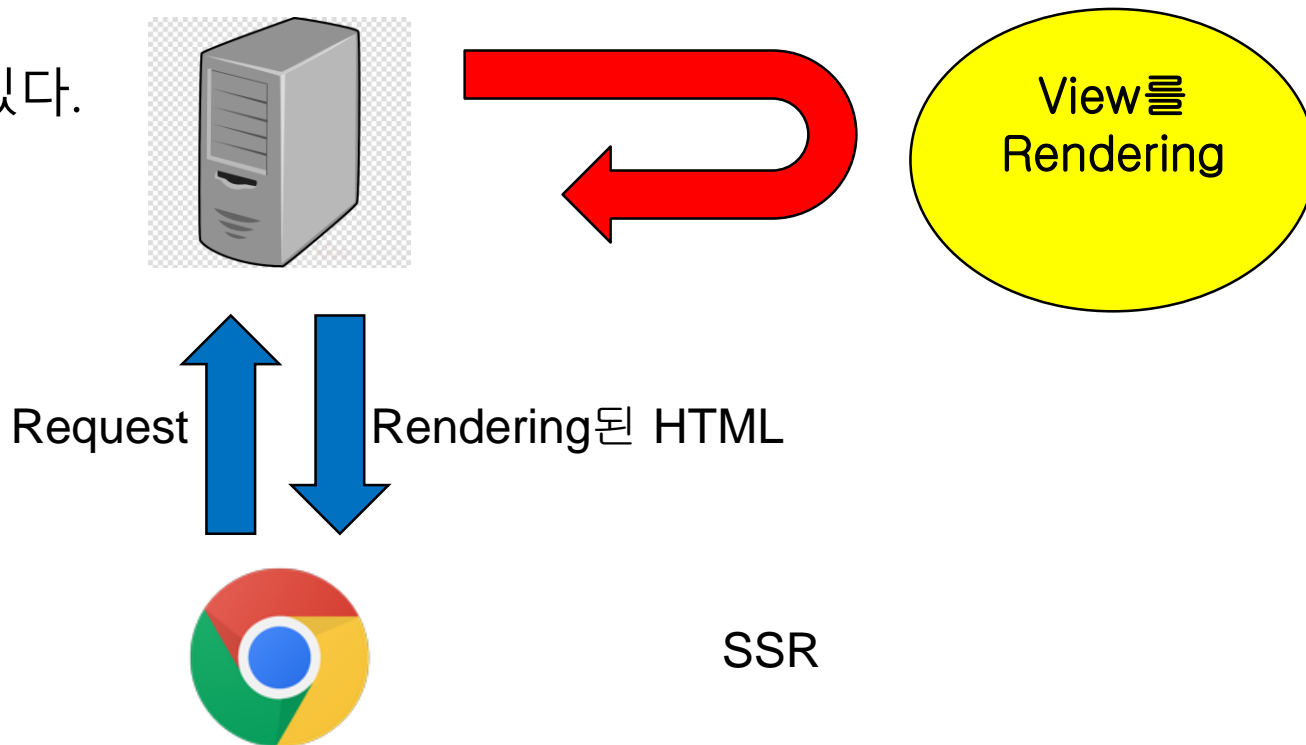
# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ SSR과 Pre-rendering

### ● 해결점1 – SSR(Server-side Rendering)

- Server에서 Client로 전달되는 Response 결과에 이미 Contents까지 Rendering을 마친 상태를 전달한다.
- 이를 구현하려면 어떤 방법이든지 Server-side에서 SPA에서 만들 Rendering 결과를 완성해야 한다.
- Vue.js에서는 vue-server-renderer가 있다.

### ● 해결점2 – Pre-rendering



# Front-end 구현 기술의 최신 동향 (Cont.)

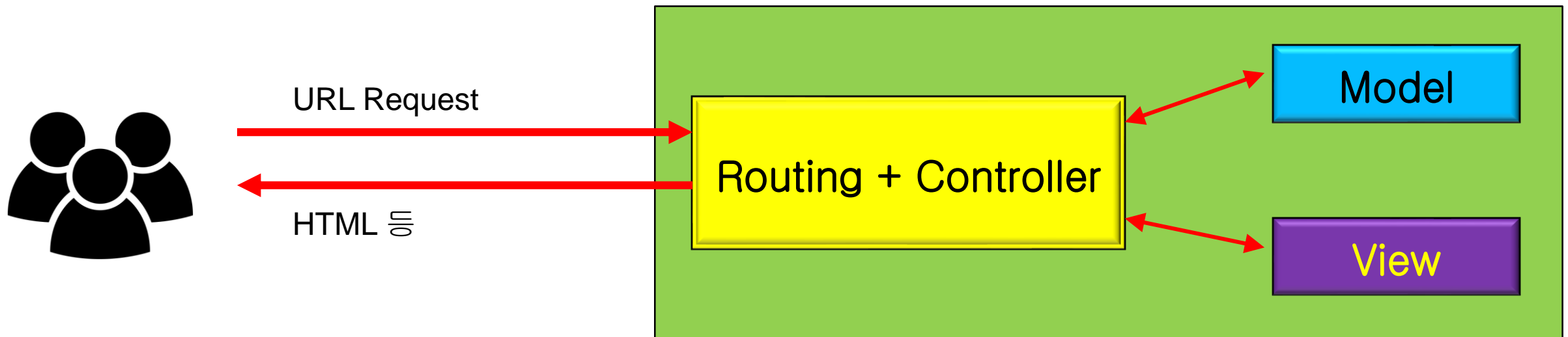
## ■ Virtual DOM

- JavaScript가 HTML을 Rendering 하는 방법 중 하나.
- 실제 DOM을 수정하는 과정은 JavaScript에서 Memory상의 객체를 수정하는 것보다 시간이 많이 소요된다.
- 따라서 일단 Memory 상에 있는 DOM 구조를 Update한 다음, 가상 DOM의 현재 상태와 이전 상태의 차이를 구해서 그 차이만을 실제 DOM에 반영하여 효율적으로 DOM을 수정하는 것이다.
- 가상 DOM을 도입하면 성능 향상 뿐만 아니라, 어떤 Data를 인자로 넘기면 HTML 구조 전체를 반환하는 함수를 통해 나머지는 가상 DOM이 어디를 수정할지 관리하고 직접 수정까지 가능하다.

# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ Web Application의 MVC

- Model-View-Controller
- Model은 Data와 Data에 접근하기 위한 기능
- View는 Contents를 어떻게 외부에 출력할지를 정의하는 부분, 사용자에게 전달되는 형태는 HTML일 수도 있고, JSON일 수도 있으며, 다른 Format일 수도 있다.
- Controller는 특정 URL을 요청받았을 때(Routing), 필요에 따라 Controller가 Model과 정보를 주고받은 다음 적절한 View를 골라 사용자에게 전달한다.



# Front-end 구현 기술의 최신 동향 (Cont.)

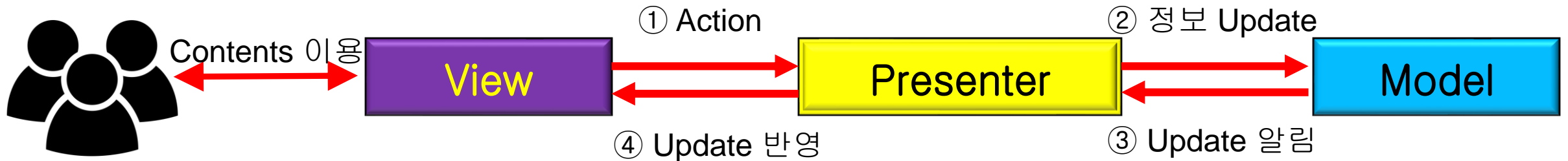
## ■ Front-end Application의 MV

- Front-end Application에서 MV(Model-View)는 MVC Application의 View 부분에 해당한다.
- Web Application은 본래 Contents가 HTML에 포함된 상태로 User에게 Response 해서 전달되기 때문에 View안에 MV와 같은 구성을 할 필요가 없었다.
- 그러나, SPA에서는 View안에도 Ajax등을 통한 Data 접근 계층이 추가되거나 필요에 따라 URL을 교체하는 처리 등이 포함되기 때문에 MV 구조가 필요하게 된다.

# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ MVP Pattern

- Model-View-Presenter
- MV는 Model과 View이다.
- 여기에 P(Presenter)가 추가된 Pattern이다.
- Model과 View 사이에 Presenter가 위치하여 양자 간 입출력 Interface를 담당한다.
- User가 Model의 정보를 수정하거나 읽으려면 반드시 View를 거쳐서 Presenter가 제공하는 Interface를 통해야 한다.

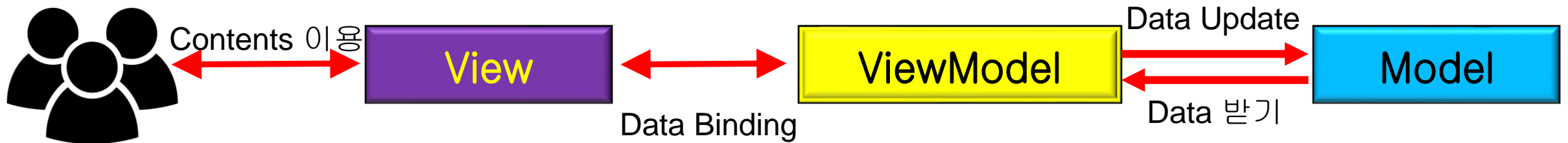




# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ MVVM Pattern

- Model-View-ViewModel
- ViewModel은 Model과 View 사이에 위치한다는 점에서 Presenter와 비슷하지만, 그 역할이 View와 Model간의 Interface 대신 양방향 Data Binding을 담당한다는 점이 다르다.
- 여기서 양방향이란 View를 통해 변경하려는 Model의 값이 ViewModel을 거쳐 변경 내용을 탐지하고 Model에 변경 내용이 반영된다는 뜻이다.



# Front-end 구현 기술의 최신 동향 (Cont.)

## ■ PWA

- Progressive Web Apps
- Google
- <https://developers.google.com/web/progressive-web-apps>
- 여기서 Progressive란 단계적으로 적용이 가능하며, 최신 Browser에는 최신 기술을 적용하고, 그렇지 못한 환경에서도 Contents를 열람할 수 있도록 하는 Progressive Enhancement(단계적 개선)의 사상을 말한다.
- Service Worker를 통한 Contents 제어
- Application Shell(App Shell)을 이용한 Application化
- Web Notification과 Web Push를 이용한 Push Notification

# Web Tech Terminology

## ■ Application State

- 전역으로 관리되는 Application의 상태
- Application의 어떠한 Component들도 이 상태에 접근할 수 있다.
- 그러나 쉽게 상태를 바꾸도록 허용되지는 않는다.
- 각 상태들은 Application의 Component 내부에서 특정 Event가 발생했을 때 이를 처리하는 변이들과 연결된다.

## ■ Bootstrap

- CSS에 대해 깊이 알지는 못해도 멋진 반응형 웹을 개발할 수 있도록 JavaScript 도구들과 관련 Style들을 제공

# Web Tech Terminology (Cont.)

## ■ CDN

- Content Delivery Network
- 사용자에게 Data 고가용성, 고속으로 제공하는 데 특화된 Server를 말한다.
- Framework를 개발한 사람들이나 회사는 주로 CDN을 통해 배포하는데, CDN의 URL만 명시하는 방법으로 설치를 유도할 수 있기 때문이다.

## ■ Component

- 자체 Data를 가지고 재사용될 수 있는 Application의 구성 요소
- 집을 짓는 데 벽돌 같은 역할을 한다.

## ■ CSS

- Cascading Style Sheet
- HTML 문서에 적용하면 해당 문서를 깔끔하고 아름답게 만들어주는 Style들의 모음

# Web Tech Terminology (Cont.)

## ■ 선언전인 View

- 일반 JavaScript 객체와 표현 간의 직접적인 Data Binding을 제공하는 View의 한 종류.

## ■ Directives

- Vue.js의 특별한 HTML Element로 다른 방식의 Data Binding을 지원

## ■ DOM

- Document Object Model
- HTML, XML, XHTML 같은 Markup 언어에서 Node들을 표현하는 방법
- 문서의 Node들은 DOM Tree 형태로 구성된다.
- DOM 작업을 한다는 것은 곧 HTML 요소들을 다룬다는 것

# Web Tech Terminology (Cont.)

## ■ npm

- JavaScript의 Package 관리자로 JavaScript Package 검색, 설치, 관리를 도와줌.

## ■ Markdown

- Web에서 글을 작성할 때 사람이 읽기 쉬운 문법을 사용해서 HTML 문법과 Style에 구애받지 않도록 해준다.
- .md 확장자

## ■ MVVM

- Architecture의 한 종류로 View와 Data Model 사이의 Data 흐름을 다루는 중계자로서 역할하는 ViewModel이 중심이된다.

## ■ One-way Data Binding

- Data Model의 변경이 View layer로 자동 전파되는 Data Binding 방식을 말한다.
- 반대 방향으로의 변경은 지원되지 않는다.

# Web Tech Terminology (Cont.)

## ■ Rapid Prototyping

- Web에서 기본적인 Action들만 포함하는 UI의 Mockup을 빠르게 작성하는 것

## ■ Reactivity

- Web에서 Data Model에 변화가 일어났을 때 즉각적으로 View에 반영되는 것

## ■ Two-way Data Binding

- Data Model의 변화가 자동적으로 View layer에 전파되고 View layer의 변화도 Data Model에 반영되는 것

## ■ UI

- 사용자가 Application과 상호작용할 수 있게 해주는 시각적인 Component들.

## ■ Vuex


- Application의 상태를 간단히 관리할 수 있게 해주는 Vue Application의 Architecture.

# Vue.js

■ <https://vuejs.org/>

Vue.js


Learn ▾ Ecosystem ▾ Team ▾ Resources ▾ Support Vue ▾ Translations ▾



## The Progressive JavaScript Framework

[▶ WHY VUE.JS?](#)[GET STARTED](#)[GITHUB](#)

Special Sponsor



Standard Library

Build APIs you need in minutes instead of days, for free.

### Approachable

Already know HTML, CSS and JavaScript? Read the guide and start building things in no time!

### Versatile

An incrementally adoptable ecosystem that scales between a library and a full-featured

### Performant

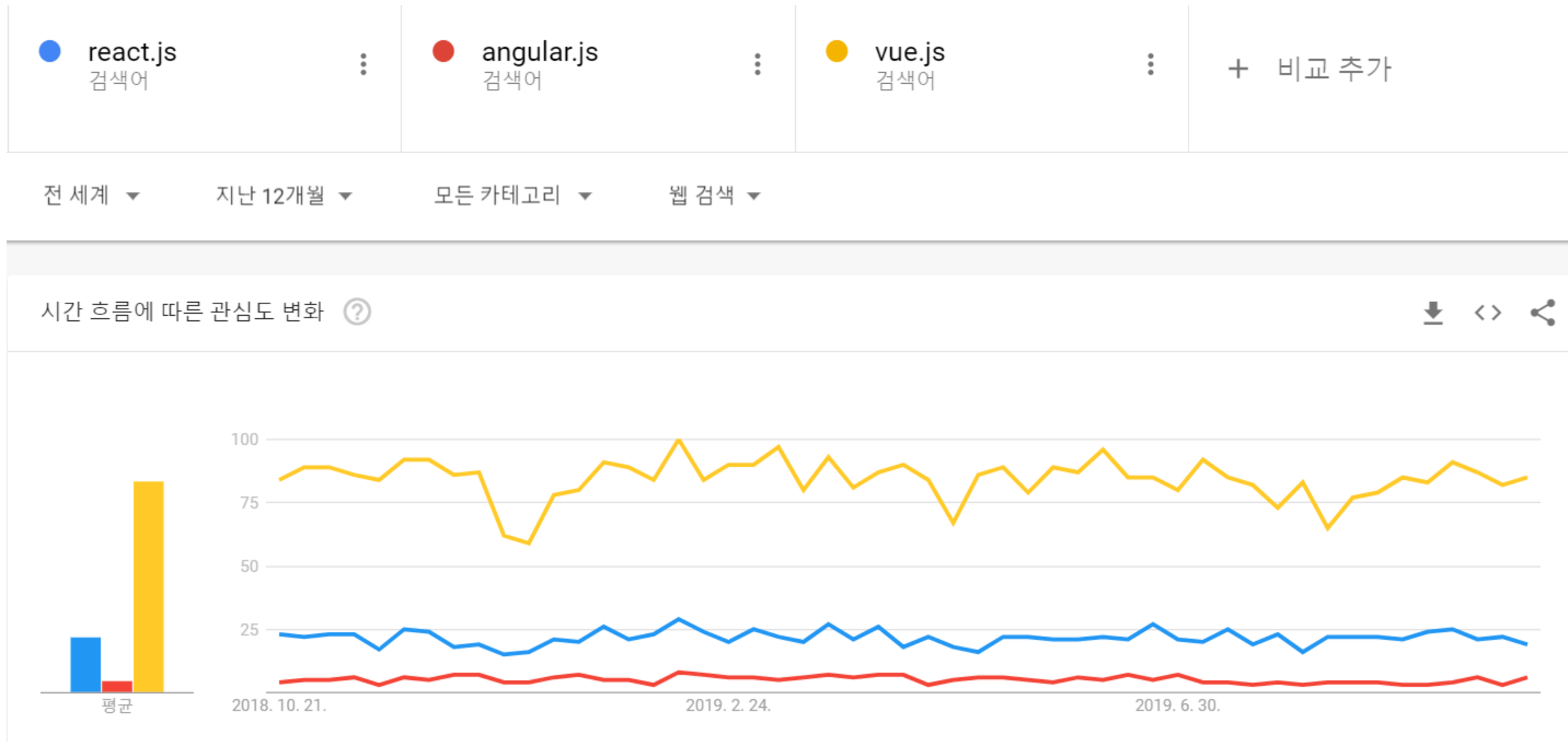
20KB min+gzip Runtime  
Blazing Fast Virtual DOM  
Minimal Optimization Efforts



# Vue.js (Cont.)

- Pronounced /vju:/, like view
- Is a progressive framework for building user interfaces.
- Google Creative Lab의 Evan You가 2013년 12월 개발
- UI를 빠르게 개발하기 위한 Framework(Library).
- 즉 Web page 화면을 개발하기 위한 Front-end Framework
- 규모가 큰 UI를 위한 Prototype을 빠르게 작성해야하고, 중복되는 HTML을 작성하는 일은 시간과 자원을 소비하는 일이기 때문에, 당시 존재하는 도구들(Angular, React.js, Backbone.js 등)에서는 빠른 UI Prototype을 위해 필요한 높은 유연함과 가벼움을 찾을 수 없었음.
- 그래서 재사용 가능한 Component를 지원하는 동시에 Reactive한 Data Binding을 쉽고 유연성 있게 제공해서 빠르게 Prototype을 만들 수 있는 Library를 만들게 됨.

# Vue.js (Cont.)



# Vue.js (Cont.)

bestofjs.org



All projects (1449 projects)

★ POPULAR

🔥 TRENDING



freeCodeCamp

306 k ★

The <https://www.freeCodeCamp.org> open source codebase and cu...

Learning resource

🔗 a day ago

👤 3,918



Vue.js

150 k ★

A progressive, incrementally-adoptable framework for building U...

UI Framework

Virtual DOM

Vue

🔗 a week ago

👤 282



React

138 k ★

A declarative, efficient, and flexible JavaScript library for building...

UI Framework

Virtual DOM

React

🔗 a day ago

👤 1,339



Bootstrap

136 k ★

The most popular HTML, CSS, and JavaScript framework for devel...

CSS Toolkit

Bootstrap

🔗 a day ago

👤 1,091

# Vue.js의 장점

## ■ Template

- Vue Instance나 Component에 있는 Data를 표시하는 방법으로 기존 HTML Template을 그대로 활용
- 따라서, 기존의 Web Application에 Vue.js Code를 부분적으로도 통합이 가능

## ■ 확장성

- jQuery처럼 script tag로 CDN을 추가할 수 있다.
- 점진적으로 커지는 Program에 맞게 필요한 Library를 선택해서 사용할 수 있다.