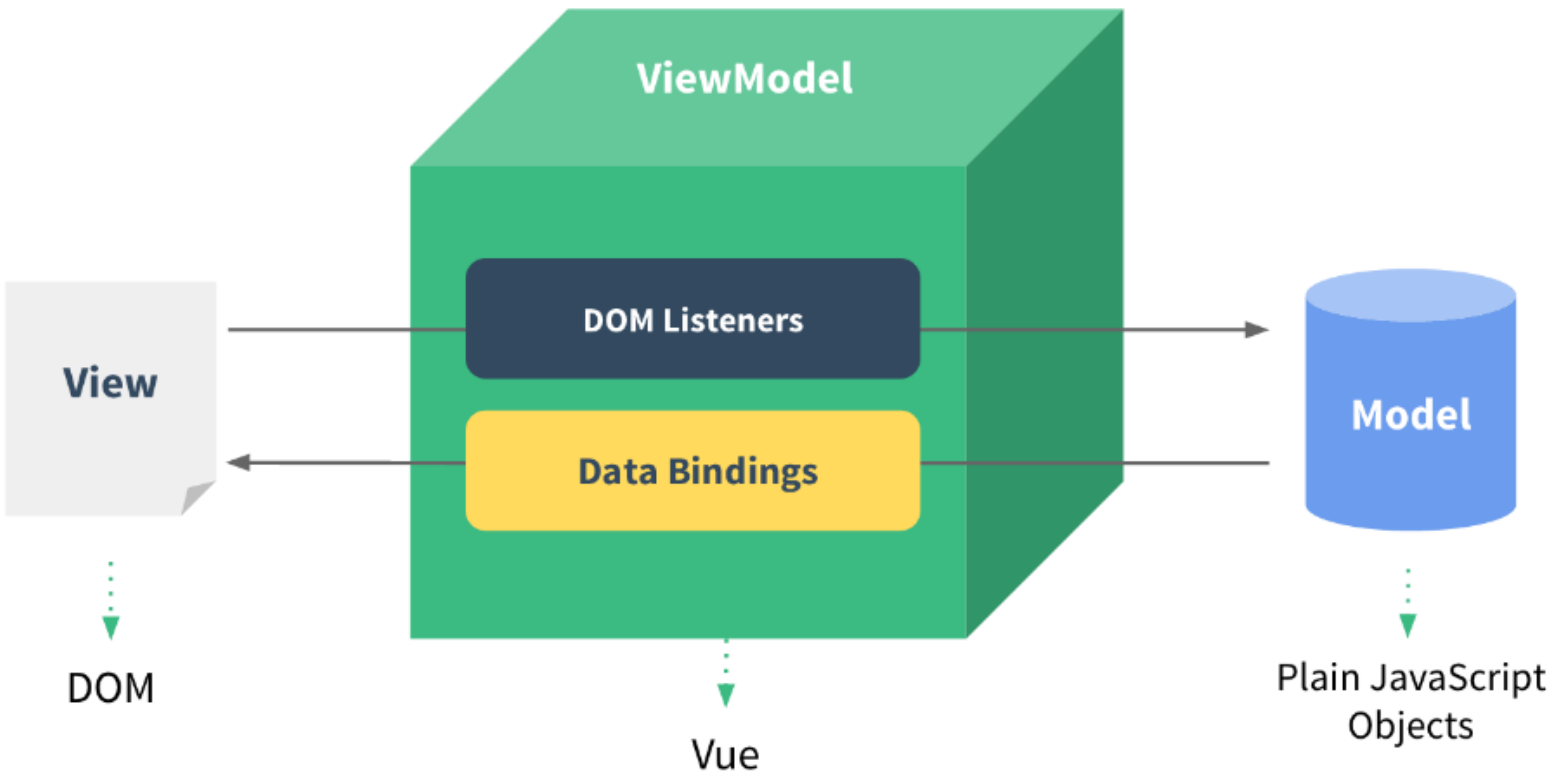
Vue 디자인 패턴

Status	Not started
Assign	Vue.js

Vue.js란 무엇인가?

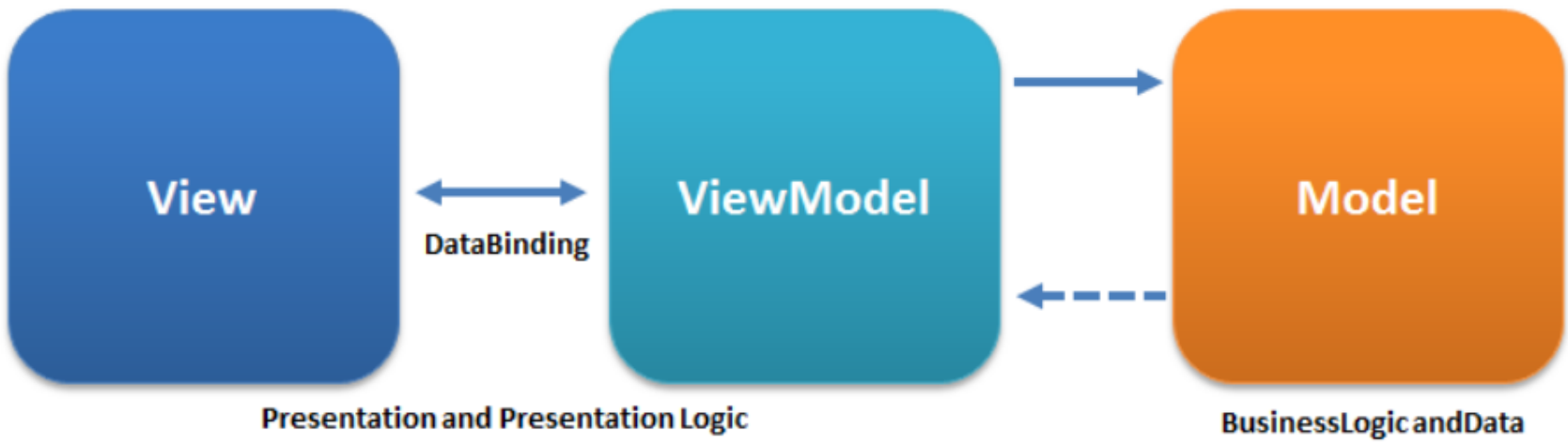
MVVM패턴의 **ViewModel**레이어에 해당하는 화면단 라이브러리



- 데이터 바인딩과 화면 단위를 컴포넌트 형태로 제공하며, 관련 API를 지원하는데에 궁극적인 목적이 있다.

MVVM 패턴

위키에 명시된 것처럼, Backend로직과 Client의 마크업 & 데이터 표현단을 분리하기 위한 구조로 전통적인 MVC패턴의 방식에서 기인하였다. 간단하게 생각해서 화면 앞단의 화면 동작 관련 로직과 뒷단의 DB데이터 처리 및 서버 로직을 분리하고, 뒷단에서 넘어온 데이터를 Model에 담아 View로 넘어가는 중간 지점이라고 보면 되겠다.



Vue Instance

Vue Instance 생성자

```
new Vue({
  template: "",
  el: "",
  data() {
    return {
    }
  },
  methods: {}
})
```

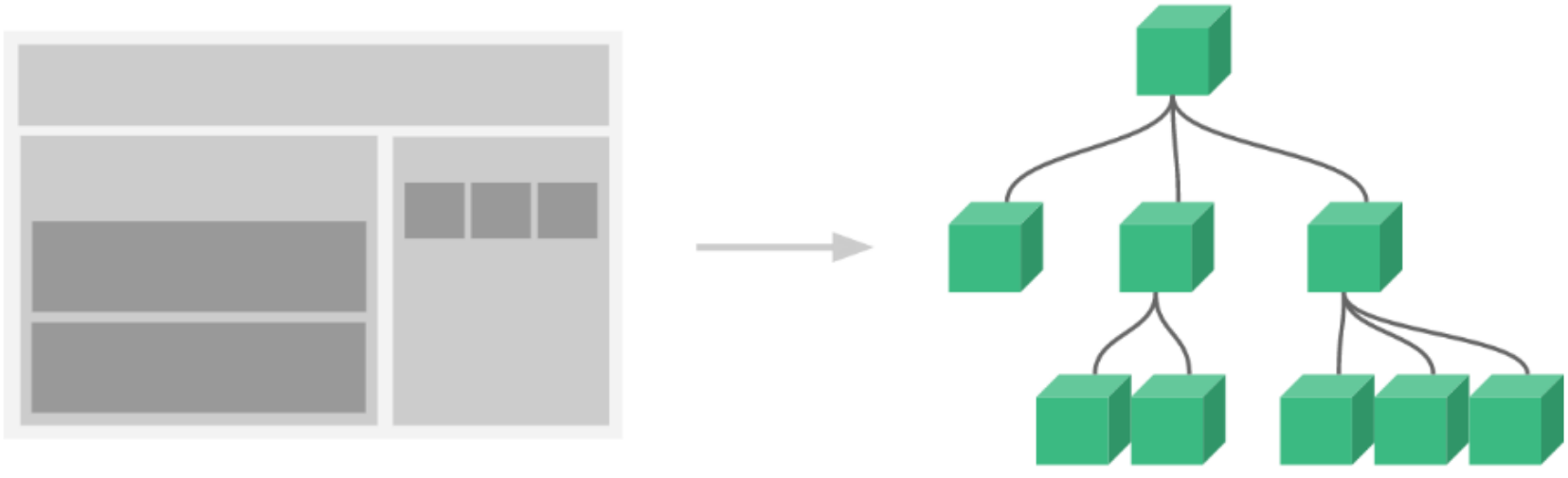
Vue Instance 라이프싸이클 초기화

인스턴스가 생성될 때 아래의 초기화 작업을 수행한다.

- 데이터 관찰
- 템플릿 컴파일
- DOM에 객체 연결
- 데이터 변경시 DOM 업데이트

Vue Component

화면의 영역을 일정한 단위로 쪼개어 재활용 가능한 형태로 관리하는 것이 컴포넌트



Global or Lacal Component

전역으로 등록하는 경우 : 애플리케이션의 전역에서 사용가능한 컴포넌트를 사용가능하게 만든다.

```
Vue.component('my-component', {
  template: '',
})
```

지역 컴포넌트로 등록하는 경우가 있다.

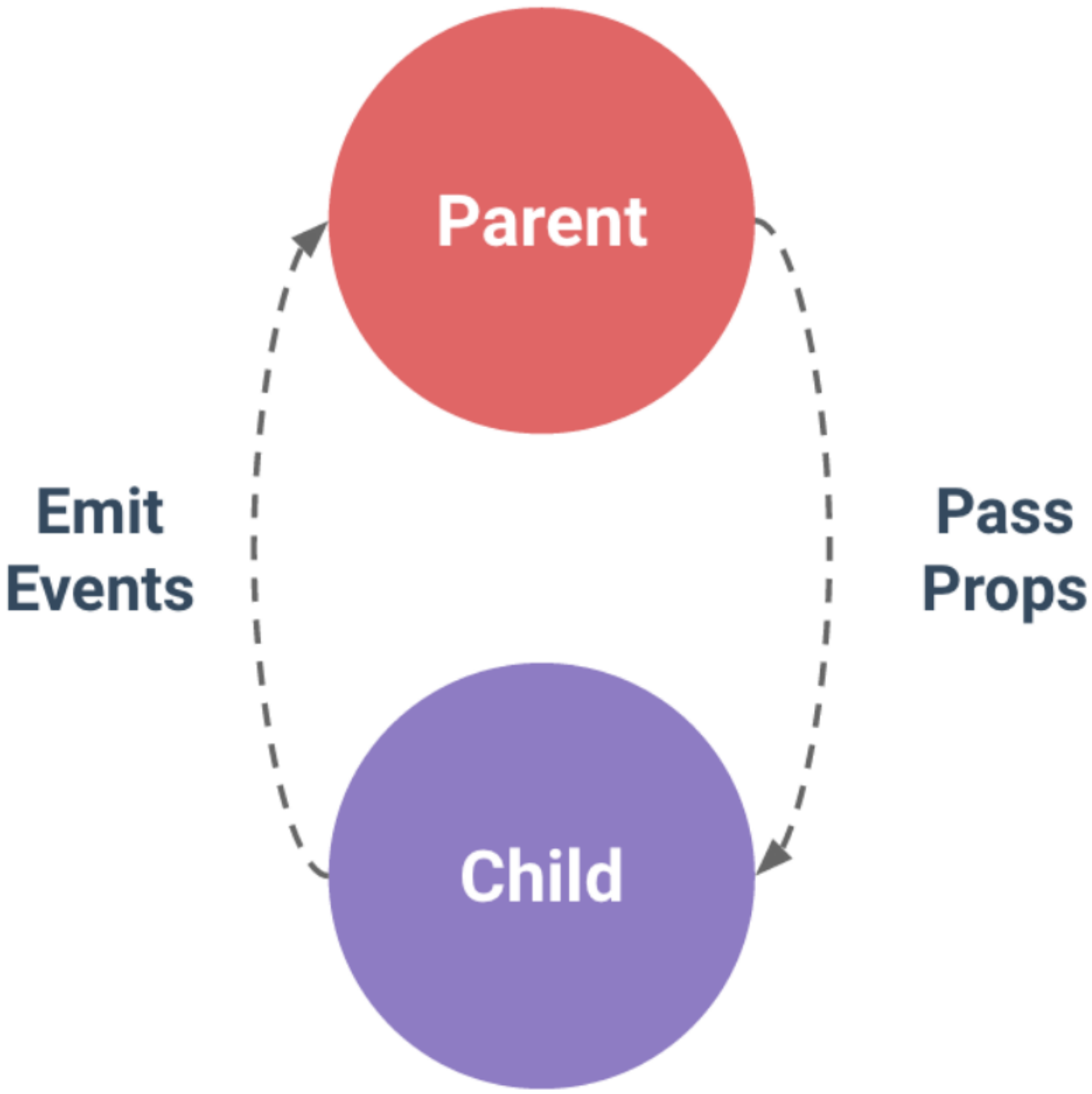
```
var cmp = {
  template: '',
}

new Vue({
```

```
  components: {
    cmp,
  }
})
```

부모와 자식 컴포넌트 관계

컴포넌트 관계도에서 상-하 관계에 있는 컴포넌트의 통신은

- 위에서 아래로 데이터(Props)를 내리고
- 아에서 위로는 이벤트를 올린다(event emit)



Props & $emit

```
const childComponent = {
  // 부모컴포넌트에서 자식컴포넌트로 데이터 내리기
  props: {
    parentData: String
  },
  template: '',
  data() {
    childData: '',
  }
  methods: {
    // 부모 컴포넌트에서 자식컴포넌트로 이벤트 올리기
    childToParent () {
      this.$emit('childToParent', childData)
    }
  }
```

```
}

new Vue({

  data() {
    return {
      parentData: '부모',
  },
  template: '<childComponent  :parentData="parentData" @childToParent="childToParent" />'
  methods : {
    childToParend(childData) {
      this.childData = childData
    }
  }
})
```

Event Bus

같은 레벨의 컴포넌트 간 통신

상위 - 하위 관계가 아닌 컴포넌트 간의 통신을 위해 Event Bus를 활용할 수 있다.

```
var eventBus = new Vue();

// 이벤트를 발생시킬 컴포넌트에서 $emit() 호출
eventBus.$emit("refresh", 10)

// 이벤트를 받을 컴포넌트에서 $on() 이벤트 수신
new Vue({
  created: function() {
    eventBus.$on("refresh", function(data) {
      console.log(data);
    })
  }
});
```

Vue Routers

뷰를 이용하여 싱글 페이지 애플리케이션을 제작할 때 유용한 라우팅 라이브러리, 공식 라이브러리로 지원되고 있다.

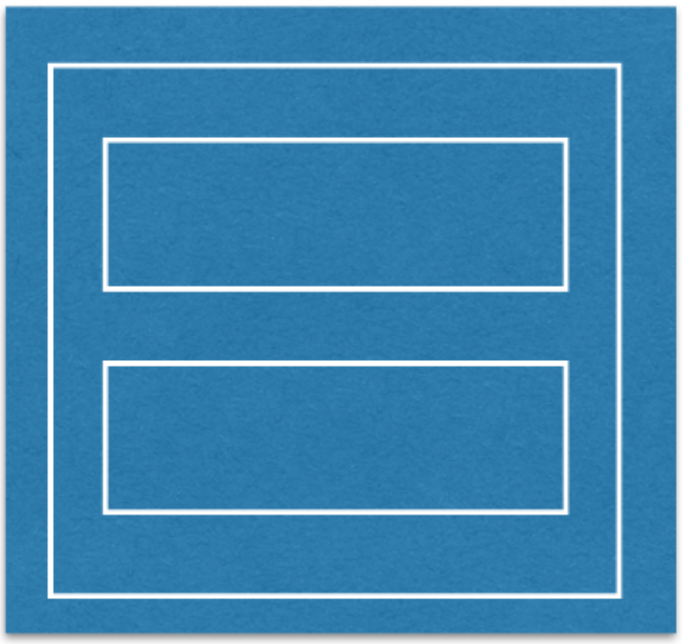
Nested Routes

라우터로 화면을 이동할 때 네스티드 라우터를 이용하여 지정된 하위 컴포넌트를 표시할 수 있다. 이 때 컴포넌트의 구조는 가장 큰 상위의 컴포넌트가 하위의 컴포넌트를 포함하는 `Parent - Child` 형태와 같다.

Nested Router vs Named Views

- 특정 URL에 지정된 1개의 컴포넌트가 여러 개의 하위 컴포넌트를 갖는 것을 Nested Router
- 특정 URL에 여러 개의 컴포넌트를 영역 별로 지정하여 렌더링 하는 것을 Named View

Nested Routes



Named View