※ Created 와 mounted 의 차이.

① Created

인스턴스가 작성된 후 동기적으로 호출됩니다. 이 단계에서 인스턴스는 데이터 처리, 계산된 속성, 메서드, 감시/이벤트 콜백 등과 같은 옵션처리 과정이 마운트가 시작되며 맨앞으로 $el 속성사용이 불가능.

- Virtual DOM 사용X
- created에서 이제 data와 event가 활성화 되며 접근할수있다.

② mounted

el 이 새로 생성된 vm.$el 로 대체된 인스턴스가 마운트 된 직후 호출. 루트인스턴스가 문서 내의 엘리먼트에 마운트 되어 있으면, mounted 가 호출 되면 vm.$el 도 문서 안에 있는다.

- Virtual DOM 사용 O
- 컴포넌트, 템플릿, 렌더링 된 돔에 접근 가능.

⇒ Created 는 데이터 초기화에 대한 목적 / Mounted는 DOM 조작에 대한 목적으로 주로 사용.

* Rendering?

html로 읽힌 값을 해석하여 표현 형식 (모델)로 출력하는 작업.

Renderer Engine (Skia) 의 실행.

* Binding

값이 서버상에서 들어간것 / name을 속성에 bind 시키는 과정. / 특정 객체에서 실행 되게 끔 프로시저는 역할

- 변수 / 함수 / 라인 / 절차 등의 명칭, 식별자가 그대상인 메모리 주소, 데이터형 또는 실제 값으로 해석되는것

1) 정적 바인딩 : 문서 트리그랩의 컴파일시 또는 링크 시에 확정되는 바인딩.

2) 동적 바인딩 : 프로그램이 실행되는 과정에서 바인딩되는것.

* DOM

Document Object Model (문서객체 모델)

HTML 문서의 객체 표현, 외부를 향하는 자바스크립트와 같은 HTML 와의 연결 지점.

* Template

렌더링된 DOM은 기본 Vue 인스턴스의 데이터에 선언적으로 바인딩 할 수 있는 HTML 기반 템플릿 문법을 사용.
Vue는 템플릿을 가상 DOM 렌더링 함수로 컴파일, 반응형 시스템과 결합된 Vue는 앱 상태가 변경될 때 최소한으로 DOM을 재계산하고 다시 적용할 수 있는 최소한의 컴포넌트를 지능적으로 파악. 렌더링 함수를 직접 작성이 가능.

* Hook

코드의 특정 지점을 가로채서 동작 방식에 영향을 주는 것.
대상 코드의 소스를 수정하지 않고 원하는 동작을 하도록 해야 하므로 기술적으로 어렵고, OS 선행 흐름을 파악 해야 하므로 위험하기도함.

1. CREATION

선행되는 hook 들이 라이프 사이클 중에서 가장 처음 실행.
컴포넌트가 DOM 에 추가되기 전이며, 서버 렌더링에서도 지원되는 HOOK
⇒ 클라이언트 단과 서버단 렌더링 모두에서 처리해야 할 필요가 없다면 이 단계에서 실행.

① beforeCreate()

모든 Hook 중 가장 먼저 실행되는 Hook. 아직 data와 events 가 세팅 x → 접근시 Error.

② Created()

data와 events 가 활성 → 접근가능 ⇒ 컴포넌트 초기에 세팅되어야 할 데이터 패치는 이 단계 사용.
템플릿과 가상돔은 마운트/렌더링 되지 x 상태.

2. MOUNTING : DOM 삽입단계.

초기 렌더링 직전에 컴포넌트가 직접 접근 할 수 있다.

① beforeMount()

템플릿과 관련 함수들이 컴파일 된 후에 첫 렌더링이 일어나기 직전에 실행.
사용 x 좋음! 서버사이드 렌더링시에는 호출되지 x

② mounted()

컴포넌트, 템플릿, 렌더링된 DOM 에 접근이 가능 / 모든 하위 컴포넌트가 마운트된 상태보장 x
서버 렌더링에서는 호출 x.

※ 부모와 자식 관계의 컴포넌트에서 mounted 는 자식 → 부모 mounted 순서로 실행.
(created 는 부모→자식 순서)

3. UPDATING : Diff / 재 렌더링 단계

컴포넌트에서 사용되는 반응형 속성들이 변경되거나 어떠한 이유로 재 렌더링이 발생되면 실행

디버깅/프로파일링 등을 위해 컴포넌트 재 렌더링 시점을 알고 싶을 때 사용

서버 렌더링에는 호출 X.

① beforeUpdated

컴포넌트의 데이터가 변화해 업데이트 사이클이 시작되며 실행.

(DOM 이 재 렌더링 되고 패치 되기 직전 실행)

재렌더링 전의 새 상태의 데이터를 얻을 수 있고 더 많은 변경이 가능.

② Updated

컴포넌트의 데이터가 변화해 재렌더링이 이어난후에 실행

DOM 이 업데이트 완료된 상태이므로 종속적인 연산을 할수 있다.

여기서 상태를 변경 하면 무한 루프에 빠질수도 있음!

4. DESTRUCTION : 해체 단계

① beforeDestroy

Vue 인스턴스가 제거 되기 전에 호출. 컴포넌트는 훨씬 모습과 모든 기능들을 그대로 가지고 있다.

이벤트 리스너 제거 혹은 reactive subscription 을 제거 하고자 한다면 여기서 하기

서버렌더링시 호출 X.

② destroyed

Vue 인스턴스가 제거된 후 호출

Vue 인스턴스의 모든 디렉티브가 바인딩 해제 되고 모든 이벤트 리스너가 제거, 모든 하위 Vue 인스턴스도 제거.

서버 렌더링시 호출 X.