

VUE

Vue és un framework JavaScript de codi obert pensat per a la creació de frontend i inteficies d'usuari.

La versió actual, i en la qual es basa aquest curs, és la versió 3. Als següents apartats es mostraran les bases que podeu ampliar amb la [documentació oficial](#).

Composition

Vue suporta dos "estils" d'API: options i composition. Al llarg d'aquest curs ens centrarem en la composition API. Els dos estils són semblants, però divergeixen en la forma de manejar el codi i les variables JS.

Un fitxer `.vue` tindrà tres parts `template`, `script` i `style` encara que prescindirem d'aquest darrer en la majoria de casos.

```
<template>
  <!-- Aquí definirem la plantilla amb html i variables vue -->
</template>

<script setup lang="ts">
  // Dins script hi haurà tota la lògica, definició de variables...
</script>

<style scoped>
  /* Si fes falta disposam d'un apartat styles per a definir estils del
  component. */
</style>
```

Template

La sintaxis del template serà la mateixa que html de tota la vida amb l'excepció de certs elements vue per a intercalar les nostres variables.

En el cas de voler pintar un valor dintre d'una etiqueta usarem els caràcters `{{ i }}` per a delimitar el codi js

```
<span>Nom de la persona: {{ persona.nom }}</span>
```

En el cas de propietat d'una etiqueta html precedirem de `:` l'etiqueta per a indicar que el valor és una propietat js.

```
<button :disabled="formulariOk">Envia</button>
```

Script

Dins l'apartat script es definiran els imports, propietats, mètodes... que usarem al template. Al usar el composition API totes les propietats definides seran automàticament accessibles al template, cosa que no passa amb el options API

```
import { Persona } from "@model/Persona";

const persona: Persona = new Persona("Joan");

const formulariOk: boolean = false;
```

El codi anterior és un exemple no funcional a falta de veure la reactivitat

Directives

Les directives són atributs especials amb el prefixe **v-** que doten a les etiquetes html d'interoperabilitat amb el codi JS.

Vue disposa d'un conjunt de directives que són suficients per a la majoria de casos, però es poden crear noves directives o usar les que venen en llibreries externes.

A continuació veurem les més representatives.

v-bind

Aquesta directiva, que ja hem vist en la forma abreujada, permet enllaçar una propietat d'una etiqueta html a una propietat JS.

```
<a v-bind:href="url"></a>

<!-- forma abreujada -->
<a :href="url"></a>
```

v-on

De forma semblant a l'anterior aquesta directiva ens permet enllaçar una etiqueta o element html a un event JS executant un mètode definit

```
<button v-on:click="metodeButtonClic">...</button>

<!-- forma abreujada -->
<button @click="metodeButtonClic">...</button>
```

v-if

Aquesta directiva, com el seu nom indica, ens permet renderitzar de forma opcional elements HTML. Es pot combinar, de forma opcional, amb **v-else-if** i **v-else**. En aquests casos els elements han d'anar precedits

un darrere l'altre.

```
<div v-if="persona.especialitat === 'T'">Tècnic</div>
<div v-else-if="persona.especialitat === 'A'">Administratiu</div>
<div v-else>Sense especialitat</div>
```

v-for

Com el seu nom indica ens permet recórrer un llistat per a renderitzar el contingut per a cada element.

```
<div v-for="observacio in observacions">
  {{ observacio.data }}: {{ observacio.text }}
</div>

<!-- Alternativament es pot obtenir també l'index -->
<div v-for="(titol, index) in titols">{{ index }}: {{titol}}</div>
```

NOTA: No està recomanant l'ús de `v-for` i `v-if` al mateix element HTML degut a que el comportament pot resultar no intuïtiu. EL següent codi no funcionaria.

```
<div v-for="nota of notes" v-if="nota > 5"></div>
```

<template>

Les directives es poden aplicar a elements HTML, això implica que generen una nova etiqueta. VUE incorpora l'etiqueta `<template>` que no renderitza cap element, simplement aplica la directiva en qüestió. Això ens permet solventar la problemàtica anterior.

```
<div v-for="nota of notes">
  <template v-if="nota > 5"> {{ nota }} </template>
</div>
```

Lifecycle Hooks

La part de l'script s'executa al carregar el component en qüestió. Això pot suposar un problema en certs casos que volem executar un codi en un moment concret, per exemple, al carregar o tancar el component.

Per aquest motiu VUE disposa de varis hooks que s'executaran en moments concrets del cicle de vida del component.

Per a usar un hook s'ha de definir aquest amb un mètode que s'executarà arribat el moment. El més usat és `onMounted` que s'executa una vegada el component està carregat.

```
onMounted(() => {  
  // Codi a executar  
});
```

A la següent imatge es pot veure el cicle complet.



