

Podrá descargar algunos elementos de este libro en la página web de Ediciones ENI: <http://www.ediciones-eni.com>.
 Escriba la referencia ENI del libro **EITVUEJS** en la zona de búsqueda y valide. Haga clic en el título y después en el botón de descarga.

Capítulo 1 Introducción

1.	Algunas nociones de lo que abarca este libro	17
2.	Requisitos previos	19
3.	Historia de Vue.js	20
3.1	¿Por qué Vue.js?	20
3.1.1	Sus orígenes	20
3.1.2	Sus puntos fuertes	21
3.2	¿Qué es el Virtual DOM?	24
3.2.1	Definición	24
3.2.2	API DOM para actualizar una página web	24
3.2.3	Límites del API DOM	26
3.2.4	Aparición del concepto de DOM virtual	27
3.2.5	Manipulación del DOM virtual	29
3.3	El DOM virtual con Vue.js	30
4.	Modelos de arquitectura	32
4.1	MVC (Modelo - Vista - Controlador)	32
4.1.1	MVC para todas las aplicaciones	32
4.1.2	MVC adaptado a las aplicaciones web	33
4.2	MVVM (Model - View - ViewModel)	34
5.	Comparativa con otros frameworks	36
5.1	Popularidad entre Vue, React y Angular	36
5.2	Ventajas e inconvenientes	38
5.2.1	Angular	38
5.2.2	React.js	39
5.2.3	Vue.js	39
5.2.4	Un framework para cada necesidad	40
5.2.5	Ir más lejos en la comparación con los otros frameworks	40
6.	Futuro de Vue.js	40

Capítulo 2

Nociones esenciales de JavaScript

1. Introducción	43
2. Bases algorítmicas	45
2.1 Variables y tipos de valores	45
2.1.1 Declaración de una variable	45
2.1.2 Tipos de valores	46
2.1.3 Contextos de ejecución global y local	48
2.1.4 Ámbito de una variable	50
2.1.5 Conversión de tipos de datos	53
2.1.6 Asignación por descomposición	55
2.2 Estructuras de control	55
2.2.1 Instrucciones condicionales	55
2.2.2 Bucles	58
3. Funciones	60
3.1 Definición y utilización de las funciones	60
3.2 Cierres (closures)	61
3.3 Función como argumento de otras funciones	63
3.4 Argumentos por defecto y argumentos del resto	64
3.5 Funciones de flecha	64
3.6 Funciones anónimas autoejecutables	65
4. Manipulación de arrays	67
4.1 Declarar, leer, modificar, eliminar elementos	67
4.2 Iterar en un array	69
4.3 Descomponer con el operador spread	69
4.4 Filtrar un array	70
4.5 Ordenar un array	70
5. Manipulación de objetos	70
5.1 Definir un objeto y su prototipo	70
5.2 Instanciar un objeto	72
5.3 Leer, añadir, modificar o eliminar una propiedad	75
5.4 Copiar o fusionar los objetos	76
5.5 Iterar sobre un objeto	77
5.6 Encadenar los métodos de objeto	77

6.	Utilización de la palabra clave <code>this</code>	78
6.1	Fuera de una función	78
6.2	En una función llamada clásica	78
6.3	En una función llamada con <code>call()</code> y <code>apply()</code>	79
6.4	Con <code>bind()</code> para asociar un objeto a <code>this</code>	79
6.5	En una función llamada como método del objeto	80
6.6	En un administrador de eventos	80
6.7	En una función de flecha	80
7.	Gestión de excepciones	81
7.1	La utilidad	81
7.2	La estructura <code>try...catch...finally</code>	82
7.3	El objeto <code>Error</code> y los errores personalizados	84
8.	Utilización de las promesas.	85
8.1	Objeto nativo <code>Promise</code>	85
8.2	Método <code>then()</code>	86
8.3	Administradores de <code>then()</code>	87
8.4	Método <code>catch()</code>	89
8.5	Encadenamiento de procesos asíncronos con promesas.	91
8.6	Creación de promesas mantenidas o fracasadas	92
8.7	Ejecución de tareas asíncronas en paralelo	93
8.8	Gestión de la pila de llamadas	93
8.9	Funciones asíncronas con <code>async</code>	94
9.	Utilización de los módulos JavaScript	96
9.1	La historia de los módulos JavaScript	96
9.1.1	Introducción.	96
9.1.2	Aplicación sin módulo.	97
9.1.3	Objetos como módulos y funciones auto-invocadas.	99
9.1.4	Módulo <code>CommonJS</code>	100
9.1.5	Módulos <code>AMD</code>	102
9.1.6	Cargador de módulos.	104
9.1.7	Empaquetador de módulos	106
9.2	Los módulos ES 6	108
9.2.1	Introducción.	108
9.2.2	Declarar y utilizar un módulo.	108
9.2.3	Evitar los conflictos de nomenclatura	109
9.2.4	Agregar varios módulos.	111

9.2.5	Importar dinámicamente.	111
-------	---------------------------------	-----

Capítulo 3

Entender los fundamentos de Vue.js

1.	Instalación	113
1.1	Una versión por entorno.	113
1.2	Mediante descarga manual.	114
1.3	Mediante la inclusión de un CDN (lo más sencillo).	114
1.4	Mediante npm o yarn para proyectos grandes	115
1.4.1	Descarga del paquete vue	115
1.4.2	Explicación de las diferentes builds	116
1.5	Mediante Vue-CLI	118
2.	Herramientas de desarrollo	119
2.1	VS Code y sus plugins.	119
2.1.1	Instalar y configurar VS Code.	119
2.1.2	Depurar con VS Code	121
2.2	Vue Devtools.	127
2.3	Vue Performance Devtool.	130
2.4	CodeSandbox.	132
2.5	Git	132
3.	Instancia Vue.js	133
3.1	Hello World	133
3.2	Enlace de datos entre View y ViewModel.	137
3.3	Ciclo de vida de una instancia	139
3.4	Utilización de la palabra clave this	142
4.	Virtual DOM y reactividad.	144
4.1	El sistema de unión de datos	144
4.2	El funcionamiento de la fila de espera de actualización asíncrona	147
4.3	La declaración de propiedades reactivas.	150
4.4	Las limitaciones en la detección de cambios	150
4.5	Las directivas para manipular el DOM	152
4.5.1	Un atributo HTML especial	152
4.5.2	Los argumentos y los argumentos dinámicos	155
4.5.3	Los modificadores.	155

5.	Unión de datos	156
5.1	Unión reactiva unidireccional	156
5.1.1	Mostrar datos reactivos con la interpolación de texto	156
5.1.2	Hacer un atributo reactivo con v-bind	156
5.2	Unión bidireccional con v-model	157
6.	Visualizar los datos en la View	159
6.1	La interpolación con los datos más complejos	159
6.2	La directiva v-html	160
6.3	La directiva v-once	161
6.4	La representación condicional	161
6.4.1	La directiva v-show	161
6.4.2	Las directivas v-if, v-else-if y v-else	162
6.5	La etiqueta <template>	164
7.	Tratamiento y formateado de los datos	166
7.1	Valores tratados con las propiedades calculadas	166
7.2	Valores tratados con los métodos	168
7.3	Propiedades calculadas vs métodos	169
7.4	Setters calculados	171
7.5	Valores procesados con observadores watch	174
7.6	Utilización de los filtros para formatear los datos	176
8.	Visualización de listas de datos	178
8.1	Utilizar la directiva v-for	178
8.2	Filtrar los elementos de una lista	181
8.2.1	No usar v-if con v-for	181
8.2.2	Usando una propiedad calculada	182
8.3	Ordenar los elementos de una lista	184
8.4	Cambiar un array o un objeto	185
9.	Captura de eventos desencadenados por el usuario	189
9.1	Directiva v-on	189
9.1.1	Actualizar un dato como consecuencia de un evento del DOM	189
9.1.2	Recuperar el evento en el método	191
9.2	Modificadores de eventos	192
9.3	Evento dinámico y sintaxis de objeto	194

10. Gestión de los estilos CSS	194
10.1 Scoped styles	194
10.2 Los estilos dinámicos	195
10.3 Los estilos dinámicos con un objeto	197
10.4 Los estilos dinámicos con un objeto sin CSS.	199

Capítulo 4

Utilizar los formularios

1. Los elementos de formulario	201
1.1 La utilización de librerías de terceros	201
1.2 Los campos text y textarea	202
1.3 Las casillas de selección y los botones de radio	205
1.4 Las listas desplegadas	208
2. Los campos particulares del formulario	210
2.1 Campo fecha	210
2.2 Campos de carga de archivo	213
2.3 Sliders	215
3. Los modificadores de la directiva v-model	217
3.1 Formatear una entrada de texto como número	217
3.2 Reaccionar al evento change en lugar de a input	218
3.3 Retirar los espacios	218
4. La validación del formulario	219
4.1 Las librerías de validación de formulario	219
4.2 Utilizar VeeValidate	219
4.2.1 Instalar	219
4.2.2 Utilizar las reglas de validación por defecto	221
4.2.3 Añadir sus propias reglas de validación	227
4.2.4 Gestionar el envío del formulario	229
4.3 Utilizar Vuelidate	232
4.3.1 Instalar	232
4.3.2 Utilizar las reglas de validación por defecto	233
4.3.3 Añadir sus propias reglas de validación	241
4.3.4 Gestionar el envío de formularios	246

Capítulo 5
Utilizar los componentes

- 1. ¿Qué es un componente? 249
 - 1.1 Definición 249
 - 1.2 Estructura de un componente 251
 - 1.3 Estructuración de una aplicación en componentes 253
- 2. Creación de un componente 254
 - 2.1 Manera global 254
 - 2.2 Manera local 254
- 3. Comunicación entre componentes. 255
 - 3.1 Padre – Hijo. 255
 - 3.1.1 Definición de los props en Vue. 255
 - 3.1.2 Tipos de datos 256
 - 3.1.3 Definición de las props en el ViewModel. 256
 - 3.1.4 Validación de las props 257
 - 3.1.5 Utilización de las props en el ViewModel 258
 - 3.2 Hijo - padre 260
 - 3.2.1 Emitir un evento desde un componente hijo. 260
 - 3.2.2 Escuchar un evento en el componente padre. 260
 - 3.3 Hijo - Hijo 262
 - 3.3.1 Límite de uso de las props y de los eventos personalizados 262
 - 3.3.2 Utilizar un bus de eventos 263
- 4. Para ir más lejos 264
 - 4.1 Las diferentes propiedades de instancia de un componente 264
 - 4.2 Utilizar la directiva v-model en un componente 265
 - 4.2.1 V-model para un componente que contiene un campo de texto. 265
 - 4.2.2 V-model para un componente que contiene una casilla de verificación o un botón de radio 265
 - 4.3 La herencia de los atributos del componente 267
 - 4.4 Los eventos nativos para vincular al componente 269
 - 4.5 La unión bidireccional de una prop con el modificador .sync 270
- 5. Varios tipos de componentes 271
 - 5.1 Los componentes dinámicos 271
 - 5.1.1 El elemento <component> y el atributo is. 271
 - 5.1.2 El elemento <keep-alive> 272

5.2	Los componentes monoarchivo	274
5.3	Los componentes básicos	276
5.4	Los componentes asíncronos para mejorar el desempeño	277
5.5	La utilización de la función render() para la representación gráfica	281
5.6	Los componentes funcionales	285

Capítulo 6

Compartir las funcionalidades

1.	Utilizar los slots para inyectar contenido.	289
1.1	Definición	289
1.2	Slots con nombre	293
1.3	Props de slot.	295
2.	Utilizar los componentes sin representación gráfica.	296
2.1	Definición	296
2.2	Creación de un componente de capturas de error.	297
3.	Utilizar los mixins	302
3.1	Definición	302
3.2	Estrategias de fusión de las opciones	304
4.	Utilizar los plugins.	307
4.1	Definición	307
4.2	Optimización del rendimiento.	310
4.2.1	Verificar el tamaño de los plugins y su velocidad de carga	310
4.2.2	Utilizar el tree-shaking para los plugins que lo permitan	312
5.	Las directivas personalizadas.	314
5.1	Definición	314
5.2	Argumentos y modificadores	317

Capítulo 7

Crear y desplegar una aplicación con Vue CLI

1.	Varios tipos de aplicaciones	321
1.1	Una SPA clásica	321
1.1.1	Definición	321
1.1.2	Restricciones de una SPA clásica	322
1.2	Una aplicación universal (SPA + SSR)	323
1.2.1	Definición	323
1.2.2	Restricciones de una aplicación universal	325
1.3	Un generador de sitios estáticos	327
1.3.1	Definición	327
1.3.2	Arquitectura JAMstack	329
1.3.3	Restricciones de los sitios estáticos	332
1.4	Una PWA	332
1.4.1	Definición	332
1.4.2	Diferentes tipos de cachés	333
1.4.3	Restricciones de una PWA	335
1.5	Un tipo de aplicación para cada necesidad	336
2.	Creación de un proyecto con Vue CLI	337
2.1	Presentación	337
2.2	Instalación	339
2.2.1	Requisitos previos	339
2.2.2	Prototipado rápido de una aplicación	339
2.2.3	Creación de una aplicación Vue.js completa	340
2.3	Entender la arborescencia del proyecto	347
2.3.1	Raíz del proyecto	347
2.3.2	Carpeta src	348
2.3.3	Modificación de la arborescencia para un proyecto de tamaño medio	350
2.3.4	Modificación de la arborescencia para un gran proyecto	355
3.	Funcionalidades de Vue CLI	357
3.1	Los plugins y los presets	357
3.1.1	Los plugins	357
3.1.2	Los presets	359
3.2	La recarga en caliente	360
3.3	La configuración de Webpack	361

3.3.1	Entender el funcionamiento	361
3.3.2	Acceder a los archivos de configuración de los diferentes modos	364
3.3.3	Modificar la configuración	365
3.3.4	Utilizar el API de encadenamiento para modificar de manera más precisa la configuración	368
3.4	Los modos y variables de entorno	369
3.4.1	Los modos para cada entorno	369
3.4.2	Las variables de entorno	370
3.5	La carpeta public	375
3.5.1	Interpolación en los archivos HTML	375
3.5.2	Archivos estáticos	375
3.6	La compatibilidad de los navegadores	376
3.6.1	La lista de los navegadores compatibles	376
3.6.2	El modo moderno	377
3.6.3	Los controles CORS en modo moderno para los módulos ES6	377
4.	Despliegue de su aplicación en producción	380
4.1	Procedimiento de despliegue	380
4.1.1	Compilar su aplicación con Webpack	380
4.1.2	Previsualizar en local	380
4.1.3	Desplegar su aplicación en un servidor	381
4.2	Despliegue simple con las plataformas PaaS	382
4.2.1	Desplegar en Netlify	382
4.2.2	Desplegar en Heroku	385
4.2.3	Crear una imagen Docker con Nginx	388

Capítulo 8

Consumir API REST y GraphQL

1.	Varios tipos de API (REST y GraphQL)	391
1.1	Definición e histórico	391
1.2	API REST	392
1.2.1	Definición	392
1.2.2	Restricciones de una API REST	395

1.3	API GraphQL	399
1.3.1	Definición	399
1.3.2	Descripción del funcionamiento	402
2.	Seguridad y modos de autenticación	403
2.1	Principios que hay que adoptar	403
2.2	Cookie de autenticación	405
2.2.1	Cookie de sesión	405
2.2.2	Ataques CSRF (Cross Site Request Forgery)	406
2.2.3	Ataques XSS (Cross Site Scripting)	407
2.3	Autenticación básica (Basic)	408
2.4	Autenticación con token al portador (Bearer)	409
2.5	Autenticación con token firmado (Bearer + JWT)	409
2.5.1	Firma	409
2.5.2	Principio clave privada - clave pública	410
2.5.3	Composición de un token JWT	410
2.6	Autenticación con una clave API	411
2.6.1	Autenticación de la aplicación por el proveedor de la API	411
2.6.2	Problemática de seguridad en la red del usuario	412
2.6.3	¿Dónde conservar las claves API?	412
2.7	Autenticación con OAuth 2.0	413
2.7.1	Un servidor de autenticación	413
2.7.2	El problema de conservar el token	415
3.	Crear una API rápidamente con Strapi	416
3.1	Crear el backend de su API	416
3.1.1	Presentación	416
3.1.2	Instalación	418
3.2	Crear el esquema de base de datos	420
3.2.1	Presentación del panel de control	420
3.2.2	Crear el esquema de base de datos	422
3.2.3	Gestionar roles y permisos	427
3.3	Probar la API REST	429
3.3.1	Probar las consultas GET, POST, PUT y DELETE	429
3.3.2	Utilizar los argumentos para filtrar una colección	434
3.4	Probar la API GraphQL	435
3.4.1	Utilizar Postman o el editor GraphQL de Strapi	435
3.4.2	Recuperar los datos	439
3.4.3	Crear un registro	441

3.4.4	Modificar un registro	443
3.4.5	Eliminar un registro	444
3.4.6	Utilizar los filtros	445
3.4.7	Funciones de agregación y cláusula <code>groupBy</code>	447
4.	Fetch y Axios para consumir API REST	452
4.1	Fetch	452
4.1.1	Definición y uso	452
4.1.2	Los objetos <code>Request</code> y <code>Response</code>	452
4.1.3	Gestionar los errores	457
4.1.4	Interrumpir una consulta	458
4.1.5	Limitar las llamadas demasiado frecuentes con <code>debounce</code> y <code>throttle</code>	459
4.1.6	Ejecutar consultas en paralelo	461
4.2	Crear un catálogo de productos	462
4.2.1	Mostrar la lista de productos con <code>fetch</code>	462
4.2.2	Gestionar los errores con un componente sin representación	467
4.2.3	Compartir la funcionalidad de llamada a la API	473
4.2.4	Implementar una paginación	476
4.2.5	Agregar un menú desplegable para filtrar la lista por categoría	481
4.2.6	Utilizar una barra de búsqueda	488
4.3	Axios	491
4.3.1	Definición y uso	491
4.3.2	Los objetos <code>config</code> y <code>response</code>	493
4.3.3	Gestionar los errores	497
4.3.4	Interrumpir una petición	499
4.3.5	Utilizar los interceptores	500
4.3.6	Centralizar la configuración <code>axios</code> en la aplicación	501
5.	Apollo para consumir las API GraphQL	503
5.1	Instalación	503
5.1.1	Instalación con <code>Vue CLI</code>	503
5.1.2	Instalación manual para consumir una API <code>Strapi</code>	503
5.1.3	Instalación del plugin <code>VS Code</code>	505
5.2	Recuperar los datos	506
5.2.1	Las peticiones inteligentes	506
5.2.2	Utilizar los argumentos en la consulta	511
5.2.3	Consulta condicional	513

5.2.4	Modificar los datos recibidos	513
5.2.5	Interceptar los errores	514
5.2.6	Las opciones adicionales	515
5.3	Ubicar el código de las consultas en los archivos .gql	516
5.3.1	Crear un archivo .gql	516
5.3.2	Importar un archivo .gql	517
5.3.3	Utilizar los fragmentos	518
5.4	Crear, modificar o eliminar un recurso	519
5.4.1	El método <code>this.\$apollo.mutate</code>	519
5.4.2	El componente <code>ApolloMutation</code>	522
5.5	Actualización del catálogo de productos con Apollo	523
5.5.1	Limpiar la antigua lógica de negocio de la API REST	523
5.5.2	Definir las consultas GraphQL	523
5.5.3	Definición de las consultas inteligentes	526

Capítulo 9

Utilizar Vue Router para la navegación

1.	Definición e instalación	531
1.1	Definición	531
1.2	Instalación	532
1.2.1	Descargar el plugin o utilizar un CDN	532
1.2.2	Utilizar npm	532
1.2.3	Utilizar Vue CLI	532
1.3	Declaración del router	533
2.	Utilización	534
2.1	Definición de una ruta	534
2.2	Resolución de una ruta	537
2.3	Modo histórico	539
2.4	Visualización de los componentes en las vistas del router	540
2.4.1	La vista por defecto	540
2.4.2	Las vistas con nombre	542
2.5	Navegar con ayuda de los enlaces <code><router-link></code>	543
2.5.1	Funcionamiento	543
2.5.2	La clase activa	544
2.5.3	Las props de <code><router-link></code>	545

2.5.4	Las rutas con nombre	546
2.6	Rutas dinámicas	546
2.6.1	Utilizar segmentos dinámicos	546
2.6.2	Recuperar los segmentos dinámicos con <code>\$route</code>	548
2.6.3	Utilizar los segmentos dinámicos para pasar valores a las props de un componente	549
2.6.4	Pasar entre dos caminos relacionados con la misma ruta	551
2.7	Las rutas anidadas	551
3.	Navegación programada	553
3.1	Navegación programada con <code>\$router</code>	553
3.1.1	<code>\$router.push()</code> y <code>\$router.replace()</code>	553
3.1.2	<code>\$router.forward()</code> , <code>\$router.back()</code> , <code>\$router.go()</code>	554
3.2	Los interceptores de navegación	554
3.2.1	Interceptores globales, por ruta o por componente	554
3.2.2	Flujo de resolución durante el cambio de una ruta a otra	556
3.2.3	Declaración de un interceptor	557
3.2.4	Ejemplo de uso de un interceptor global	558
3.2.5	Ejemplo de carga de datos antes de la navegación	559
3.3	Las redirecciones	560
4.	Para ir más lejos	561
4.1	Las transiciones	561
4.1.1	El componente <code><transition></code>	561
4.1.2	Ejemplo de una transición entrante y que se desvanece	563
4.2	Optimización del rendimiento	564
4.2.1	Utilización de lazy-loading con los componentes asíncronos	564
4.2.2	Utilizar la precarga con el método <code>webpackPrefetch</code>	566

Capítulo 10

Utilizar Vuex para la gestión de estados

1.	Compartir un estado global entre varios componentes	567
1.1	Problemática	567
1.1.1	El flujo de datos unidireccional	567
1.1.2	Utilizar props y eventos para compartir y acceder al estado global	568
1.1.3	Utilizar un bus de eventos para reaccionar a las acciones	569

- 1.2 Utilizar una store para centralizar los datos 570
 - 1.2.1 Delegar los estados a un objeto compartido 570
 - 1.2.2 Trazar las mutaciones 572
 - 1.2.3 Utilizar Vuex 574
- 2. Instalar y utilizar store Vuex 575
 - 2.1 Instalación 575
 - 2.1.1 Con un CDN 575
 - 2.1.2 Con npm o yarn 575
 - 2.1.3 Con Vue CLI 577
 - 2.2 Estado 578
 - 2.2.1 Definir y acceder a un dato del estado 578
 - 2.2.2 Utilizar mapState() para generar las propiedades calculadas . . . 579
 - 2.3 Getters 581
 - 2.3.1 Acceder a un dato del estado a través de un getter 581
 - 2.3.2 Utilizar mapGetters() para generar las propiedades calculadas 583
 - 2.4 Setters 584
 - 2.4.1 Modificar un dato del estado de la store 584
 - 2.4.2 Modificar varios datos con una mutación 586
 - 2.4.3 Modificar un objeto o un array 588
 - 2.4.4 Utilizar un archivo de constantes para listar los tipos de mutaciones 589
 - 2.4.5 Utilizar mapMutations() para generar las propiedades calculadas 590
 - 2.4.6 Las mutaciones con la directiva v-model 591
 - 2.5 Acciones 593
 - 2.5.1 Utilizar las acciones para los tratamientos asíncronos 593
 - 2.5.2 Utilizar mapActions para generar los métodos 595
 - 2.5.3 Encadenar las acciones asíncronas 595
- 3. Utilizar los módulos para organizar su store 597
 - 3.1 Separar el store en varios archivos 597
 - 3.2 Utilizar los módulos Vuex 599
- Conclusión 601
- Índice 603