

# Programación Web Avanzada

## PEC 0: Tooling

### Presentación

En esta primera PEC se describe el entorno de desarrollo propuesto para la elaboración del resto de las PECs de la asignatura. Esta actividad no se evalúa, pero se recomienda encarecidamente realizar para tener el entorno preparado para la realización del resto de actividades de la asignatura.

### Conocimientos, Habilidades y Competencias

En esta PEC se desarrollan los siguientes conocimientos (K), habilidades (S) y Competencias (C) del Grado en Multimedia 2023:

- **C2.** Diseñar, desarrollar y gestionar proyectos multimedia adaptándose a la evolución de las tecnologías, los lenguajes y las necesidades del mercado y de la sociedad.
- **C5.** Aplicar las tecnologías digitales en el ámbito académico y profesional
- **S3.** Utilizar de manera adecuada los lenguajes de programación, las herramientas de desarrollo y las tecnologías disponibles para el análisis, el diseño y la implementación de aplicaciones multimedia.
- **S7.** Analizar y sintetizar información para evaluar soluciones tecnológicas y elaborar propuestas de proyectos multimedia teniendo en cuenta los recursos, las alternativas disponibles y las condiciones de mercado.

### Objetivos

Los Objetivos concretos de esta PEC son:

- Conocer un entorno de desarrollo en JavaScript concreto.

## Consultas

En caso de que tengáis que consultar algo mediante el foro o por correo electrónico, copiad vuestro código a una plataforma online y enviad el link para evitar problemas con el correo de la UOC (ya que elimina los ficheros .js para evitar inyectar código malicioso así como la indentación del código). Os recomendamos utilizar cualquiera de estas dos opciones:

- [Codepen](#) (para sencillos *snippets* de código)
- [CodeSandBox](#) (para ejercicios más complejos)

**Nota:** En caso de publicar algún código en el foro, deberán ser consultas genéricas y no directamente soluciones a los ejercicios. Hacer accesible soluciones de ejercicios a otros compañeros, aunque pueda no tener una intención directa, se considerará copia y se penalizará académicamente a nivel de asignatura.

## Propiedad intelectual y plagio

La Normativa académica de la UOC dispone que el proceso de evaluación se cimenta en el trabajo personal del estudiante y presupone la autenticidad de la autoría y la originalidad de los ejercicios realizados.

En esta actividad no está permitido el uso de herramientas de inteligencia artificial. En el plan docente y en la [web sobre integridad académica y plagio de la UOC](#) encontraréis información sobre qué se considera conducta irregular en la evaluación y las consecuencias que puede tener. Recuerda que siempre que reutilices contenido de terceros debes citar la fuente, puedes leer también esta [guía para saber cómo citar correctamente](#).

El estudiante será calificado con un suspenso (D/0) si se detecta falta de originalidad en la autoría de alguna prueba de evaluación continua (PEC) o final (PEF), sea porque haya utilizado material o dispositivos no autorizados, sea porque ha copiado textualmente de internet, o ha copiado apuntes, de PEC, de materiales, manuales o artículos (sin la cita correspondiente) o de otro estudiante, o por cualquier otra conducta irregular.

## Introducción

*Tooling* es el término anglosajón usado para describir un **conjunto de herramientas** de forma genérica. Cuando hablamos del tooling dentro del ecosistema de **JavaScript** nos referimos a todas aquellas utilidades auxiliares que nos servirán de apoyo durante el desarrollo de nuestros proyectos.

Gran parte de las acciones que llevamos a cabo durante la fase de desarrollo son mecánicas y repetitivas, por lo que tener un tooling que automatice estos procesos nos permitirá minimizar el tiempo que dedicamos dichas acciones repetitivas y centrarnos de manera más eficiente y efectiva en el desarrollo de nuestras aplicaciones software.

La historia moderna de la automatización de los entornos de desarrollo front-end, ha pasado por múltiples fases, pero cabe destacar los siguientes protagonistas ordenados cronológicamente:

- [Grunt](#)
- [Gulp](#)
- [Webpack](#)

Las diferencias entre ellos se deben a concepciones muy diferentes de cómo abordar la automatización de procesos, y deben entenderse no como competidores, sino como evoluciones. Es cierto que todos ellos comparten el mismo ecosistema, pero ello se debe a la adopción de cada uno de ellos de forma cronológica. Actualmente, es ampliamente aceptado que todo nuevo proyecto debe considerar **Webpack** como la opción por defecto.



*Aunque no lo veremos en la asignatura en profundidad, si te interesa conocer más sobre Webpack puedes visitar [su web](#) y [su documentación](#). Es una herramienta que muy posiblemente acabarás utilizando en cualquier desarrollo Web, por lo que conocerlo con un poco más de detalle te ayudará a estar más preparado a los retos que puedas encontrar en el futuro.*

Como en la asignatura trabajaremos con el framework **Vue.js**, en nuestro caso vamos a utilizar **Vite**, una herramienta de consola que permite generar proyectos de **Vue.js** utilizando **ES Modules** de forma nativa y que proporciona una serie de herramientas que requieren mucha menos configuración que este, pero que ofrecen la misma flexibilidad y potencia. A su vez, proporciona una estructura de carpetas estándar para cualquier proyecto en Vue.js.

A continuación vamos a realizar una serie de pasos de forma guiada para implementar el entorno de una **aplicación Vue.js**, que será nuestro entorno de trabajo para el resto del curso, tanto para los ejercicios teóricos como prácticos de las PECs de la asignatura.

Sin embargo, antes de ponernos a automatizar procesos, analizaremos cada uno de los desafíos que se nos presentan, las tecnologías que entran en juego y los problemas que quedan resueltos por cada una de ellas.

## Node.js y npm

Históricamente, JavaScript había sido un lenguaje ejecutado en el navegador. Los navegadores son aplicaciones de escritorio que albergan en su interior un motor (*engine*) que compila y ejecuta Javascript. Sin embargo, fue en 2009 cuando [Ryan Dahl](#) cambió este paradigma proponiendo Node.js como entorno de ejecución independiente del navegador.



Puedes ver la presentación donde Ryan Dahl presenta Node.js y explica su propuesta de cambio de paradigma en [este video](#) de Youtube.

[Node.js](#) es un **entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome**, dicho de modo muy simple, una emulación de lo que sucede en el navegador sin el renderizado visual (no existe el objeto *window*).

Este hecho fue disruptivo, ya que permitió que JavaScript, uno de los lenguajes de programación más populares y con una curva de entrada más baja, pasara a ser, simultáneamente, un lenguaje de cliente y de servidor, lo que permitía que muchos ingenieros de *front-end* pudieran dar el salto a *back-end* sin tener que cambiar de lenguaje.

**Node.js** permitió el desarrollo de muchas aplicaciones no destinadas directamente a los navegadores y facilitó la aparición de un ecosistema de utilidades a las que se les llama paquetes (*packages*).

Al instalar **node.js** en nuestro ordenador se instala también un gestor de paquetes: el **NPM** (*Node Package Manager*, o gestor de paquetes de node). Esta utilidad nos permite, entre otras cosas:

- Crear nuestros propios paquetes (nuestras aplicaciones).
- Instalar y usar paquetes desarrollados por terceros en nuestra aplicación.

Para poder tener una idea del tipo de aplicaciones a las que nos referimos os invitamos a visitar [npmjs.com](#), el repositorio de paquetes oficial de **Node.js**, una especie de supermercado de utilidades. Podemos, por ejemplo, instalar [Vue.js](#), mediante las instrucciones presentadas en [esta página](#).



En esta asignatura te daremos las indicaciones básicas para poder instalar y utilizar Node.js. No es el objetivo de la asignatura profundizar en Node.js, pero si te interesa conocer más sobre la herramienta, además de visitar [su web](#), te recomendamos estos libros:

- James Hibbard, Craig Buckler, Mark Brown, Nilson Jacques, James Kolce, Paul Orac, M. David Green, Florian Rappl. [Your First Week With Node.js](#), 2nd Edition. SitePoint. 2020
- Bethany Griggs. [Node Cookbook](#). Packt Publishing. 2020

## Instalación de node.js y NPM


Ahora que sabemos qué es y para qué sirven **Node.js** y **NPM**, podemos dirigirnos a su [página oficial](#) para descargarnos el binario más adecuado para nuestra plataforma de desarrollo.

Se nos presentan dos opciones: última versión o LTS (*Long Term Support*); esta última es la opción más segura y aconsejable en nuestro caso, ya que no tenemos necesidad de hacer uso de las últimas novedades ni versiones experimentales con poco soporte.

Para comprobar que la instalación ha sido correcta, debemos abrir un terminal y ejecutar los siguientes comandos:


```
node -v
npm -v
```

Obtendremos como respuesta de cada uno de estos comandos la versión de node y de npm respectivamente.

 El porcentaje de usuarios que utilizan un entorno Linux/Mac en el desarrollo de aplicaciones JavaScript supera de manera notable a los que usan entornos Windows. De aquí en adelante los comandos presentados serán todos en formato Linux/Mac. Los conceptos serán los mismos e independientes de la plataforma, pero los usuarios de Windows deberán investigar en el caso de que los comandos no sean directamente ejecutables en la consola.

La utilidad **npm** es lo que se conoce como una **interfaz de línea de comandos** (*Command Line Interface*, CLI) y es lo que nos permite importar código de terceros desde el [registro de npm](#). Esto implica que el uso de la herramienta debe hacerse mediante la consola de nuestro sistema operativo. A continuación podéis encontrar unos enlaces en los que podéis familiarizaros con estas aplicaciones que serán los entornos en los que trabajaremos esta práctica:

- [Mac](#)
- [Linux](#)
- Windows (dos opciones): [cmd](#) / [terminal](#)

 Los usuarios de [Visual Studio Code](#) (nuestro editor recomendado) pueden obviar este paso, ya que lleva incluida una herramienta de consola. Si quieres ahorrarte el paso anterior, no dudes en instalarte desde [aquí](#) este editor y abrir la consola de [este modo](#).

## Vue.js y Vite

Vue.js (conocido como Vue y pronunciado /vju:/, que suena como el verbo inglés “view”) es un framework JavaScript para la construcción de front-end, **aplicaciones de una sola página** (también llamadas *Single-Page Applications*, o **SPA**) y **aplicaciones web progresivas** (conocidas como *Progressive Web Apps*, o **PWA**). Fue creado por [Evan You](#) como una alternativa a otros frameworks como [React](#) o [Angular](#).



*¿En qué se diferencia una SPA de una PWA?*

*Una SPA es una aplicación Web que no sigue el método clásico de web (múltiples archivos HTML comunicados por links), sino que la página se va reescribiendo conforme el usuario navega por ella.*

*En cambio, podríamos llamar a las PWA como una versión avanzada de las SPA: de la misma manera que existen aplicaciones desarrolladas para diferentes sistemas operativos (Android, iOS, Windows, etc.), las aplicaciones web progresivas son multiplataforma, ya que se ejecutan en un navegador, pero disponen de características de las apps nativas como la ejecución sin conexión, en segundo plano o soporte para mostrar notificaciones.*

*Aunque no nos centraremos en ellas durante el desarrollo de esta asignatura, aquí tienes más información:*

- MDN contributors, [Progressive Web Apps \(PWAs\)](#). MDN Web Docs. 2021

En el contexto de esta asignatura, utilizaremos el framework Vue.js para llevar a la práctica los diferentes conceptos que queremos trabajar durante el curso. En esta PEC te damos las indicaciones para poder crear un esqueleto de un proyecto en Vue.js, que más adelante, en las siguientes PECs, desarrollaremos.



*En el ámbito Web existe mucha volatilidad entre las herramientas Web. Esto significa que el tiempo de vida de estas herramientas puede ser muy corto: acumulan una importancia notable durante un tiempo, y luego son sustituidas por otras. En otros casos, se convierten en estándares de facto. Por este motivo, un profesional del ámbito Web tiene que estar preparado para aprender y adoptar nuevas tecnologías con frecuencia.*

*Los frameworks Web React, Angular y Vue.js son quizás los más conocidos y se han establecido como las principales opciones en el desarrollo Web, y por eso merece la pena estudiarlos. En esta asignatura nos centraremos solamente en uno de ellos, pero si tienes interés en conocer un poco más del resto, te dejamos aquí algunas referencias:*

- Stoyan Stefanov, [React: Up & Running, 2nd Edition](#). O'Reilly Media, Inc. 2021
- Aristeidis Bampakos, Pablo Deeelman. [Learning Angular - Third Edition](#). Packt Publishing. 2020

## Creación del proyecto

Para generar un proyecto con Vite es necesario ejecutar el comando de npm *create vite*. Este se encarga de lanzar la consola y guiarnos por el proceso de creación de proyectos.

```
npm create vite@latest
```

Tras esto aparecerá un pequeño menú para seleccionar el tipo de proyecto que deseamos crear. En nuestro caso seleccionaremos la primera opción basada en Vue 3 y JavaScript

```
zsh: command not found: npm
> npm create vite@latest

Need to install the following packages:
  create-vite@4.1.0
Ok to proceed? (y) y
✓ Project name: ... vite-project
✓ Select a framework: > Vue
✓ Select a variant: > JavaScript

Scaffolding project in /Users/Developer/vite-project...

Done. Now run:

  cd vite-project
  npm install
  npm run dev
```

Una vez terminado el proceso tenemos listo el entorno para realizar las siguientes PEC.