

Trabajo Práctico 5 - Herramientas de construcción de software

1- Ejemplo con C# y .NET Core

- Instalar el SDK de .NET Core: Asegúrate de tener el SDK de .NET Core instalado en tu sistema. Puedes descargarlo desde el sitio web oficial de .NET: <https://dotnet.microsoft.com/download>

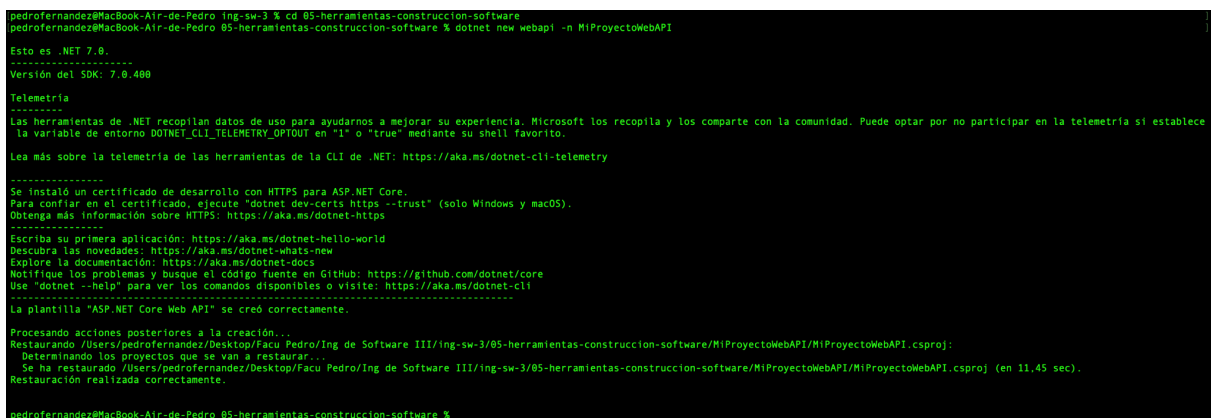


```
pedrofernandez — -zsh — 80x24
Last login: Thu Aug 31 14:21:32 on ttys000
pedrofernandez@MacBook-Air-de-Pedro ~ % dotnet --info
SDK DE .NET:
  Version:      7.0.400
  Commit:       73bf45718d

Entorno de tiempo de ejecución:
  OS Name:      Mac OS X
  OS Version:   13.4
  OS Platform:  Darwin
  RID:          osx.13-arm64
  Base Path:    /usr/local/share/dotnet/sdk/7.0.400/

Host:
  Version:      7.0.10
  Architecture: arm64
  Commit:       a6dbb800a4
```

- Crear un Proyecto de Web API:



```
pedrofernandez@MacBook-Air-de-Pedro: /ing-sw-3 % cd 05-herramientas-construccion-software
pedrofernandez@MacBook-Air-de-Pedro: 05-herramientas-construccion-software % dotnet new webapi -n MiProyectoWebAPI

Esto es .NET 7.0.
-----
Versión del SDK: 7.0.400

Telemetría
-----
Las herramientas de .NET recopilan datos de uso para ayudarnos a mejorar su experiencia. Microsoft los recopila y los comparte con la comunidad. Puede optar por no participar en la telemetría si establece la variable de entorno DOTNET_CLI_TELEMETRY_OPTOUT en "1" o "true" mediante su shell favorito.
Lea más sobre la telemetría de las herramientas de la CLI de .NET: https://aka.ms/dotnet-cli-telemetry

-----
Se instaló un certificado de desarrollo con HTTPS para ASP.NET Core.
Para confiar en el certificado, ejecute "dotnet dev-certs https --trust" (solo Windows y macOS).
Obtenga más información sobre HTTPS: https://aka.ms/dotnet-https

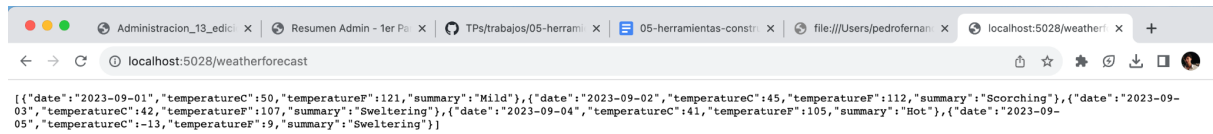
-----
Escriba su primera aplicación: https://aka.ms/dotnet-hello-world
Descubre las novedades: https://aka.ms/dotnet-whats-new
Explore la documentación: https://aka.ms/dotnet-docs
Notifique los problemas y busque el código fuente en GitHub: https://github.com/dotnet/core
Use "dotnet --help" para ver los comandos disponibles o visite: https://aka.ms/dotnet-cli

-----
La plantilla "ASP.NET Core Web API" se creó correctamente.

Procesando acciones posteriores a la creación...
Restaurando /Users/pedrofernandez/Desktop/Facu Pedro/Ing de Software III/ing-sw-3/05-herramientas-construccion-software/MiProyectoWebAPI/MiProyectoWebAPI.csproj:
Determinando los proyectos que se van a restaurar...
Se ha restaurado /Users/pedrofernandez/Desktop/Facu Pedro/Ing de Software III/ing-sw-3/05-herramientas-construccion-software/MiProyectoWebAPI/MiProyectoWebAPI.csproj (en 11,45 sec).
Restauración realizada correctamente.

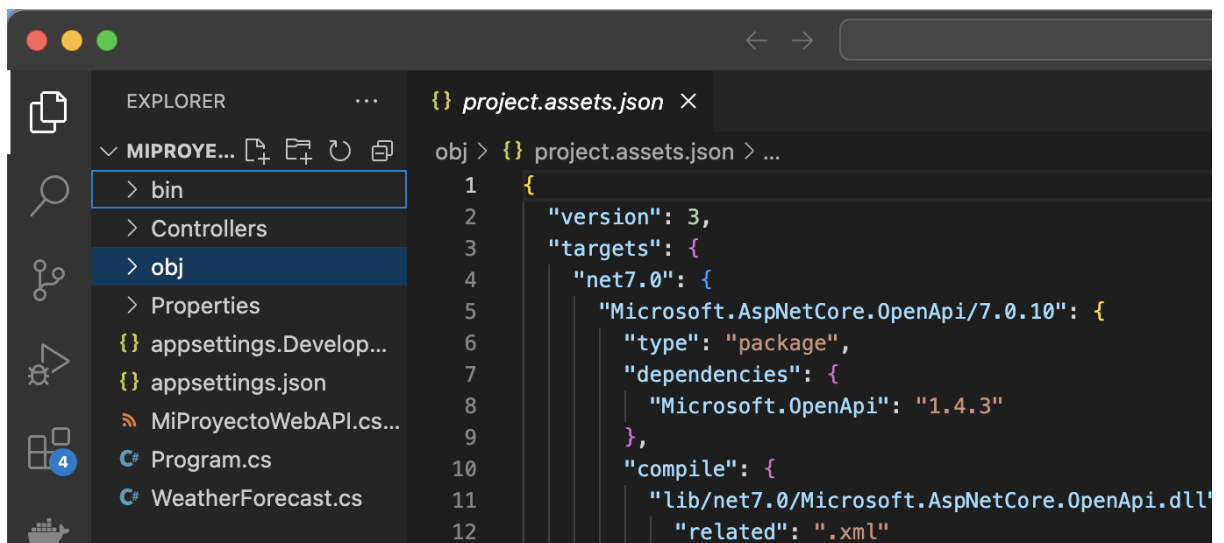
pedrofernandez@MacBook-Air-de-Pedro: 05-herramientas-construccion-software %
```

Ejecutar la Aplicación:

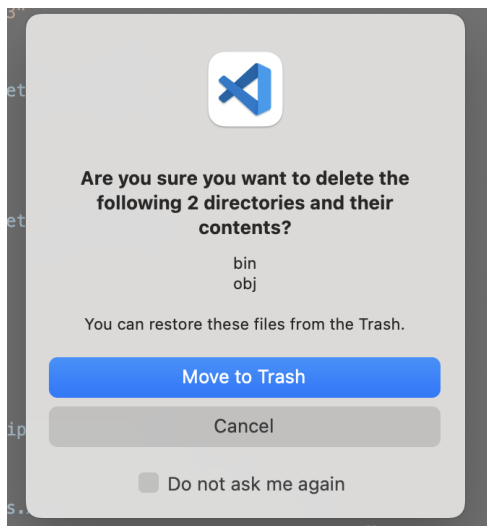


Esto iniciará la aplicación y la hará disponible en una URL local. Navegar a la url indicada en el mensaje recibido por consola añadiendo /weatherforecast

- Revisar el archivo MiProyectoWebAPI.csproj:
- Revisar el archivo obj/debug/project.assets.json



- Borrar directorios bin y obj



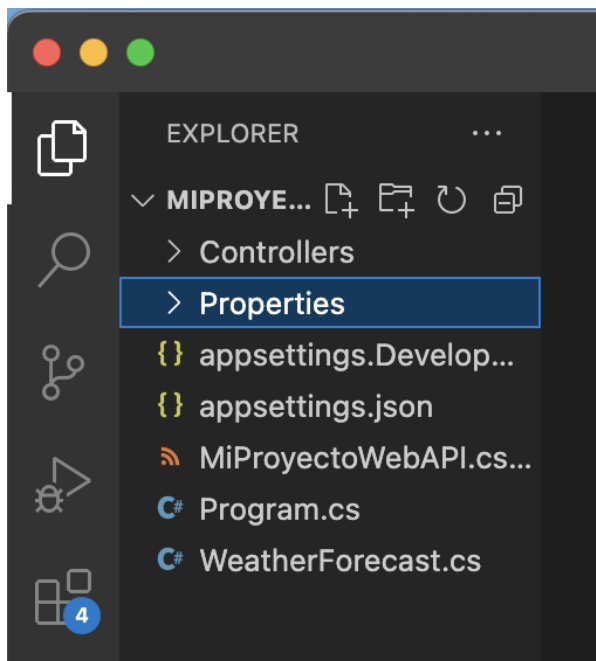
- Agregar una nueva referencia a librería Newtonsoft:

```

pedrofernandez@MacBook-Air-de-Pedro MiProyectoWebAPI % dotnet add package Newtonsoft.Json
Determinando los proyectos que se van a restaurar...
Writing /var/folders/4c/S16vRdJ4eqgU87fmw0nJ4dr0000gn/T/tmpKqU5F.tmp
Info : X.509 certificate chain validation will use the fallback certificate bundle at '/usr/local/share/dotnet/sdk/7.0.400/trustedroots/codesignctl.pem'.
Info : X.509 certificate chain validation will use the fallback certificate bundle at '/usr/local/share/dotnet/sdk/7.0.400/trustedroots/timestampctl.pem'.
Info : Agregando PackageReference para el paquete "Newtonsoft.Json" al proyecto "/Users/pedrofernandez/Desktop/Facu Pedro/Ing de Software III/ing-sw-3/05-herramientas-construccion-software/MiProyectoWebAPI/MiProyectoWebAPI.csproj".
Info : GET https://api.nuget.org/v3/registration5-gz-semver2/newtonsoft.json/index.json
Info : OK https://api.nuget.org/v3/registration5-gz-semver2/newtonsoft.json/index.json 918 ms
Info : Restaurando paquetes para /Users/pedrofernandez/Desktop/Facu Pedro/Ing de Software III/ing-sw-3/05-herramientas-construccion-software/MiProyectoWebAPI/MiProyectoWebAPI.csproj...
Info : GET https://api.nuget.org/v3-flatcontainer/newtonsoft.json/index.json
Info : OK https://api.nuget.org/v3-flatcontainer/newtonsoft.json/index.json 893 ms
Info : GET https://api.nuget.org/v3-flatcontainer/newtonsoft.json/13.0.3/newtonsoft.json.13.0.3.nupkg
Info : OK https://api.nuget.org/v3-flatcontainer/newtonsoft.json/13.0.3/newtonsoft.json.13.0.3.nupkg 37 ms
Info : Se instaló Newtonsoft.Json 13.0.3 de https://api.nuget.org/v3/index.json con el hash de contenido HfC5BXdl08IP9zeV+8Z848QWPaoCr9P3bDEZguI+gkLcBKA0xix/tLEAAHC+UvDNPv4a2d1810ReHM0agPa+zQ==.
Info : El paquete "Newtonsoft.Json" es compatible con todos los marcos de trabajo especificados del proyecto "/Users/pedrofernandez/Desktop/Facu Pedro/Ing de Software III/ing-sw-3/05-herramientas-construccion-software/MiProyectoWebAPI/MiProyectoWebAPI.csproj".
Info : Se agregó PackageReference para la versión "13.0.3" del paquete "Newtonsoft.Json" al archivo "/Users/pedrofernandez/Desktop/Facu Pedro/Ing de Software III/ing-sw-3/05-herramientas-construccion-software/MiProyectoWebAPI/obj/MiProyectoWebAPI.csproj.nuget.g.props".
Info : Generación de archivo MSBuild /Users/pedrofernandez/Desktop/Facu Pedro/Ing de Software III/ing-sw-3/05-herramientas-construccion-software/MiProyectoWebAPI/obj/MiProyectoWebAPI.csproj.nuget.g.targets

```

- Revisar nuevamente los archivos MiProyectoWebAPI.csproj y obj/debug/project.assets.json
- Borrar directorios bin y obj



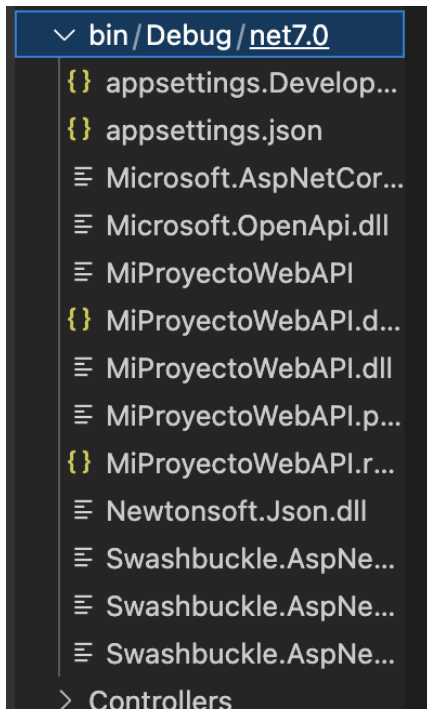
- Ejecutar nuevamente:

```

pedrofernandez@MacBook-Air-de-Pedro MiProyectoWebAPI % dotnet run
Compilando...
Info: Microsoft.Hosting.Lifetime[14]
Now listening on: http://localhost:5028
Info: Microsoft.Hosting.Lifetime[0]
Application started. Press Ctrl+C to shut down.
Info: Microsoft.Hosting.Lifetime[0]
Hosting environment: Development
Info: Microsoft.Hosting.Lifetime[0]
Content root path: /Users/pedrofernandez/Desktop/Facu Pedro/Ing de Software III/ing-sw-3/05-herramientas-construccion-software/MiProyectoWebAPI

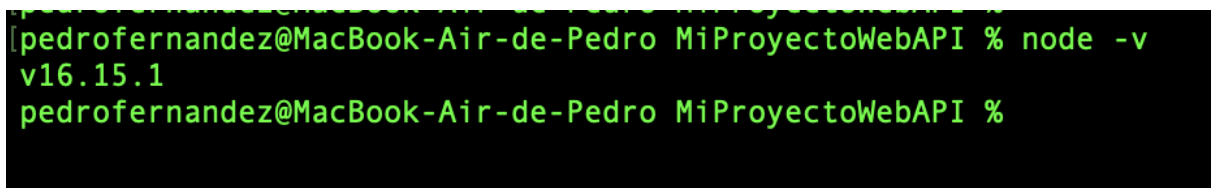
```

- Revisar contenido de directorio bin/debug/net7.0

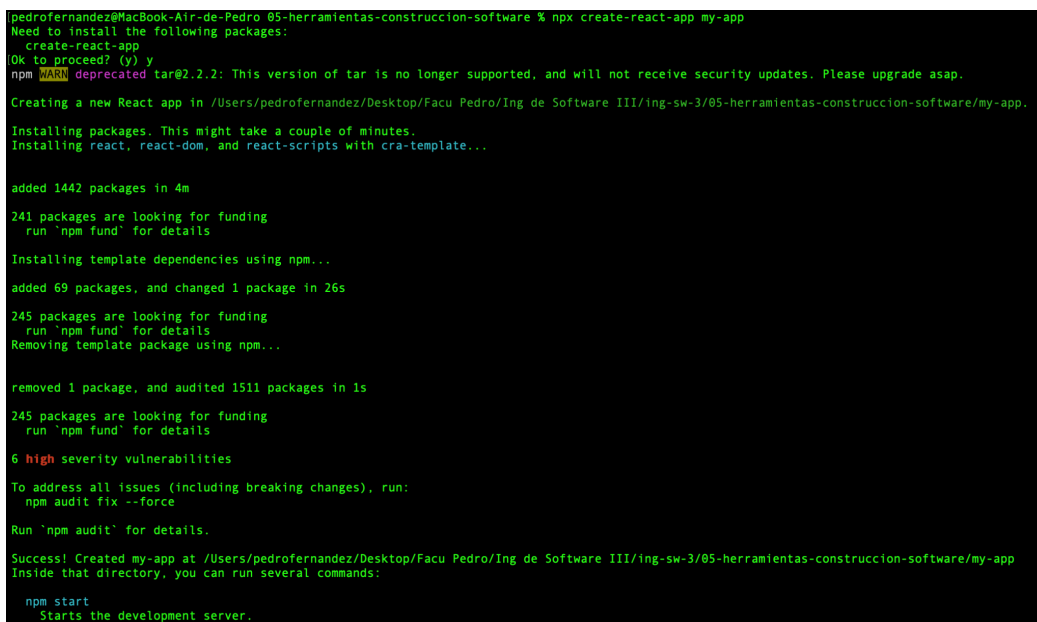


2- Ejemplo con nodejs

- Instalar Nodejs: <https://nodejs.org/en/>



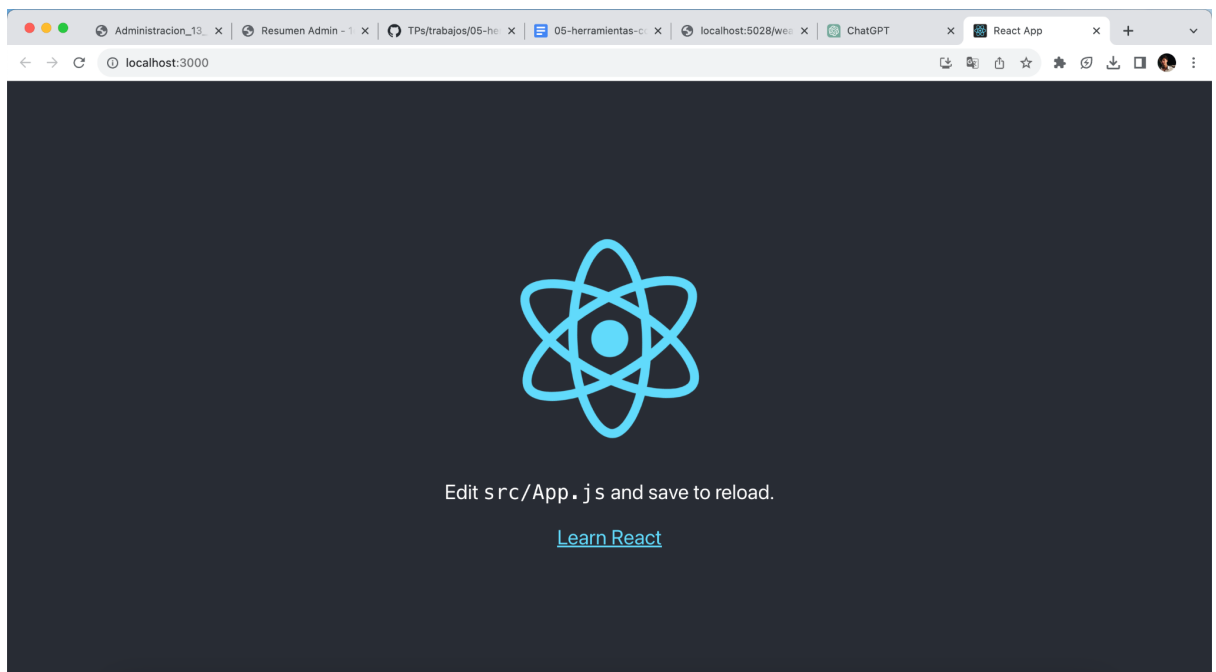
- Crear una nueva aplicación



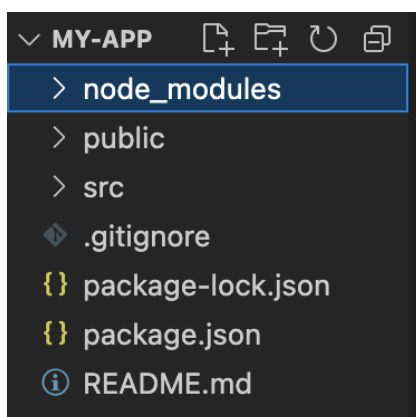
- Ejecutar la aplicación

```
Compiled successfully!  
  
You can now view my-app in the browser.  
  
http://localhost:3000  
  
Note that the development build is not optimized.  
To create a production build, use npm run build.  
  
webpack compiled successfully
```

- La aplicación web estará disponible en <http://localhost:3000>



- Analizar el manejo de paquetes y dependencias realizado por npm.



npm (Node Package Manager) se usa para administrar las dependencias y paquetes utilizados. Cuando se crea una nueva aplicación con create-react-app, automáticamente se generará un archivo llamado package.json en el proyecto. Este archivo contiene información sobre el proyecto, incluidas las dependencias, los scripts de ejecución y más.

Cuando se ejecuta el comando npm install (que se realiza automáticamente cuando usas create-react-app o manualmente si necesitas agregar nuevas dependencias), npm lee el package.json y descarga las dependencias necesarias desde el registro público de npm.

Las dependencias se almacenan en la carpeta node_modules. El archivo package-lock.json (ver imagen...)

3- Build tools para otros lenguajes

- Hacer una lista de herramientas de build (una o varias) para distintos lenguajes, por ejemplo (Rust -> cargo)
- Elegir al menos 10 lenguajes de la lista de top 20 o top 50 de tiobe:

<https://www.tiobe.com/tiobe-index/>

1. Python:

Herramienta: setuptools

Descripción: setuptools es una biblioteca que permite empaquetar, distribuir e instalar proyectos Python. También puede generar distribuciones, ejecutar pruebas y más.

2. Java:

Herramienta: Apache Maven

Descripción: Maven es una herramienta de construcción ampliamente utilizada para proyectos Java. Gestiona dependencias, compila, ejecuta pruebas y crea distribuciones.

3. C#:

Herramienta: MSBuild

Descripción: MSBuild es la herramienta de construcción de Microsoft utilizada para compilar y empaquetar proyectos .NET. Es ampliamente utilizado en el ecosistema de desarrollo de Microsoft.

4. C++:

Herramienta: CMake

Descripción: CMake es una herramienta de código abierto que genera archivos de construcción para diferentes sistemas operativos y entornos de compilación. Es ampliamente utilizado para proyectos C++.

5. **JavaScript:**

Herramienta: webpack

Descripción: webpack es una herramienta de construcción para aplicaciones JavaScript. Se utiliza comúnmente para empaquetar módulos y activos en una sola salida optimizada.

6. **PHP:**

Herramienta: Composer

Descripción: Composer es una herramienta de administración de dependencias para proyectos PHP. Facilita la instalación y gestión de paquetes y bibliotecas.

7. **Swift:**

Herramienta: Swift Package Manager

Descripción: Swift Package Manager es la herramienta oficial de gestión de paquetes para proyectos Swift. Permite la gestión de dependencias y la creación de módulos reutilizables.

8. **Ruby:**

Herramienta: RubyGems

Descripción: RubyGems es el sistema de gestión de paquetes estándar para proyectos Ruby. Permite la instalación y distribución de bibliotecas y gemas.

9. **Kotlin:**

Herramienta: Gradle

Descripción: Gradle es una herramienta de construcción utilizada en el ecosistema de desarrollo de Android, así como para proyectos Kotlin en general. Ofrece flexibilidad y gestión de dependencias.

10. **R:**

Herramienta: devtools

Descripción: devtools es un paquete en R que proporciona herramientas para el desarrollo de paquetes R. Facilita la construcción, instalación y prueba de paquetes personalizados.