I. E. S. MARE NOSTRUM
Beato Fco. Castelló Aleu s/n • ☎ 965936520
03008 ALICANTE
www.iesmarenostrum.com
correo@iesmarenostrum.com

Práctica: Documentación automática de código en Visual Studio

Contenido

En esta práctica vamos a generar la documentación técnica del código C# usando los comentarios de documentación XML.

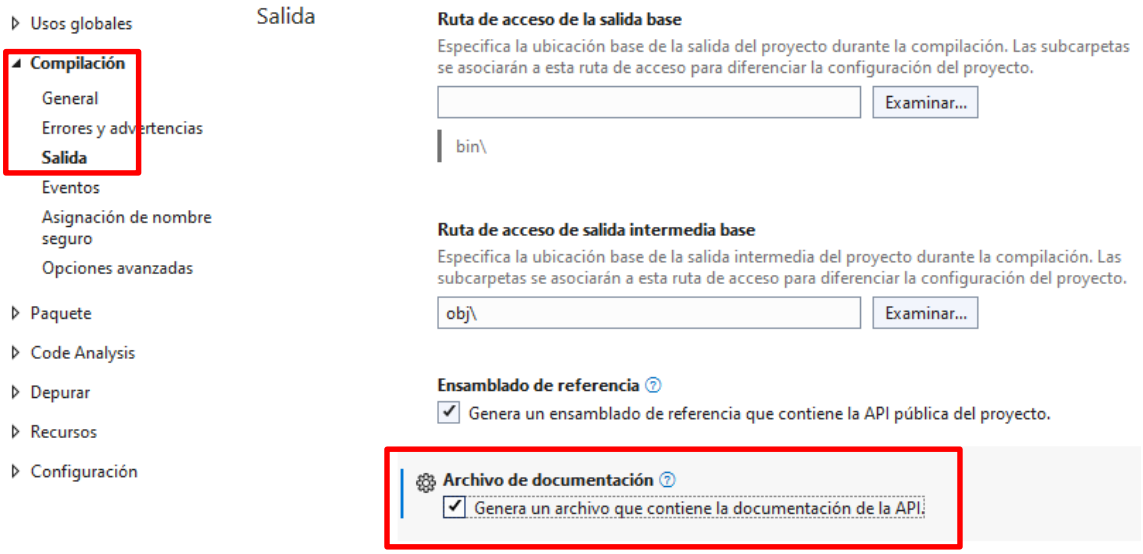
1. Creación y documentación de un proyecto

1. En primer lugar vamos a crear un proyecto, llamado **ProyectoDocumentacion**, en el que vamos a crear una clase **Trabajador** (que representa a un trabajador de una hipotética empresa), en la cual documentaremos sus elementos más importantes: la clase, los campos, los métodos y las propiedades. El namespace no se puede documentar.

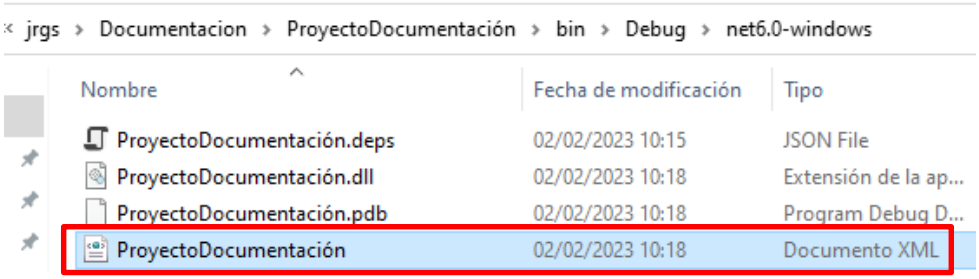
   Puedes descargarte la clase Trabajador con o sin comentarios para trabajar sobre ella.

   Si trabajamos con Visual Studio, para poner un comentario de documentación basta con escribir tres barras /// encima del elemento que queremos comentar para que nos autocomplete el comentarios con las etiquetas que detecte. Por ejemplo, si escribimos /// en la línea encima del método **CalculoAnyosJubilacion(int edadJubilacion)** veremos que nos crea las etiquetas **summary**, **param** y **returns**, nosotros sólo tenemos que completar el contenido.

2. Una vez completada la documentación del código, tenemos que configurar el proyecto para que genere el archivo xml cada vez que se compile. Hay que hacer click derecho en el nombre del proyecto → P*ropiedades* → *Compilación* → *Salida* y marcar la casilla ***Archivo de documentación XML***. También podemos cambiar la ruta donde se guardará el archivo, que por defecto será en bin\Debug dentro de nuestro directorio de proyecto.



3. Ahora procedemos a compilar (ejecutar) y buscamos el archivo generado en la ruta anterior:




```
<doc>
  <assembly>
    <name>ProyectoDocumentación</name>
  </assembly>
  <members>
    <member name="T:ProyectoDocumentación.Trabajador">
      <summary>
        <para>Clase que representa a un trabajador de una empresa.</para>
        <para>
          Los trabajadores pueden ser de las siguiente secciones:
          <list type="bullet">
            <item>
              <description>Marketing</description>
            </item>
            <item>
              <description>Ventas</description>
            </item>
            <item>
              <description>Logística</description>
            </item>
          </list>
        </para>
      </summary>
    </member>
    <member name="F:ProyectoDocumentación.Trabajador.EdadJubilacionDefecto">
      <summary> Edad por defecto de jubilación del trabajador </summary>
    </member>
    <member name="F:ProyectoDocumentación.Trabajador.nombre">
      <summary> Nombre y apellidos del trabajador </summary>
```

2. Generación de documentación navegable

Como hemos visto, la documentación generada por Visual Studio es un simple archivo xml que no resulta navegable y es de poca utilidad. Para generar documentación en el formato de ayuda de Microsoft debemos usar programas o extensiones de terceros. En esta práctica usaremos el programa **Sandcastle HelpFile Builder** (no confundir con Sandcastle a secas, que es el proyecto que mantenía Microsoft pero que abandonó hace años).

4. Descargamos el programa de su web oficial: https://github.com/EWSoftware/SHFB/releases

5. Una vez descargado, descomprimimos y ejecutamos el instalador **SandcastleInstaller.exe**. El proceso de instalación consta de varios apartados y en algunos tenemos que instalar los componentes que nos propone.

   Después de la pantalla de bienvenida, otra con las novedades y otra en la que se comprueba que tenemos la versión correcta del Framework .NET, viene otra en la que nos informa los formatos de salida que puede generar. Nosotros sólo vamos a generar salidas en formato **HTML Help 1**, pero conviene leer toda la información para saber cómo generar salidas en el resto de formatos.

6. Si no está instalado el compilador de HTML Help 1, nos pedirá que lo instalemos, siguiendo las instrucciones que proporciona.

7. En los siguientes pasos nos pide que instalemos los diversos componentes del programa. Los más importantes son **Sandcastle Help Builder and Tools**, que es la aplicación independiente, y **SHFB Visual**

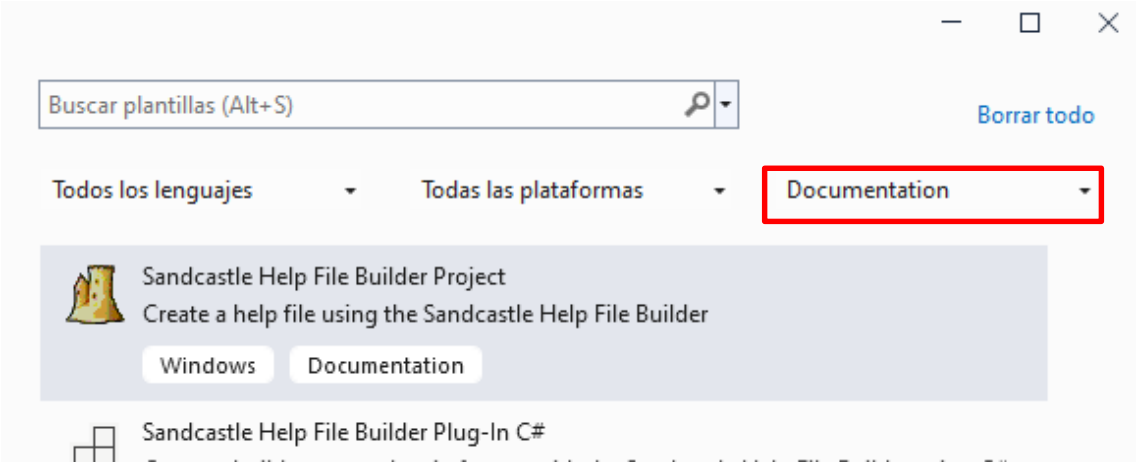
Studio Package, que es la extensión de Visual Studio. Instalamos ambos y seguimos. El resto de componentes los instalamos también, aunque no los usaremos. Una vez finalidad la instalación, reiniciamos el equipo.

Para crear nuestros archivos de ayuda tenemos 2 opciones: usar el programa independiente o usar la extensión de Visual Studio. Nosotros usaremos la extensión, pero el proceso es similar en ambos.

Nota: al instalar la extensión, deberemos tener Visual Studio cerrado. Si no es así, el instalador esperará hasta que lo cerremos manualmente. También se puede cerrar dándole al botón 'finalizar tareas', pero perderemos el trabajo de Visual Studio.

Finalizaremos instalando los Schemas y los Snippets. El Spell Checker (comprobador de sintaxis) se puede instalar opcionalmente como extensión.

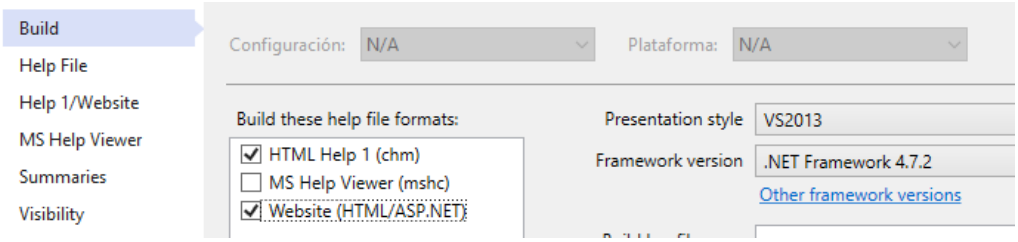
9. Primero debemos crear un proyecto de documentación en Visual Studio: *Archivo → Nuevo → Proyecto → Documentación → Sandcastle Help File Builder Project*. Ponemos nombre al proyecto, **Documentación Trabajador**.



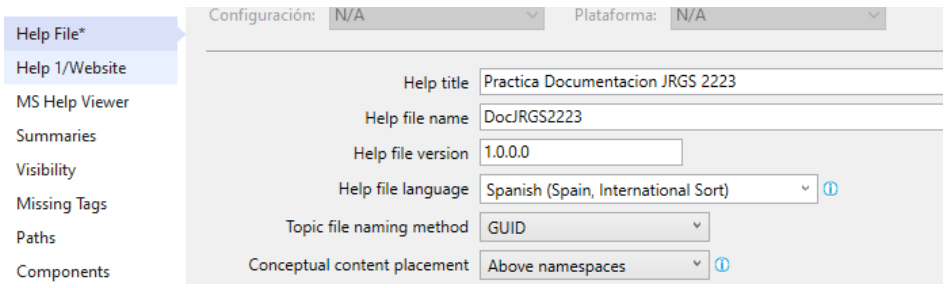
11. Vemos que se genera un nuevo proyecto (de documentación, en este no podemos programar) con un archivo de ejemplo llamado Welcome.aml. Lo cerramos y pasamos a configurar nuestro proyecto.

12. Primero configuramos el proyecto. Dentro del **Explorador de Soluciones** hacemos doble click en **Project Properties**. Sólo configuraremos las opciones básicas:

    - Apartado **Build**: Estos son los tipos de salidas que generará. Podemos generar un archivo .chm, un sitio web, o ambas cosas. Importante marcar el **Presentation Style** de VS2013.



    - Apartado **Help File**: Aquí establecemos el idioma del documento así como el título, nombre del autor, etc. Cambiamos el idioma a Spanish, el título (**Help Title**) y ponemos nuestro nombre en **Additional header content**. También podemos cambiar el nombre del archivo que se generará, poner el tipo de licencia, nuestro e-mail, etc.

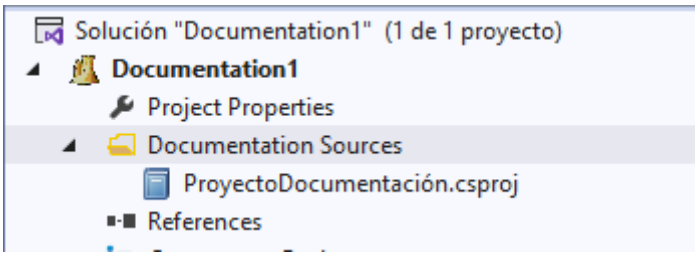


    - Apartado **Visibilty**: Aquí de momento no tocaremos nada, pero es un apartado importante porque indica los elementos de nuestra API

que queremos incluir en la documentación en base a su visibilidad. En nuestro caso, los campos no se mostrarán porque son privados, pero si queremos que aparezcan tendríamos que marcar la casilla **Private fields**.

Guardamos y seguimos, pero es interesante leer el resto de opciones.

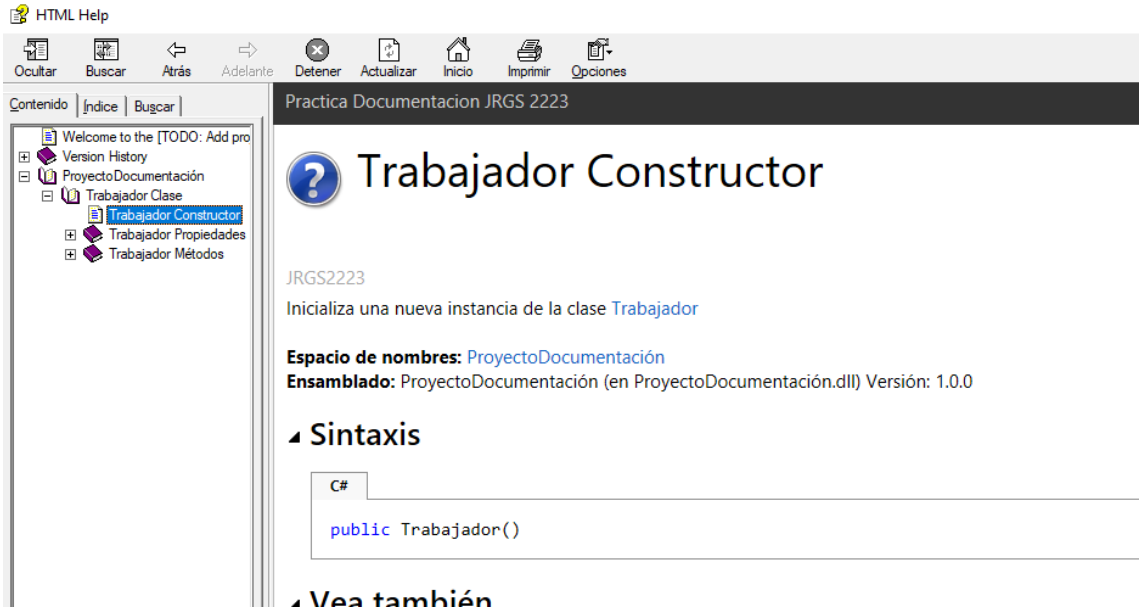
13. Lo siguiente es añadir los archivos a partir de los cuales queremos extraer la documentación. Hacemos click derecho en **Documentation Sources** (en el **Explorador de soluciones**) → *Add Documentation Source* y buscamos el archivo del proyecto donde tenemos el código fuente (extensión .csproj). También podemos añadir ejecutables compilados (exe), archivos XML, librerías, (dll), etc.



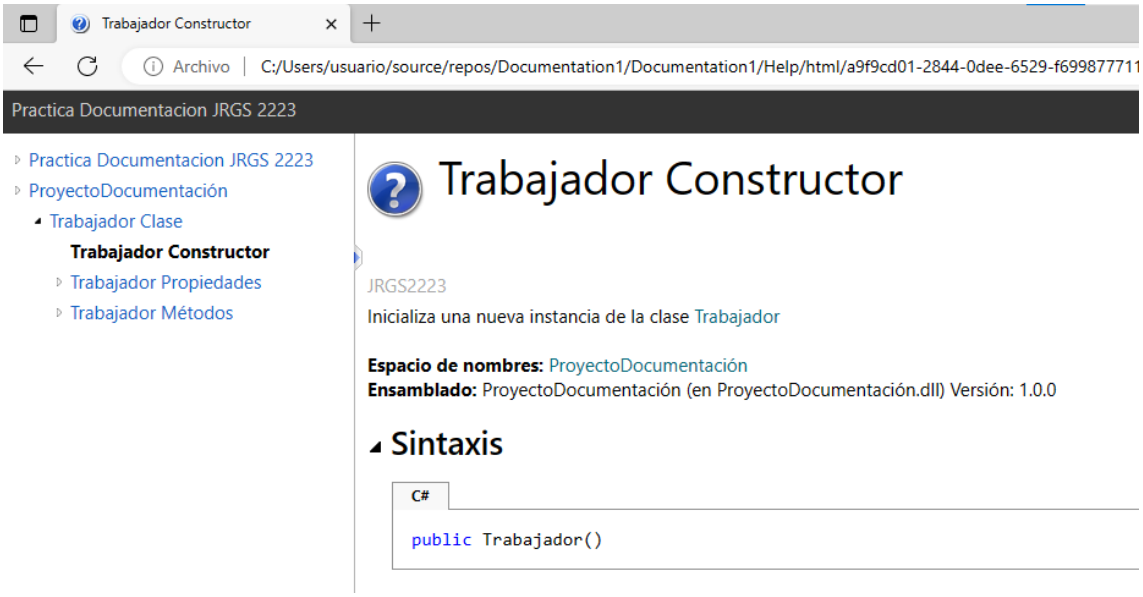
14. Ya podemos crear el archivo de documentación simplemente pulsando en **Iniciar**. Tardará un buen rato. Una vez terminado (podemos seguir todo el proceso en la ventana Resultados) y si no se han producido errores, tendremos el archivo de ayuda (con extensión **chm**) dentro de la carpeta de nuestro proyecto de documentación, en la subcarpeta Help.

15. Para ver el resultado, podemos abrir directamente el archivo chm o podemos añadir y usar la barra de herramientas de Sandcastle: V*er → Barras de herramientas → Sandcastle Help File Builder*.

    El resultado debería ser similar a este:



O, en versión Website:



3. Práctica a entregar

El alumno deberá documentar y generar el archivo de ayuda en formato chm de la clase **Cuenta.cs**, que representa una cuenta corriente con operaciones para consultar el saldo, ingresar y retirar dinero.

I. E. S. MARE NOSTRUM
Beato Fco. Castelló Aleu s/n • ☎ 965936520
03008 ALICANTE
www.iesmarenostrum.com
correo@iesmarenostrum.com

Se tendrá que documentar los siguiente elementos: **clase**, **campos**, **propiedades** y **métodos**. Deberán usarse obligatoriamente las etiquetas **summary**, **remarks**, **param**, **returns**, **value**. El resto de etiquetas son opcionales. Para realizar este apartado, se puede utilizar un plugin como *Code Formatter*, visto en la práctica anterior.

Se deberá entregar **solamente** el código de la clase con los comentarios y el archivo de ayuda (extensión **chm**), comprimidos en un archivo **zip** (**rar** no, por favor). Como en el resto de prácticas, en el archivo de ayuda deben aparecer vuestras iniciales y el año actual. La forma más sencilla es incluirla en los métodos/propiedades.