Frameworks, librerías, IDEs y editores de texto Certified Tech Developer

The Ultimate Degree

**Digital**House >

#### Índice

- 1. Framework
- 2. <u>Librerías</u>
- 3. <u>Diferencias entre</u> <u>frameworks y librerías</u>
- 4. Editores de texto
- 5. IDEs

## 1 Frameworks

Un framework es un patrón o esquema que ayuda a la programación a estructurar el código, ahorrando tiempo y esfuerzos a los programadores.





#### **Frameworks**

Un framework es un conjunto de herramientas que nos facilitan el desarrollo de software. Para poder lograrlo incluyen implementos como los siguientes:

- APIs
- Librerías
- Herramientas de depuración
- Edición
- Prototipado
- Programas de soporte













#### **Frameworks**

Los frameworks no están ligados necesariamente a un lenguaje concreto, aunque así sea en muchas ocasiones. Por ejemplo, en Ruby on Rails, "Ruby" es el lenguaje de programación y "Rails", el framework.

Sin embargo, no existen impedimentos para definir el mismo framework para lenguajes diferentes.

También es posible que el framework defina una estructura para una aplicación completa, o bien solo se centre en un aspecto de ella.







# 2 Librerías



Una librería no es más que un conjunto de código que alguien ha realizado para que podamos reutilizar dentro de nuestros proyectos.



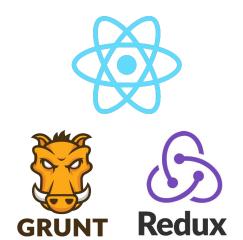




#### Librerías

El objetivo de una librería no es otro que hacer más fácil y rápido el desarrollo de ciertas funciones dentro de nuestro programa o aplicación.

Normalmente las librerías están enfocadas a solucionar problemas concretos. Es decir, no nos brindan una estructura para nuestro proyecto, pero sí van a ayudar a resolver funcionalidades específicas.









# 3 Diferencias entre frameworks y librerías

#### Librerías vs. Frameworks

Una **librería** es un código escrito previamente, ya utilizado por otros desarrolladores, listo para que lo utilicemos y pretende hacernos la vida más fácil y su trabajo más rápido.



Un **framework** es un esqueleto. Podría decirse que es como el marco de una casa. La estructura está predeterminada, y el trabajo del desarrollador es llenar los vacíos con su código.

## 4 Editores de texto

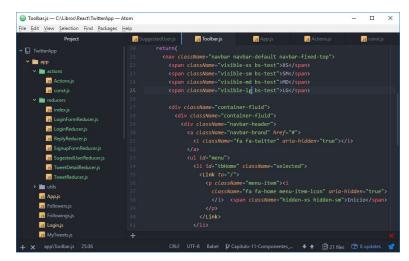
Los editores de texto **se crearon** para mostrar el código de una forma agradable y realizaban algunas acciones muy simples. Sin embargo, los editores de texto modernos, siguen agregando capacidades que solo los IDEs tenían.



#### Editores de texto

Los editores de texto son herramientas mucho más simples y compactas. Proporcionan un entorno de desarrollo simple. Los editores de texto tienen la peculiaridad de trabajar con archivos de texto y carpetas, es decir, al abrir una carpeta podemos trabajar con todo lo que hay dentro.

Existen editores sofisticados como Atom, Sublime Text y Brackets, los cuales muestran el código de una forma pintoresca y atractiva.



```
def bitblt_fast(self, src, x, y):
    """Blit without range checks, clipping and a hardwired rop_copy
    raster operation.
"Construct and return a Glyph object from a FreeType GlyphSlot.""
                                                                pixels = Glyph.unpack_mono_bitmap(slot.bitmap)
width, height = slot.bitmap.width, slot.bitmap.rows
top = slot.bitmap_top
                                                                                                                                                                                                                                     pixels = self.pixels
src_width, src_height = src.width, src.height
                                                                # The advance width is given in FreeType's 26.6 fixed point format, # which means that the pixel values are multiples of 64. advance_x = slot.advance.x / 64
                                                                                                                                                                                                                                      dstpixel = y * width + x
                                                                                                                                                                                                                                     for _ in range(src_height):
    for _ in range(src_width):
    pixels[dstpixel] = src_pixels[srcpixel]
    srcpixel += 1
                                                               unpack_mono_bitmap(bitmap):
    """Unpack a freetype FT_LOAD_TARGET_MONO glyph bitmap into a
                                                                                                                                                                                                                                 def bitblt(self, src, x=0, y=0, op=rop_copy):
                                                                                                                                                                                                                                     # This is the area within the current surface we want to draw in.
# It potentially lies outside of the bounds of the current surface
                                                                # iterating over every pixel in the resulting unpacked bitmap
# we're iterating over the packed bytes in the input bitmap.
                                                                                                                                                                                                                                     dstrect = Rect(x, y, src.width, src.height)
cliprect = self.rect.clipped(dstrect)
                                                                              # Read the byte that contains the packed pixel data.
byte_value = bitmap.buffer[y * bitmap.pitch + byte_index]
                                                                                                                                                                                                                                     # the left or top edge.

xoffs = src.width - cliprect.width if x <= 0 else 0
yoffs = src.height - cliprect.height if y <= 0 else 0
                                                                              # We've processed this many bits (=pixels) so far
# This determines where we'll read the next batch
                                                                                                                                                                                                                                     # Copy pixels from 'src' to 'cliprect'
dstrowwidth = cliprect.rx · cliprect.x
srcpixel = yoffs * src._width + xoffs
                                                                              # of pixels from.
num bits done = byte index * 8
```

5 IDEs



Los IDEs, a diferencia de los editores de texto, no trabajan con archivos y carpetas. En su lugar, emplean el concepto de proyectos.





#### **IDEs**

Un **IDE o** *Integrated Development Environment* (entornos de desarrollo integrado) posee muchas más herramientas que se integran en un mismo programa.

Los IDEs tienen como principal característica que **no trabajan con archivos y carpetas.** En su lugar, **emplean el concepto de proyectos**.

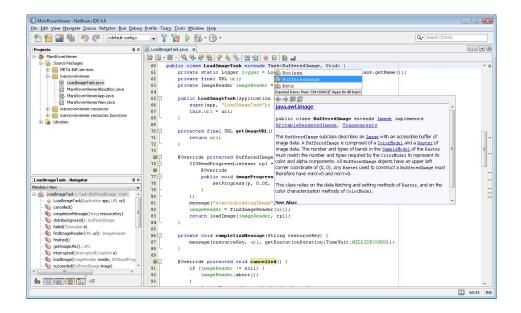
Un proyecto es una carpeta en el disco duro, pero tiene la diferencia que el IDE **crea archivos adicionales al código para optimizar la experiencia del usuario**. En estos archivos puede tener configuraciones de ejecución, deploy, tipo de proyecto, etc.

Debido a que los IDEs son plataformas muy complejas, es posible hacer un sinfín de cosas, y los plugins que ofrecen son prácticamente ilimitados.

#### **IDEs**

Las características que más resaltan de un IDF son:

- Debugger en tiempo real.
- Visualiza gráficamente casi cualquier cosa, desde XML, JSON, UML, bases de datos, interfaces gráficas, etc.
- Ayuda en tiempo real.



### DigitalHouse>