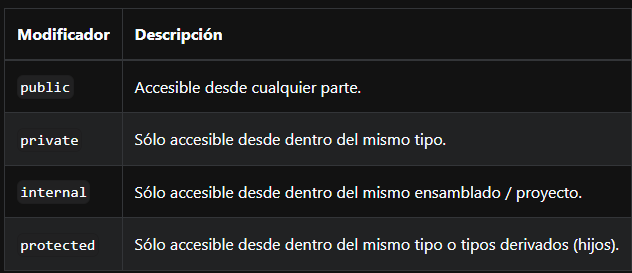
**Métodos Estáticos:**

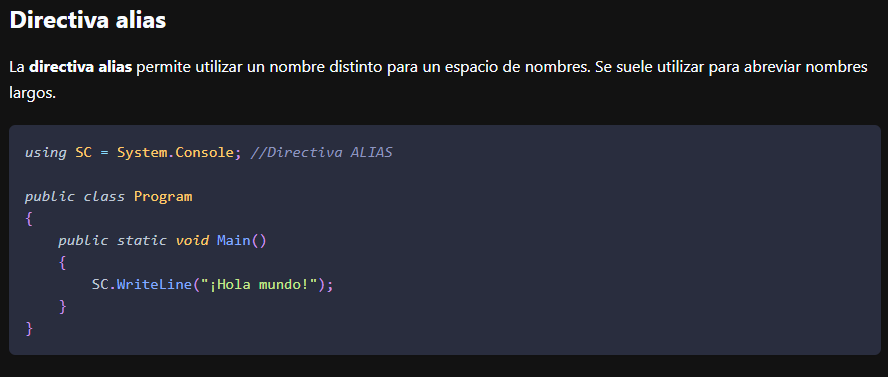
Los **métodos estáticos** son pequeñas piezas de código que se encargan de realizar una tarea concreta. También se los suele llamar como **métodos de clase**.   
**Modificador de acceso**: Indica y limita desde donde se puede interactuar con un método u otros tipos de miembros. Por defecto es private.



**Clases Estáticas:**

Las **clases estáticas** son agrupadores de atributos estáticos y métodos estáticos, es decir de datos y funciones que operan sobre esos datos respectivamente.  
Las constantes son como atributos estáticos, la única diferencia es que su valor no puede cambiar en tiempo de ejecución.

Si seguimos buenas prácticas de programación, los atributos de una clase siempre deberán ser privados.



**Métodos estáticos**[**#**](https://codeutnfra.github.io/programacion_2_laboratorio_2_apuntes/docs/clases/estaticos/apuntes/cuestionario#m%C3%A9todos-est%C3%A1ticos)

1. ¿Qué propone el principio DRY?

Dry, DON´T REPEAT YOURSELF, propone que toda pieza de conocimiento debe tener una representación única, inequívoca y fidedigna dentro de un sistema.  
La idea del principio DRY es simple: Cuando ocurra un cambio no deberíamos necesitar actualizar múltiples cosas en paralelo. Si el conocimiento está duplicado en el código y necesitamos hacer un cambio, podríamos olvidar hacer las modificaciones en todos lados. En la documentación, podría conducir a malas interpretaciones, confusión y, en última instancia, implementaciones incorrectas.

1. ¿Qué modificadores podemos aplicar a un método y qué efecto tienen?

* Public, se puede acceder a ese método desde cualquier parte.
* Private, solo es accesible dentro del mismo tipo.
* Internal, solo es accesible dentro del mismo proyecto.
* Protected, solo es accesible dentro de un mismo tipo o derivados (hijos)
* Static,

1. Enumere y detalle las buenas prácticas a la hora de diseñar un método.
2. ¿Qué significa refactorizar?

**Namespaces[#](https://codeutnfra.github.io/programacion_2_laboratorio_2_apuntes/docs/clases/estaticos/apuntes/cuestionario" \l "namespaces" \o "Direct link to heading)**

1. ¿Qué es un namespace y cuál es su función principal?
2. ¿Puedo tener distintos namespaces dentro de un mismo proyecto o ensamblado de .NET?
3. ¿Para qué se usa la directiva using?
4. ¿Para qué se usa la directiva alias?
5. ¿Puedo declarar dos clases independientes/distintas con el mismo identificador dentro del mismo namespace? ¿Y en namespaces distintos?

**Strings[#](https://codeutnfra.github.io/programacion_2_laboratorio_2_apuntes/docs/clases/estaticos/apuntes/cuestionario" \l "strings" \o "Direct link to heading)**

1. ¿Es lo mismo declarar una variable como string (en minúscula) o como String (con la primera letra en mayúscula)? ¿Por qué?
2. ¿Se puede recorrer un string con un foreach? ¿Por qué?
3. ¿Qué significa que los strings son inmutables? ¿qué sucede en realidad cuando usamos métodos u operadores para modificar un string?