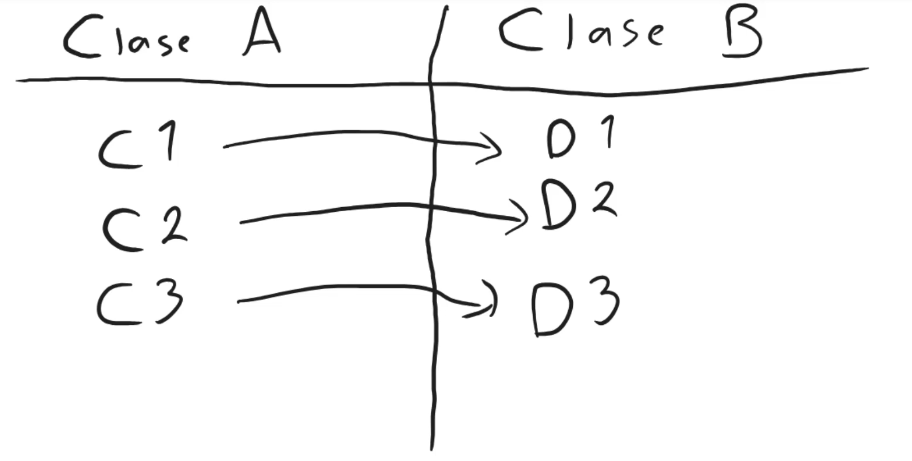
**📌 Resumen – AutoMapper**



**🔹 ¿Qué es AutoMapper?**

* Es una herramienta que evita tener que asignar propiedades **campo por campo** entre objetos (ej. de un DTO a una entidad o viceversa).
* Permite transformar un objeto origen en un objeto destino de manera automática y reutilizable, reduciendo código repetitivo.

**🔹 Instalación y configuración**

1. Instalar paquete NuGet: **AutoMapper.Extensions.Microsoft.DependencyInjection**.
2. Crear carpeta AutoMapper y clase MappingProfile que herede de Profile.
3. Registrar el perfil en Program.cs con:

builder.Services.AddAutoMapper(typeof(MappingProfile));

**🔹 Casos de uso principales**

1. **Propiedades con el mismo nombre**
   * Si origen y destino tienen campos con el mismo nombre (Name, Alcohol, BrandID…), AutoMapper los asigna automáticamente.
   * Ejemplo:

CreateMap<BeerInsertDto, Beer>();

* + Uso:

var beer = \_mapper.Map<Beer>(beerInsertDto);

1. **Propiedades con distinto nombre**
   * Cuando los nombres no coinciden (BeerID vs Id), se usa .ForMember:

CreateMap<Beer, BeerDto>()

.ForMember(dto => dto.Id,

m => m.MapFrom(b => b.BeerID));

* + Así se define explícitamente qué campo de origen llena a cuál campo de destino.

1. **Mapeo sobre objeto existente (Update)**
   * Útil cuando ya tenés una entidad cargada de la BD y querés actualizarla con valores de un DTO.
   * Ejemplo:

\_mapper.Map(beerUpdateDto, beerExistente);

* + 👉 Solo se actualizan las propiedades presentes en el DTO.
  + Los valores que no vengan en el DTO (ej. Id) se mantienen intactos.

**🔹 Beneficios principales**

* **Menos código repetido**: se eliminan asignaciones manuales.
* **Reutilización**: el mismo perfil sirve en cualquier parte del proyecto.
* **Claridad**: las reglas de transformación quedan centralizadas en MappingProfile.
* **Consistencia**: se asegura que todos los mapeos sigan las mismas reglas.