# 21. Parámetros en métodos HTTP (ASP.NET Core)

Cuando escribís un método en un controlador, ASP.NET Core se encarga de rellenar automáticamente los parámetros con los valores que llegan en la request. Este proceso se llama Model Binding.

## ¿Cómo sabe de dónde sacar el valor?

ASP.NET Core tiene reglas de búsqueda. Según el atributo que uses (o no uses), sabe si debe mirar en la URL, en la query string, en el body, en los headers o en un formulario.

## 1. [FromRoute] → Valor en la ruta (segmento de la URL)

[HttpGet("search/{search}")]  
public IActionResult Get([FromRoute] string search)  
  
👉 URL: /api/people/search/nico  
✅ search = "nico"  
  
¿Cuándo usarlo?  
- Cuando el dato identifica un recurso de forma única.  
- Ej: /users/15 → quiero ese usuario con ID=15.  
- Se usa en GET principalmente.  
- Es más legible y se puede cachear/facilitar en SEO.

## 2. [FromQuery] → Query string (?key=value)

[HttpGet("search")]  
public IActionResult Get([FromQuery] string search)  
  
👉 URL: /api/people/search?search=nico  
✅ search = "nico"  
  
¿Cuándo usarlo?  
- Cuando el dato es un filtro, búsqueda o parámetro opcional.  
- Ej: /products?category=ropa&page=2.  
- Ideal para GETs con filtros o paginación.  
- Permite combinar varios parámetros sin alterar la ruta.

## 3. [FromBody] → Cuerpo de la request (JSON en POST/PUT/PATCH)

[HttpPost("search")]  
public IActionResult Post([FromBody] string search)  
  
👉 Body JSON: { "search": "nico" }  
✅ search = "nico"  
  
¿Cuándo usarlo?  
- Cuando mandás información compleja (objetos, listas, formularios grandes).  
- Ej: crear usuario → { "name": "Nico", "email": "nico@test.com" }.  
- Usado en POST/PUT/PATCH, nunca en GET.  
- El body es más seguro y flexible que la URL para datos sensibles o largos.

## 4. [FromHeader] → Headers personalizados

public IActionResult Get([FromHeader(Name = "x-custom-header")] string value)  
  
👉 Header: x-custom-header: hola  
✅ value = "hola"  
  
¿Cuándo usarlo?  
- Para metadatos de la request, no para los datos principales.  
- Ej: Authorization: Bearer <token> (autenticación).  
- También útil para tracking, versiones de API, claves API.  
- No se usa para información de negocio (ej: nombre del usuario), sino para datos de control.

## 5. [FromForm] → Datos de formularios HTML (form-data o multipart/form-data)

[HttpPost("upload")]  
public IActionResult Upload([FromForm] string name)  
  
👉 FormData: name=nico  
✅ name = "nico"  
  
¿Cuándo usarlo?  
- Cuando el front manda datos desde un formulario HTML tradicional.  
- Especialmente útil en subida de archivos (porque se codifican como multipart/form-data).  
- Ej: un form con inputs de texto + un archivo adjunto.  
- Si solo son textos, suele ser mejor [FromBody] con JSON.

# Resumen rápido (con lógica de uso)

|  |  |  |
| --- | --- | --- |
| Fuente | Ejemplo URL / Data | ¿Cuándo usarlo? |
| [FromRoute] | /users/15 | Identificador único de un recurso |
| [FromQuery] | /products?category=ropa&page=2 | Filtros, búsqueda, parámetros opcionales |
| [FromBody] | { "name": "Nico", "email": "test@test.com" } | Objetos complejos, datos sensibles, POST/PUT |
| [FromHeader] | Authorization: Bearer token123 | Metadatos: auth, tracking, API keys |
| [FromForm] | name=nico + archivo.jpg | Formularios HTML, especialmente subida de archivos |