Paginação

O **Spring Data JPA** fornece uma implementação de página, a maneira mais básica é trabalhar com os objetos **Pageable** e **Page,** o **Pageable** cria um **"padrão"** *(baseado nos parâmetros passados)* **de como a página irá se comportar** *(tamanho, qual página, ordenação etc...):*

```
@GetMapping("/pagina")
public Page<CursoDto> buscarCursosPaginados(@RequestParam int pagina, @RequestParam int qtdPorPagina
                                            @RequestParam String ordenacao){

    Pageable paginacao = PageRequest.of(pagina, qtdPorPagina, Sort.Direction.ASC, ordenacao);
    Page<Curso> cursos = cursoRepository.findAll(paginacao);

    return CursoDto.converterPaginado(cursos);
}
```

Existe um modo de oferecer uma **flexibilidade maior da paginação ao cliente da API** de maneira muito simples, usando o próprio suporte do **Spring Web.** Por **default esse suporte não é habilitado**, então é preciso **habilitar no main com uma anotação**

```
@SpringBootApplication
@EnableSpringDataWebSupport
public class Application {

    public static void main(String[] args) { SpringApplicat

}
```

A ideia é que **quem for consumir essa API passe os padrões direto na URL** e o **Spring** automaticamente **vai passar esses padrões** ao **Spring Data**, enquanto no **método do controller é necessário apenas declarar o Pageable como parâmetro** *(inclusive sendo possível oferecer uma configuração padrão):*

```
@GetMapping("/pagina")
public Page<UsuarioDto> buscaFlexivelPaginada(
        @PageableDefault(sort = "id", direction = Sort.Direction.ASC,page = 0, size = 5) Pageable paginacao){

    Page<Usuario> usuarios = usuarioRepository.findAll(paginacao);
    return UsuarioDto.converterPaginado(usuarios);
}
```

Obviamente os **parâmetros da URL** têm que ser passados em inglês: