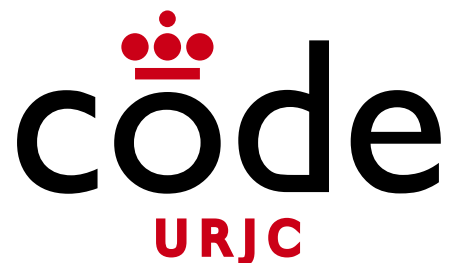


Desarrollo Web

Tecnologías de servidor web

Tema 2.6: Ficheros e imágenes



©2025

Micael Gallego, Francisco Gortázar, Michel Maes, Óscar Soto, Iván Chicano

Algunos derechos reservados

Este documento se distribuye bajo la licencia
“Atribución-CompartirIgual 4.0 Internacional”
de Creative Commons Disponible en
<https://creativecommons.org/licenses/by-sa/4.0/deed.es>

Imágenes

- En vez de guardar las **imágenes (o ficheros)** en disco, se pueden **guardar en la BD**
- Eso favorece la **escalabilidad** porque las aplicaciones son ***stateless*** (no tienen estado) y se pueden replicar en **diferentes máquinas**
- Simplifica el despliegue en entornos en los que **no se tiene acceso al disco duro** de forma persistente (Heroku, Kubernetes...)
- Otra opción es usar **servicios de persistencia de ficheros** (S3, Minio...)

- **Atributo Blob (*Binary Large Object*)**

```
@Entity
public class Post {

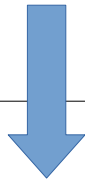
    @Id
    @GeneratedValue(...)
    private long id;

    private String username;
    private String title;
    private String text;
    private String image;

    @Lob
    private Blob imageFile;
    ...
}
```

- Upload Image

```
@PostMapping("/posts/new")  
public String newPost(Model model, Post post, MultipartFile image) throws Exception {  
    postService.save(post, image);  
    return "saved_post";  
}
```



```
public void save(Post post, MultipartFile imageFile) throws IOException{  
    if(!imageFile.isEmpty()) {  
        post.setImageFile(BlobProxy.generateProxy(imageFile.getInputStream(),  
                                                    imageFile.getSize()));  
    }  
    this.save(post);  
}
```

- Download Image

```
@GetMapping("/posts/{id}/image")
public ResponseEntity<Object> downloadImage(@PathVariable long id) throws
SQLException {

    Optional<Post> op = postService.findById(id);

    if (op.isPresent() && op.get().getImageFile() != null) {
        Blob image = op.get().getImageFile();
        Resource file = new InputStreamResource(image.getBinaryStream());

        return ResponseEntity.ok().header(HttpHeaders.CONTENT_TYPE, "image/jpeg")
            .contentTypeLength(image.length()).body(file);
    } else {
        return ResponseEntity.notFound().build();
    }
}
```