HTML, exactamente igual que en el paso anterior. Es decir, el resultado visual es el mismo, pero la estructura interna del código es mucho más mantenible.

```
$entradas = [];
include "datos/arrayobjetos.php";

for($i = 0; $i < count($entradas); $i++) {
    echo $entradas[$i]->damePublicacion();
}
```

4.3.5. Cambios en clases/Publicacion.php

Este archivo no sufre modificaciones en esta etapa. Se sigue utilizando la clase `Publicacion` como clase abstracta, y `Entrada` como subclase con la funcionalidad de representación HTML a través de `damePublicacion`.

4.3.6. Justificación de los cambios

- **Mantenimiento**: editar el contenido del blog ahora solo requiere modificar un archivo externo.
- **Separación de responsabilidades**: `index.php` se encarga del flujo de control y del renderizado, mientras que `arrayobjetos.php` se ocupa únicamente del contenido.
- **Escalabilidad**: permite que el contenido sea sustituido fácilmente por un archivo JSON, una API externa o una base de datos.

4.3.7. Conclusión

Este paso no cambia la funcionalidad del proyecto desde el punto de vista del usuario, pero sí supone un salto cualitativo en cuanto a arquitectura del código. Introducir `include` para separar la lógica del contenido es una práctica fundamental en PHP y en el desarrollo web en general. Este patrón será recurrente a lo largo del proyecto y se irá perfeccionando con