7.B. Composición.

1. Composición.

1.4. Clases anidadas o internas.

En algunos lenguajes, es posible definir una clase dentro de otra clase (**clases internas**):

```
class claseContenedora {

    // Cuerpo de la clase

…

    class claseInterna {

        // Cuerpo de la clase interna

…

    }

}
```

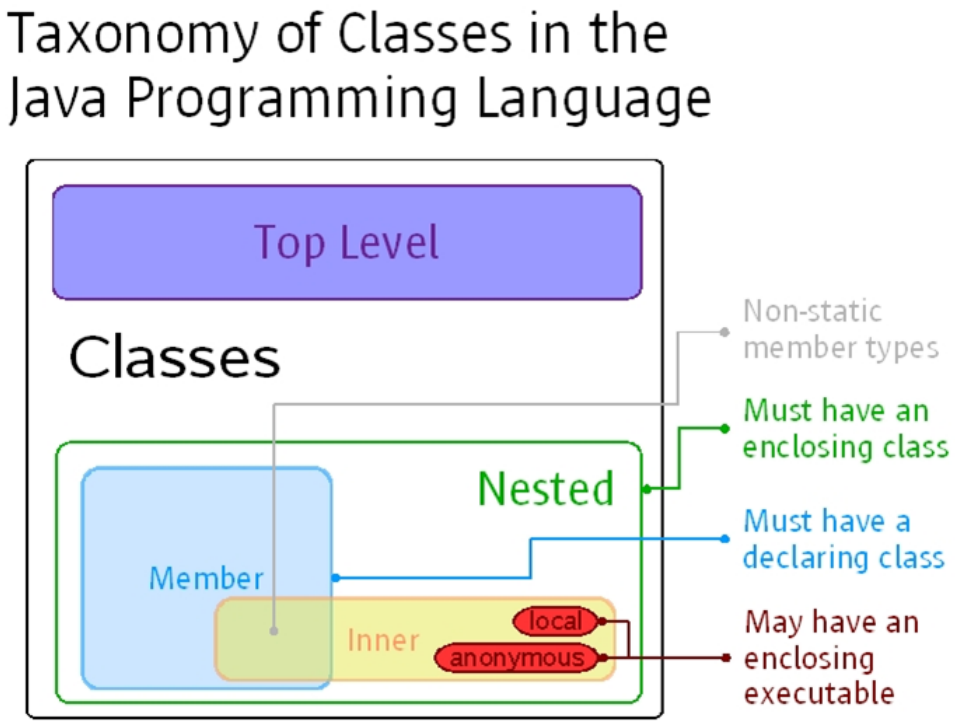

Imagen extraída de curso Programación del MECD.

Se pueden distinguir varios tipos de **clases internas**:

- **Clases internas estáticas** (o **clases anidadas**), declaradas con el modificador `static`.
- **Clases internas miembro**, conocidas habitualmente como **clases internas**. Declaradas al máximo nivel de la clase contenedora y no estáticas.
- **Clases internas locales**, que se declaran en el interior de un bloque de código (normalmente dentro de un método).
- **Clases anónimas**, similares a las internas locales, pero sin nombre (sólo existirá un objeto de ellas y, al no tener nombre, no tendrán constructores). Se suelen usar en la **gestión de eventos** en los **interfaces gráficos**.

Aquí tienes algunos ejemplos:

```
class claseContenedora {

…
```

```
    static class claseAnidadaEstatica {
…
    }
    class claseInterna {
…
    }
```

Las **clases anidadas**, como miembros de una clase que son (miembros de `claseExterna`), pueden ser declaradas con los modificadores `public`, `protected`, `private` o **de paquete**, como el resto de miembros.

Las **clases internas** (no estáticas) tienen acceso a otros miembros de la clase dentro de la que está definida aunque sean privados (se trata en cierto modo de un miembro más de la clase), mientras que las anidadas (estáticas) no.

Las **clases internas** se utilizan en algunos casos para:

- **Agrupar** clases que sólo tiene sentido que existan en el entorno de la clase en la que han sido definidas, de manera que se oculta su existencia al resto del código.
- Incrementar el nivel de **encapsulación** y **ocultamiento**.
- Proporcionar un **código fuente más legible y fácil de mantener** (el código de las **clases internas** y **anidadas** está más cerca de donde es usado).

En Java es posible definir **clases internas** y **anidadas**, permitiendo todas esas posibilidades. Aunque para lo ejemplos con los que vas a trabajar no las vas a necesitar por ahora.