



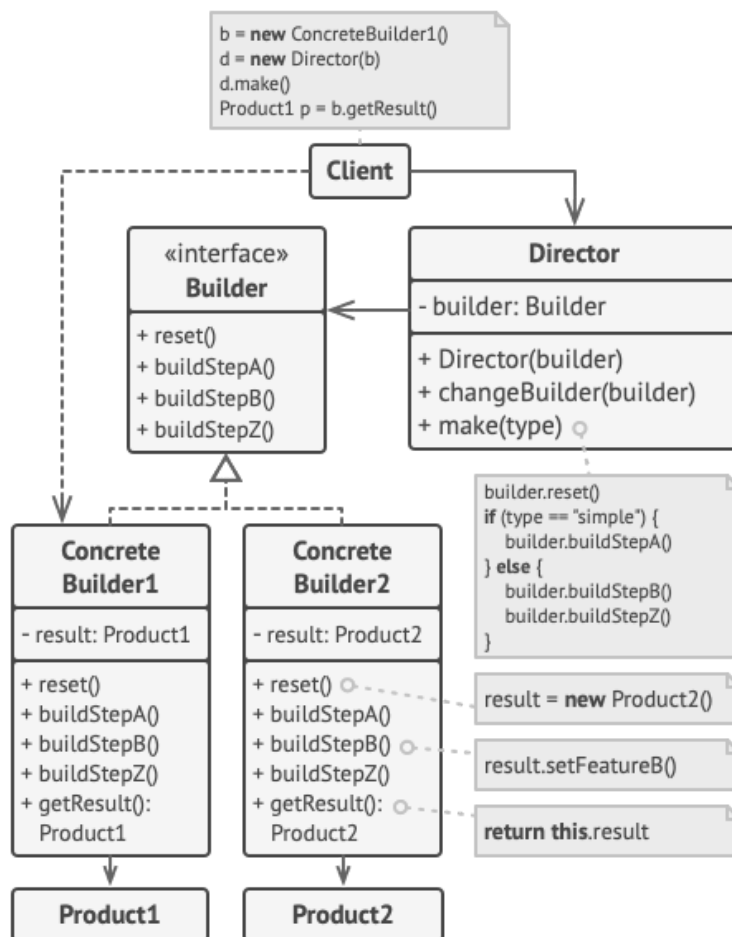
TALLER: PATRONES DE DISEÑO CREAACIONALES

OBJETIVO

- Implementar el patrón de diseño creacional **Builder** para modelar la creación de proyectos en el Sistema de Gestión de Proyectos de Grado.

EL PATRÓN BUILDER

A continuación, recordamos la **estructura** del patrón **Builder**, tomada de la página refactoring.guru.





Descargar el [siguiente ejemplo](#) que implementa el Patrón Builder para crear productos tipo Pizza. Ejecutar y entender el código fuente.

En seguida, aplicar el patrón builder que permita crear distintos tipos de proyectos de grado. A continuación, describimos los pasos.

1. **Definir el Producto:** Crear una clase *Project* que represente un proyecto de grado, con atributos como:

- Título del proyecto de grado
- Modalidad (investigación, práctica profesional, plan coterminal)
- Estudiante 1
- Estudiante 2
- Fecha de creación
- Director del proyecto de grado
- Codirector 1 del proyecto de grado
- Codirector 2 del proyecto de grado
- Objetivo general
- Objetivos específicos

2. **Crear una Clase Abstracta Builder:**

Implementar una clase abstracta *ProjectBuilder*, que contenga métodos para establecer cada uno de los atributos del proyecto aplicando las reglas de negocio:

- Las modalidades de investigación pueden ser de 1 o 2 estudiantes, las demás modalidades solo uno.
- Un proyecto de grado, máximo debe tener 4 objetivos específicos.
- El director de trabajo de grado es obligatorio, los codirectores son opcionales.

3. **Implementar Builders Concretos:**

- Crear tres implementaciones concretas de *ProjectBuilder*, cada tipo de modalidad:
- *ResearchProjectBuilder*: Para modalidad investigación.
- *ProfessionProjectBuilder*: Para prácticas profesionales.
- *CoterminalPlanProjectBuilder*: Plan Coterminal.

4. Definir la Clase *Director*:





- Implementar la clase *ProjectDirector*, encargada de orquestar el proceso de construcción de un proyecto de grado, utilizando un *ProjectBuilder*.

- La clase *ProjectDirector* debe exponer un método *buildProject()* que defina el orden de construcción de los atributos del proyecto.

5. Implementar la Clase Cliente:

- En la clase *Main*, instanciar *ProjectDirector* y utilizar los diferentes *ProjectBuilder* para construir y mostrar distintos tipos de proyectos.

ENTREGABLES

Criterios de Evaluación:

- Implementación correcta del patrón Builder.
- Uso adecuado de clases abstractas y herencia.
- Aplicación del principio de separación de responsabilidades.
- Pruebas unitarias

Se calificará el taller bajo los siguientes rangos:

Excelente (5.0): el taller cumple con todos los criterios de evaluación.

Bueno (4.0): el taller tiene algunos criterios por mejorar.

Regular (3.0): el taller tiene importantes criterios por mejorar.

Deficiente (1 o 2): el taller no cumple con los criterios de evaluación.

Trabajar en grupos de DOS personas. El taller se debe hacer y sustentar al docente en la misma clase.

