**Integrantes:**

**Samuel Felipe Díaz M**

**Santiago Silva**

**La Herramienta Maven**

* **Mayor utilidad:** Maven es una herramienta ampliamente utilizada en el desarrollo de proyectos Java debido a su capacidad para simplificar y automatizar muchas de las tareas esenciales. Entre sus funciones principales, se destaca la compilación del código, la ejecución de pruebas automatizadas y la gestión de dependencias, lo cual resulta especialmente útil en proyectos de gran escala.
* **Fases de Maven:** Maven tiene varias fases en su ciclo de vida, como “validate”, “compile”, “test”, “package”, “verify”, “install”, y ”deploy”. Cada una de estas fases ejecuta una tarea específica en el proceso de construcción del proyecto.
* **Ciclo de vida de la construcción:** El ciclo de vida de Maven abarca todas las fases desde la validación del proyecto hasta su despliegue. Empieza en “validate”, que verifica si el proyecto está bien configurado, y termina en “deploy”, donde el proyecto se sube a un repositorio.
* **Para qué sirven los plugins:** Los plugins en Maven son fundamentales porque añaden funcionalidades a cada fase del ciclo de vida. Por ejemplo, el plugin “compiler” se usa para compilar el código Java, y el plugin “surefire” para ejecutar las pruebas unitarias.
* **Qué es y para qué sirve el repositorio central de Maven:** El repositorio central de Maven es un lugar donde están almacenadas un montón de bibliotecas y dependencias que los proyectos Java pueden usar. Cuando necesitas una dependencia, Maven la descarga automáticamente desde este repositorio.

**Compilar y Ejecutar:**

1. **Busque cuál es el objetivo del parámetro "package" y qué otros parámetros se podrían enviar al comando mvn**

El parámetro “package” en Maven empaqueta el proyecto en un archivo, como un .jar o .war, listo para ser distribuido. Otros parámetros que se pueden usar incluyen “clean“, “install“, y “deploy“, cada uno realizando tareas específicas en el ciclo de vida del proyecto.

1. **Buscar cómo enviar parámetros al plugin "exec"**

Para enviar parámetros al plugin “exec“, se usa el comando “mvn exec:java -Dexec.args="arg1 arg2"“, donde arg1 y arg2 son los parámetros que quieres pasar a la clase principal.

1. **Ejecutar la clase con su nombre y apellido como parámetro. ¿Qué sucedió?**

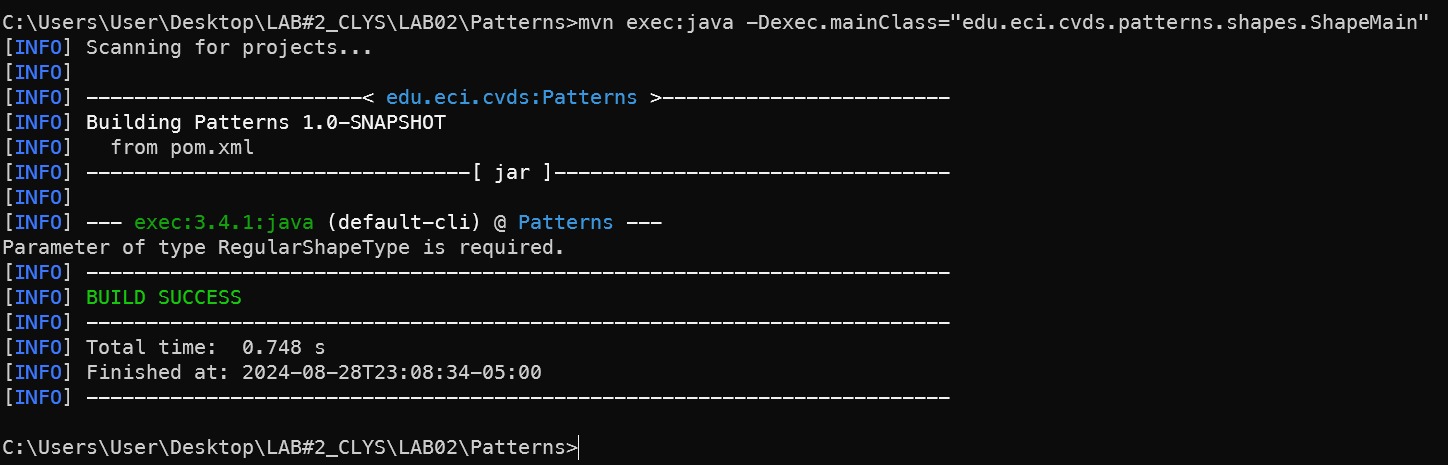
Cuando se realizo este punto, cuando digitamos nombre y apellido no se procesaba correctamente el parámetro. Se añadió un “.split(“,”) en la variable “nombres” para que procesara de la forma compuesta el nombre y apellido.

1. Ejecute múltiples veces la clase ShapeMain, usando el plugin exec de maven con los siguientes parámetros y verifique la salida en consola para cada una:

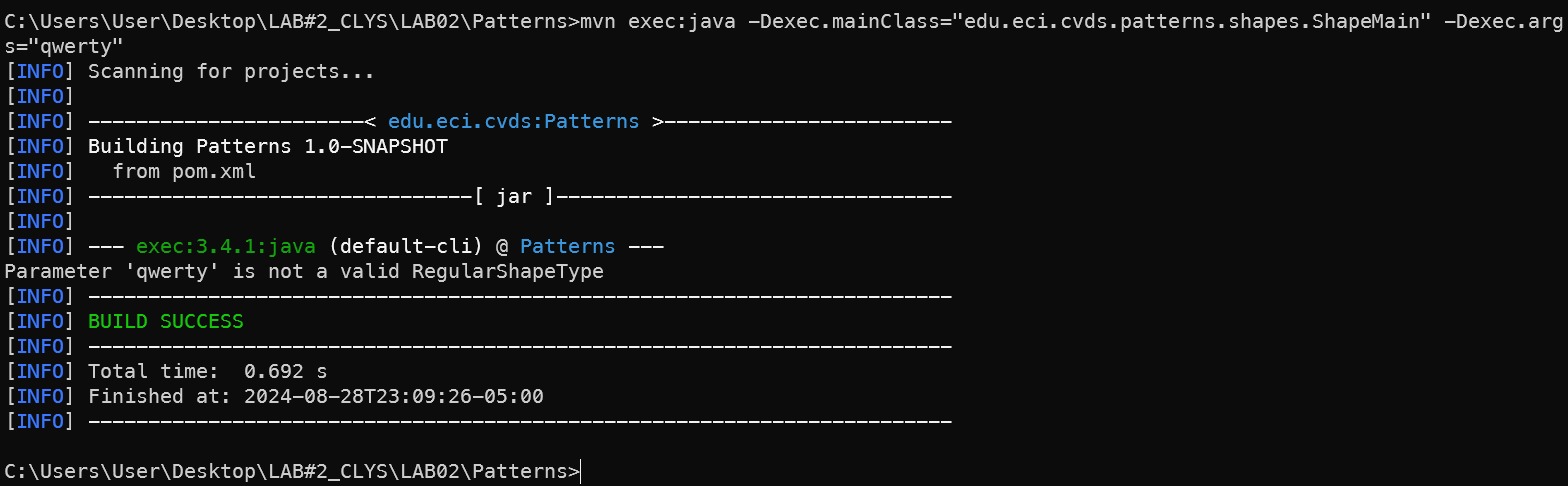
* Sin parámetros
* Parámetro: qwerty
* Parámetro: pentagon
* Parámetro: Hexagon

¿Cuál(es) de las anteriores instrucciones se ejecutan y funcionan correctamente y por qué?

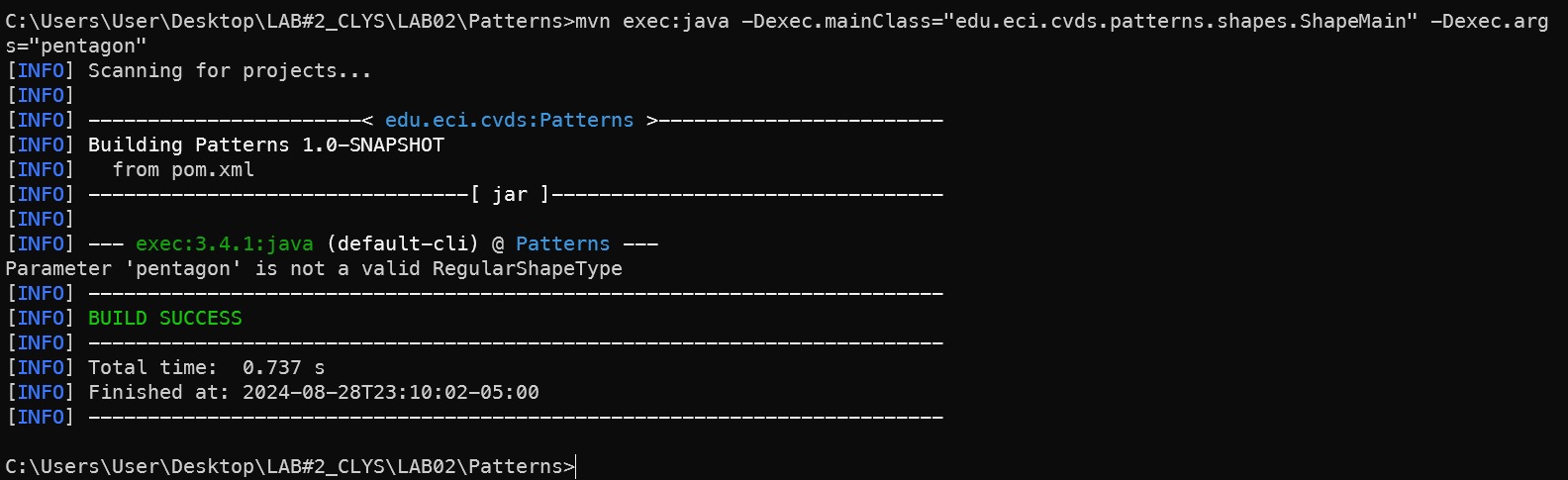
* Sin parámetros: Esta instrucción se ejecuta, pero no funciona correctamente debido a que la función requiere de un parámetro obligatorio.



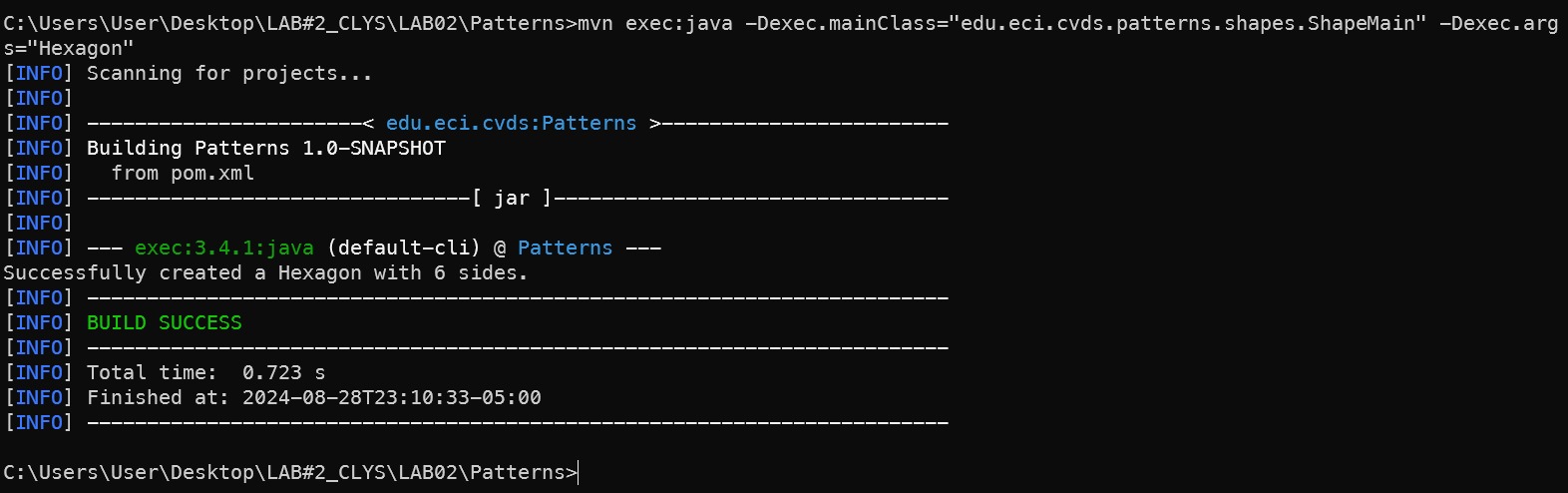
* Parámetro qwerty: Esta instrucción se ejecuta, pero no funciona correctamente porque “qwerty” no es ninguna de las opciones de figuras válidas definidas en el switch.



* Parámetro pentagon: Esta instrucción se ejecuta, pero no funciona correctamente porque el type no está escrito correctamente. En el switch la figura está escrita como “Pentagon” con mayúscula, que corresponde con el nombre de la clase.



* Parámetro Hexagon: Este parámetro se ejecuta correctamente porque es una opción válida de los type, y, por tanto, muestra el mensaje de éxito “Successfully created a Hexagon with 6 sides”.



**Entregar**

**Para qué sirve "gitignore" y cómo se usa:** El archivo. gitignore se usa para decirle a Git qué archivos o carpetas no deben ser rastreados ni versionados. Esto es útil para evitar subir archivos innecesarios como archivos de configuración local o directorios de compilación. Simplemente añades las rutas o nombres de los archivos que quieres ignorar en este archivo.

WEBGRAFÍA

* (2024). “Refactoring Guru”. https://refactoring.guru/design-patterns/factory-method
* https://github.com/CVDS-ESCUELAING/Laboratory2024/blob/main/LABORATORIOS/LAB02.md
* (2024). “Mojohaus”. https://www.mojohaus.org/exec-maven-plugin/usage.html