BRINGING IN SPRING FRAMEWORK TO MAKE JAVA APPLICATION LOOSELY COUPLED

AppGamingBasicJava.java

```java
package com.naveen.learnspringframework;

import com.naveen.learnspringframework.game.GameRunner;
import com.naveen.learnspringframework.game.MarioGame;
import com.naveen.learnspringframework.game.PacmanGame;
import com.naveen.learnspringframework.game.SuperContraGame;

public class AppGamingBasicJava {

    public static void main(String[] args) {

        //var game = new MarioGame();
        //var game = new SuperContraGame();

        var game = new PacmanGame(); //1: Object Creation

        var gameRunner = new GameRunner(game);
            //2: OBJECT CRETION + Wiring of Dependencies
            // Game is a Dependency of GameRunner.
        gameRunner.run();
    }
}
```

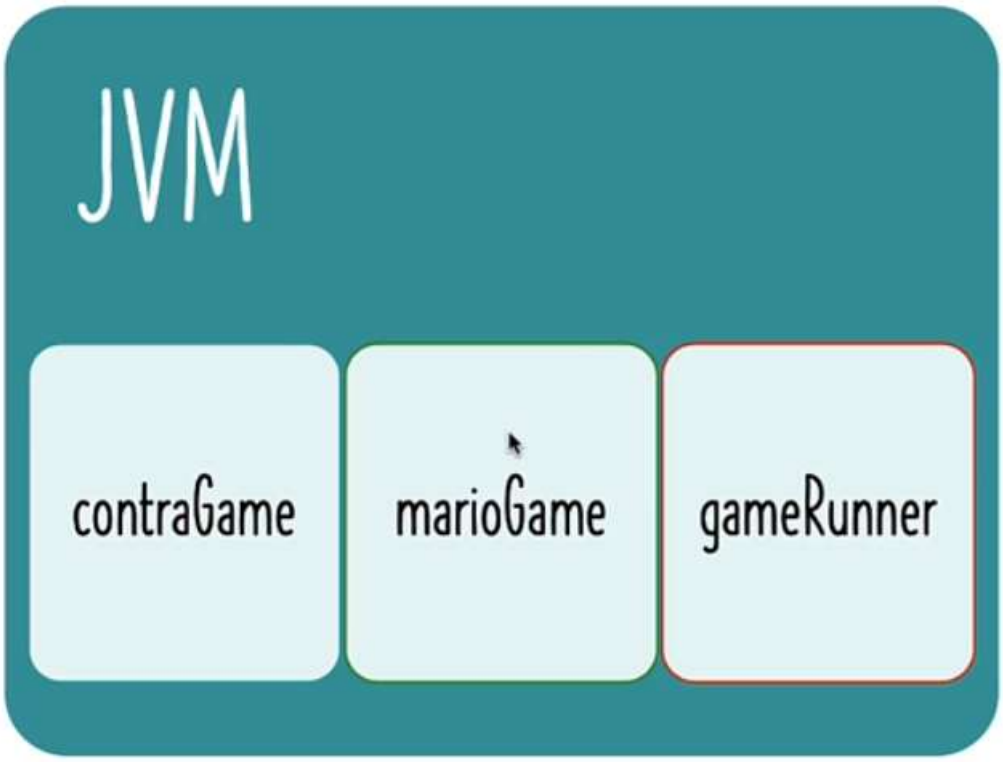
What is Wiring of dependencies?

For a **GameRunner** class to be able to execute a game, it must **possess a game to execute**. Therefore, we can say that the **game** (MarioGame, SuperContraGame or in general GamingConsole) is a **dependency of the GameRunner** class. We are **injecting or wiring** a specific game into the **GameRunner** class.

When we talk about **Enterprise applications**, we may thousands of **classes** and we will have thousands of **dependencies** that are created and thousands of **dependencies** that are **injected** wherever they are needed.

Instead of creating, managing and wiring these objects by ourselves manually, **how about getting the spring framework to do that?**

In this example, the creation of objects is completely **managed by us**. These objects are created in **JVM** (Java Virtual Machine) and the code to create these objects and wiring them is written by us.



We want **Spring framework** to manage these objects and wire them together.