**UNDERSTANDING LOOSE COUPLING AND TIGHT COUPLING**

**AppGamingBasicJava.java**

package com.naveen.learnspringframework;

import com.naveen.learnspringframework.game.GameRunner;

import com.naveen.learnspringframework.game.MarioGame;

public class AppGamingBasicJava {

    public static void main(String[] args) {

        var marioGame = new MarioGame();

        var gameRunner = new GameRunner(marioGame);

        gameRunner.run();

    }

}

In this code, we have created an **instance of the MarioGame** and we are running the **MarioGame** using the **GameRunner**. Let’s say we are creating another game, example **SuperContraGame**.

**SuperContraGame.java**

package com.naveen.learnspringframework.game;

public class SuperContraGame {

    public void up() {

        System.out.println("up");

    }

    public void down() {

        System.out.println("Sit down");

    }

    public void left() {

        System.out.println("Go back");

    }

    public void right() {

        System.out.println("Shoot a bullet");

    }

}

**AppGamingBasicJava.java**

package com.naveen.learnspringframework;

import com.naveen.learnspringframework.game.GameRunner;

import com.naveen.learnspringframework.game.MarioGame;

import com.naveen.learnspringframework.game.SuperContraGame;

public class AppGamingBasicJava {

    public static void main(String[] args) {

        //var marioGame = new MarioGame();

        var superContraGame = new SuperContraGame();

        var gameRunner = new GameRunner(superContraGame);

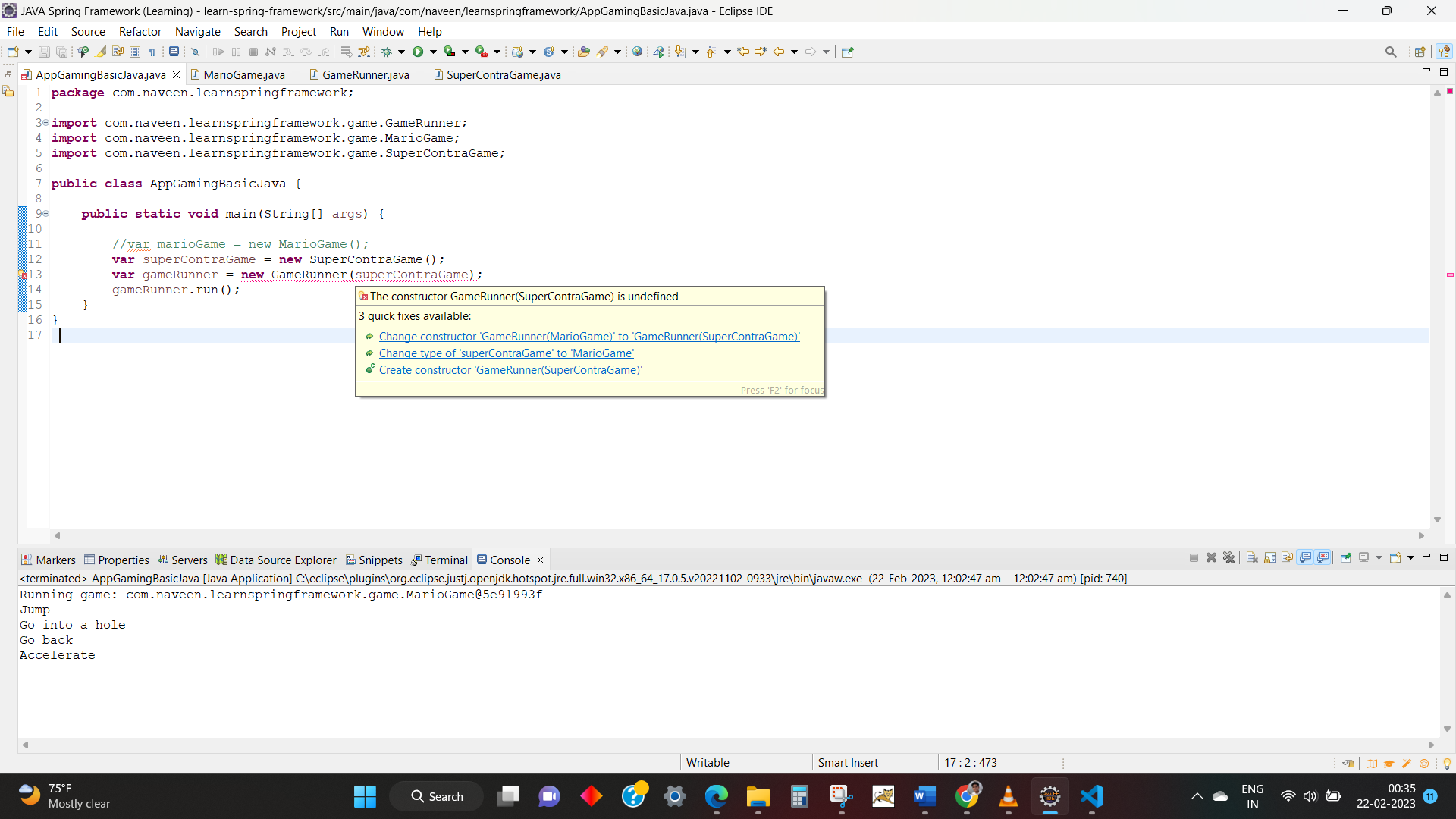
        gameRunner.run();

    }

}

If we try to get run the superContraGame using GameRunner, we will get Compilation error.

**What is the error?**



Because our **GameRunner class is tightly coupled to MarioGame**. If we want to run SuperContraGame using GameRunner class, **we need to change the code**.

**GameRunner.java**

package com.naveen.learnspringframework.game;

public class GameRunner {

    private SuperContraGame game;

    public GameRunner(SuperContraGame game) {

        this.game = game;

    }

    public void run() {

        System.out.println("Running game: " + game);

        game.up();

        game.down();

        game.left();

        game.right();

    }

}

**Text

Description automatically generated with low confidenceOUTPUT:**

**PROBLEM**: Our GameRunner class is **tightly coupled** to a specific game.

**Coupling**: How much work is involved in changing something?

**Example**:

* An engine is **tightly coupled** to a Car.
* A wheel is **loosely coupled** to a Car.
* You can take a laptop anywhere you go.
* A computer, on the other hand, is a little bit more difficult to move.

So, we want,

* **Loose Coupling** as much as possible.
* to make functional changes with **as less code changes** as possible.