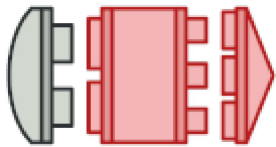




WINTER SALE IS ON!

[Home](#) / [Design Patterns](#) / [Adapter](#) / [Java](#)

Adapter in Java

Adapter is a structural design pattern, which allows incompatible objects to collaborate.

The Adapter acts as a wrapper between two objects. It catches calls for one object and transforms them to format and interface recognizable by the second object.

[Learn more about Adapter →](#)

Navigation

[Intro](#)[Fitting square pegs into round holes](#)[round](#)[RoundHole](#)[RoundPeg](#)[square](#)[SquarePeg](#)[adapters](#)[SquarePegAdapter](#)[Demo](#)[OutputDemo](#)

Complexity: ★☆☆

Popularity: ★★★

**WINTER SALE IS ON!**

based on some legacy code. In such cases, Adapters make legacy code work with modern classes.

There are some standard Adapters in Java core libraries:

- `java.util.Arrays#asList()`
- `java.util.Collections#list()`
- `java.util.Collections#enumeration()`
- `java.io.InputStreamReader(InputStream)` (returns a `Reader` object)
- `java.io.OutputStreamWriter(OutputStream)` (returns a `Writer` object)
- `javax.xml.bind.annotation.adapters.XmlAdapter#marshal()` and `#unmarshal()`

Identification: Adapter is recognizable by a constructor which takes an instance of a different abstract/interface type. When the adapter receives a call to any of its methods, it translates parameters to the appropriate format and then directs the call to one or several methods of the wrapped object.

Fitting square pegs into round holes

This simple example shows how an Adapter can make incompatible objects work together.

round

round/RoundHole.java: Round holes

```
package refactoring_guru.adapter.example.round;

/**
 * RoundHoles are compatible with RoundPegs.
 */
public class RoundHole {
    private double radius;

    public RoundHole(double radius) {
        this.radius = radius;
    }

    public double getRadius() {
```

**WINTER SALE IS ON!**

```

    public boolean fits(RoundPeg peg) {
        boolean result;
        result = (this.getRadius() >= peg.getRadius());
        return result;
    }
}

```

round/RoundPeg.java: Round pegs

```

package refactoring_guru.adapter.example.round;

/**
 * RoundPegs are compatible with RoundHoles.
 */
public class RoundPeg {
    private double radius;

    public RoundPeg() {}

    public RoundPeg(double radius) {
        this.radius = radius;
    }

    public double getRadius() {
        return radius;
    }
}

```

square

square/SquarePeg.java: Square pegs

```

package refactoring_guru.adapter.example.square;

/**
 * SquarePegs are not compatible with RoundHoles (they were implemented by
 * previous development team). But we have to integrate them into our program.
 */
public class SquarePeg {

```



WINTER SALE IS ON!



```

public SquarePeg(double width) {
    this.width = width;
}

public double getWidth() {
    return width;
}

public double getSquare() {
    double result;
    result = Math.pow(this.width, 2);
    return result;
}
}

```

adapters

adapters/SquarePegAdapter.java: Adapter of square pegs to round holes

```

package refactoring_guru.adapter.example.adapters;

import refactoring_guru.adapter.example.round.RoundPeg;
import refactoring_guru.adapter.example.square.SquarePeg;

/**
 * Adapter allows fitting square pegs into round holes.
 */
public class SquarePegAdapter extends RoundPeg {
    private SquarePeg peg;

    public SquarePegAdapter(SquarePeg peg) {
        this.peg = peg;
    }

    @Override
    public double getRadius() {
        double result;
        // Calculate a minimum circle radius, which can fit this peg.
        result = (Math.sqrt(Math.pow((peg.getWidth() / 2), 2) * 2));
        return result;
    }
}

```



WINTER SALE IS ON!



```
package refactoring_guru.adapter.example;

import refactoring_guru.adapter.example.adapters.SquarePegAdapter;
import refactoring_guru.adapter.example.round.RoundHole;
import refactoring_guru.adapter.example.round.RoundPeg;
import refactoring_guru.adapter.example.square.SquarePeg;

/**
 * Somewhere in client code...
 */
public class Demo {
    public static void main(String[] args) {
        // Round fits round, no surprise.
        RoundHole hole = new RoundHole(5);
        RoundPeg rpeg = new RoundPeg(5);
        if (hole.fits(rpeg)) {
            System.out.println("Round peg r5 fits round hole r5.");
        }

        SquarePeg smallSqPeg = new SquarePeg(2);
        SquarePeg largeSqPeg = new SquarePeg(20);
        // hole.fits(smallSqPeg); // Won't compile.

        // Adapter solves the problem.
        SquarePegAdapter smallSqPegAdapter = new SquarePegAdapter(smallSqPeg);
        SquarePegAdapter largeSqPegAdapter = new SquarePegAdapter(largeSqPeg);
        if (hole.fits(smallSqPegAdapter)) {
            System.out.println("Square peg w2 fits round hole r5.");
        }
        if (!hole.fits(largeSqPegAdapter)) {
            System.out.println("Square peg w20 does not fit into round hole r5.");
        }
    }
}
```

OutputDemo.txt: Execution result

```
Round peg r5 fits round hole r5.
Square peg w2 fits round hole r5.
Square peg w20 does not fit into round hole r5.
```



WINTER SALE IS ON!



NEW! NEW!

[← Singleton in Java](#)[Bridge in Java →](#)

Adapter in Other Languages

[Home](#) [Refactoring](#) [Design Patterns](#) [Premium Content](#)
[Forum](#) [Contact us](#)

© 2014-2025 Refactoring.Guru. All rights reserved.

Illustrations by Dmitry Zhart

[Terms & Conditions](#) [Privacy Policy](#)[Content Usage Policy](#) [About us](#)

Ukrainian office:

FOP Olga Skobeleva

Abolmasova 7

Kyiv, Ukraine, 02002

Email:

support@refactoring.guru

Spanish office:

Oleksandr Shvets

Avda Pamplona 64

Pamplona, Spain, 31009

Email:

support@refactoring.guru