Resposta ao item “Padrão para desacoplar código de biblioteca de terceiros”

1. Introdução
Desacoplar seu código de uma biblioteca de terceiros é fundamental para manter flexibilidade e testabilidade: ao isolar todas as c

2. Padrão Adotado: Adapter (ou Facade)
- O que é?
  Um Adapter define uma interface de domínio própria, com apenas os métodos que sua aplicação usa, e implementa essa interfa
- Por que usar?
  1. Isolamento de mudanças: trocar de biblioteca exige alterar apenas o adapter.
  2. Testes mais fáceis: basta mockar a interface, sem carregar a biblioteca real.
  3. Código alinhado ao domínio: sua interface expõe apenas o que importa para seu negócio.

3. Passo a Passo de Implementação
1) Definir a interface de domínio:
```
public interface JsonParser {
    String toJson(Object object);
    <T> T fromJson(String json, Class<T> clazz);
}
```

2) Criar o Adapter que usa a biblioteca (Jackson):
```
public class JacksonJsonParser implements JsonParser {
    private final ObjectMapper mapper = new ObjectMapper();
    @Override
    public String toJson(Object object) {
        try {
            return mapper.writeValueAsString(object);
        } catch (JsonProcessingException e) {
            throw new RuntimeException("Erro ao serializar JSON", e);
        }
    }
    @Override
    public <T> T fromJson(String json, Class<T> clazz) {
        try {
            return mapper.readValue(json, clazz);
        } catch (JsonProcessingException e) {
            throw new RuntimeException("Erro ao desserializar JSON", e);
        }
    }
}
```

3) Injetar e usar no serviço de negócio:
```
public class UserService {
    private final JsonParser parser;
    public UserService(JsonParser parser) {
        this.parser = parser;
    }
    public void process(String incomingJson) {
        User u = parser.fromJson(incomingJson, User.class);
        // lógica de negócio...
        String out = parser.toJson(u);
        // envia resposta...
    }
}
// Configuração:
// JsonParser parser = new JacksonJsonParser();
// UserService service = new UserService(parser);
```

4. Como trocar a biblioteca
- Crie GsonJsonParser implements JsonParser usando new Gson().toJson(...) e fromJson(...).
- Altere apenas a configuração para new GsonJsonParser().
- Nenhuma outra classe precisa mudar.