Classification patterns in tournaments

Juan L. Muñiz Ramos

October 14, 2025

Desing patterns

They are a general and applicable solution to different software design problems. These are templates that identify problems in the system and provide solutions to general problems. These patterns provide a structured, reusable approach to problem-solving. It helps to share efficient and effective solutions. It cannot be expressed in programming language. It is source code oriented.

Type of design pattern to be used:

- Creational patterns

Creational patterns

They provide various object creation mechanisms, where they increase the flexibility and reusability of the code. We are going to use Factory method.
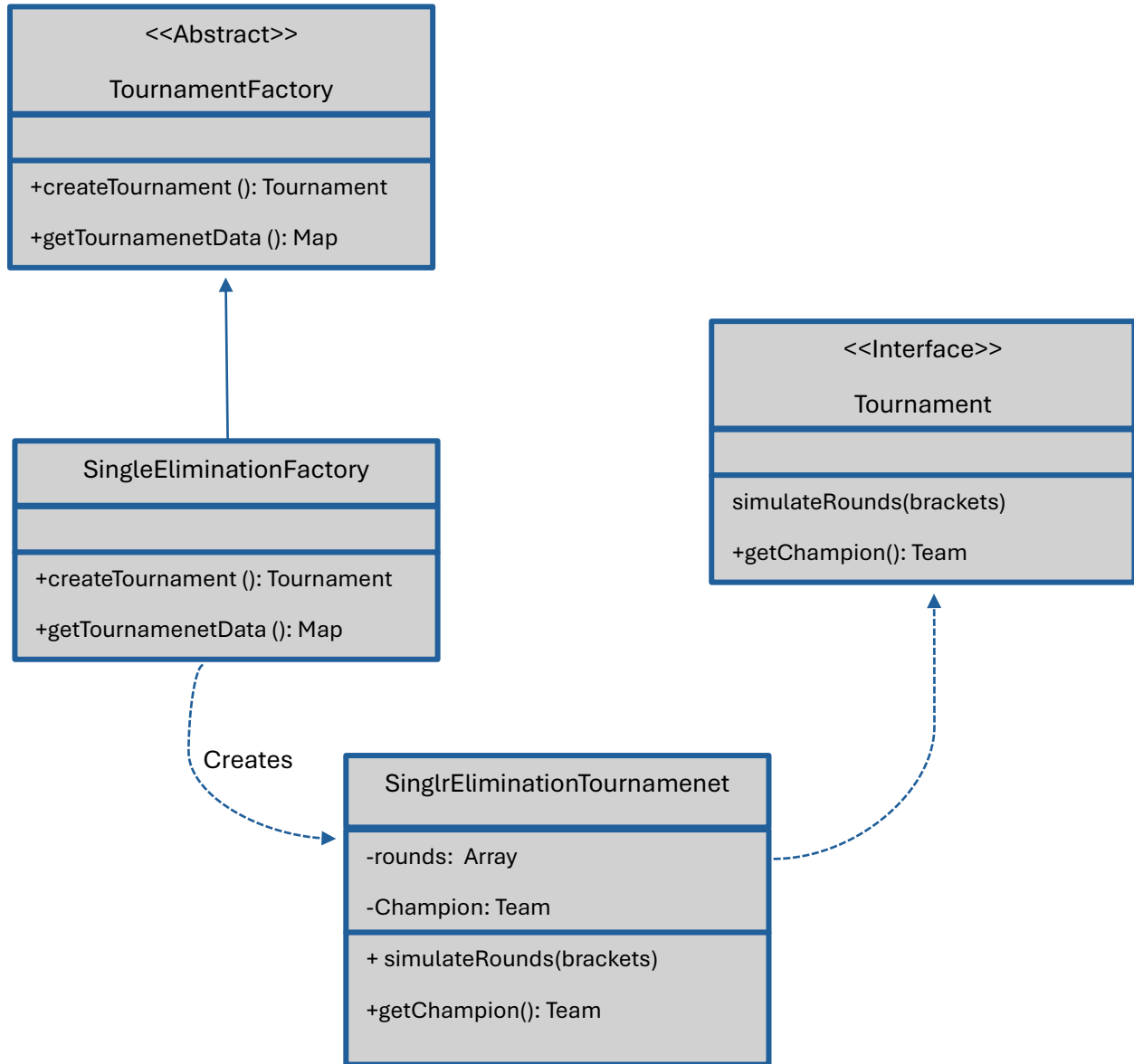
Types of Creational patterns

- Abstract Factory
- Builder Patterns
- Factory Method
- Prototype

Factory Method

An interface for creating objects in a super class, allowing subclasses to alter the type of objects to be created.

Factory method Diagram for tournaments

```
        <<Abstract>>
      TournamentFactory
 ─────────────────────────────

 ─────────────────────────────
 +createTournament (): Tournament

 +getTournamenetData (): Map
```

```
     SingleEliminationFactory
 ─────────────────────────────

 ─────────────────────────────
 +createTournament (): Tournament

 +getTournamenetData (): Map
```

```
        <<Interface>>
         Tournament
 ─────────────────────────────

 ─────────────────────────────
 simulateRounds(brackets)

 +getChampion(): Team
```

Creates

```
    SinglrEliminationTournamenet
 ─────────────────────────────
 -rounds:  Array

 -Champion: Team
 ─────────────────────────────
 + simulateRounds(brackets)

 +getChampion(): Team
```

Referencia:

- Qué son los Patrones de Diseño de software / Design Patterns
- software-design (1).pdf