

## Para saber mais

Como vimos durante a aula, o padrão de projeto `Factory` pode ser bastante útil em situações em que temos objetos que pertencem a classes semelhantes. É importante perceber que os métodos das subclasses da classe `factory` possuem os métodos com os nomes iguais! Dessa forma, podemos usar a `factory` sem se preocupar em qual classe foi instanciada. A imagem abaixo ajuda a ilustrar essa história toda:

[Factory-imagem]

Esta explicação é uma simplificação desse padrão de projeto. Abaixo listei alguns artigos com explicações mais formais.

Artigos em português:

- [Artigo 1 \(https://pt.wikipedia.org/wiki/Factory\\_Method\)](https://pt.wikipedia.org/wiki/Factory_Method)
- [Artigo 2 \(https://www.thiengo.com.br/padrao-de-projeto-factory-method\)](https://www.thiengo.com.br/padrao-de-projeto-factory-method) Artigos em inglês:
- [Artigo 3 \(https://sourcemaking.com/design\\_patterns/factory\\_method\)](https://sourcemaking.com/design_patterns/factory_method) (Site incrível!)
- [Artigo 4 \(https://python-3-patterns-idioms-test.readthedocs.io/en/latest/Factory.html\)](https://python-3-patterns-idioms-test.readthedocs.io/en/latest/Factory.html)

Os artigos estão ordenados do mais simples ao mais complexo. Recomendo a leitura de todos, pois esse padrão de projeto é bem comum no código de diversos sistemas.