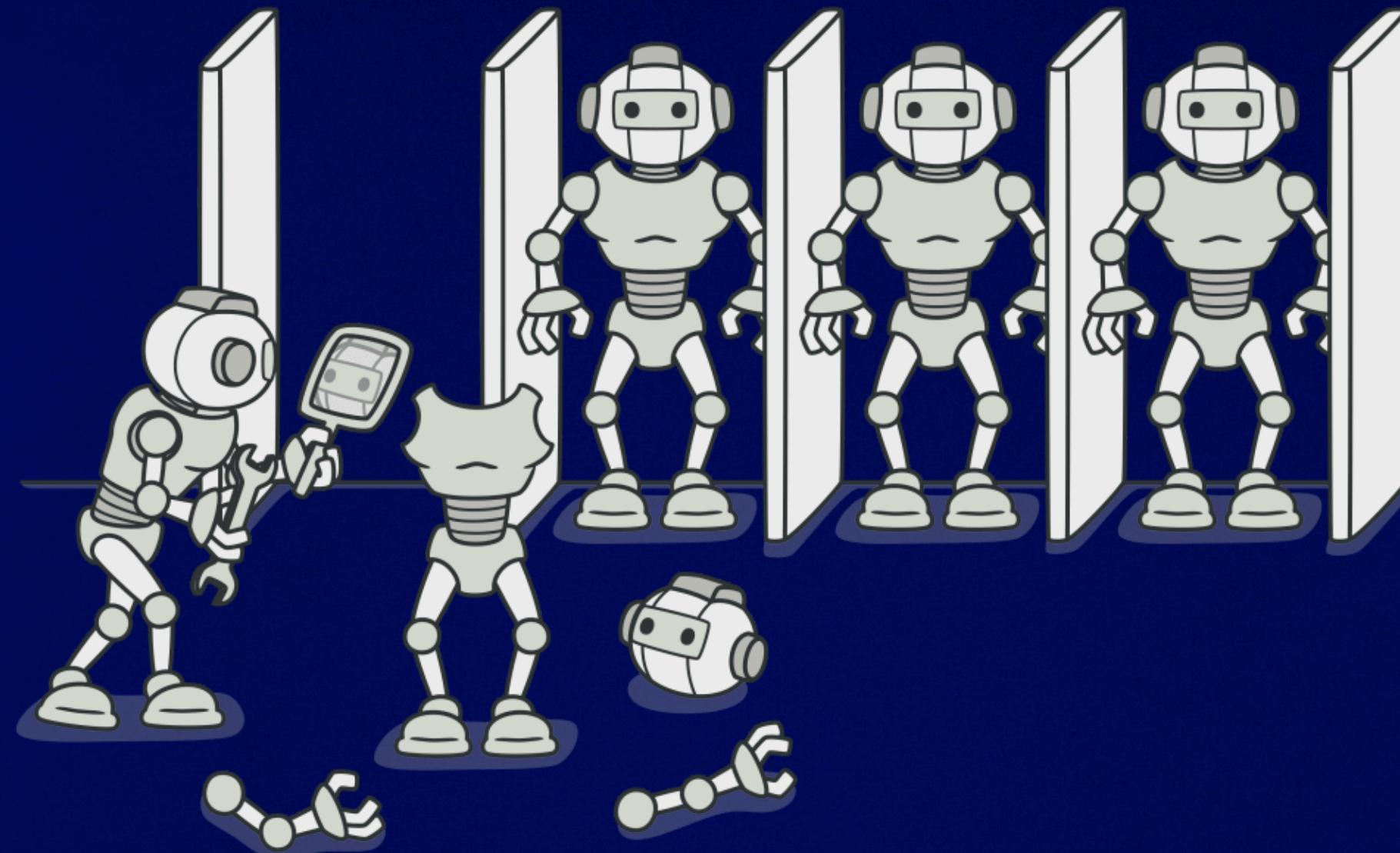


الگو کپی برداری یا (PROTOTYPE PATTERNS)



معرفی

الگوی **PROTOTYPE** می‌گوید: به جای ساخت مستقیم اشیاء از کلاس، نمونه‌های جدید را با کپی (**PROTOTYPE**) یک شیء اولیه (**CLONE**) بسازید. این کار وابستگی به کلاس‌های مشخص را حذف کرده و ایجاد شیء را ساده می‌کند.

مشکل

فرض کنید یک شیء دارید و من خواهید کپی دقیقی از آن بسازید. راه درست چیست؟



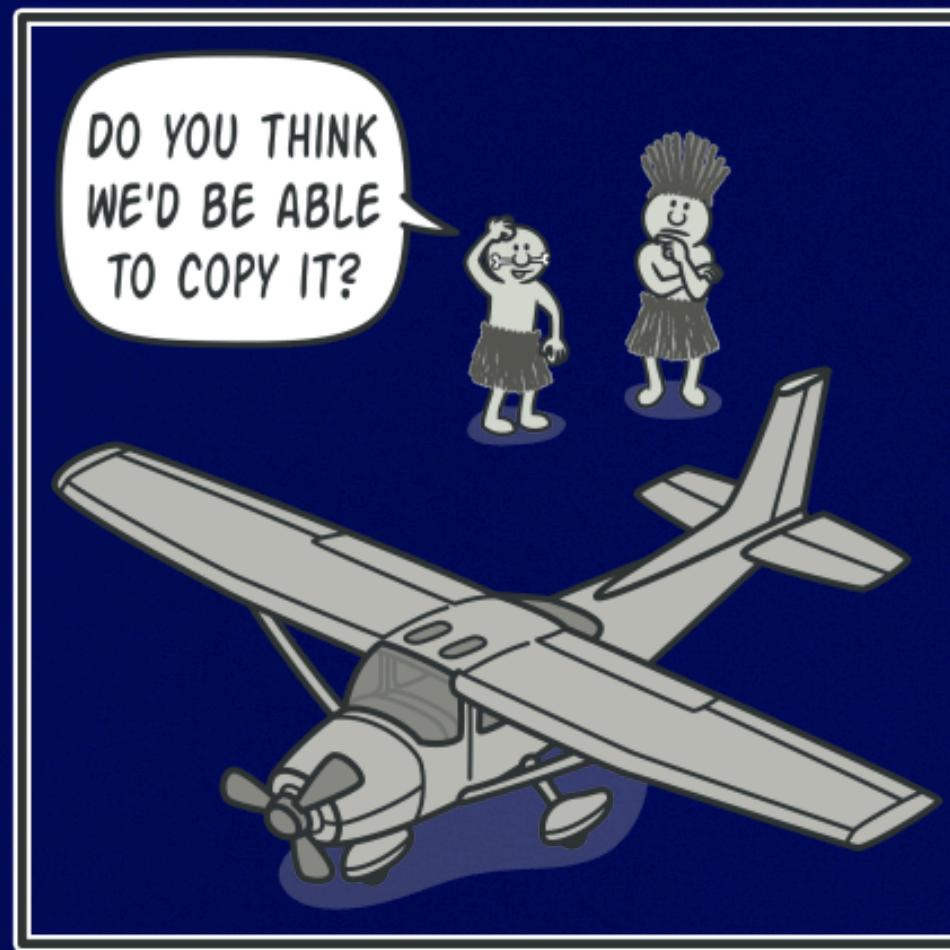
راه حل های اشتباه

ساخت یک شیء جدید از همان کلاس و کپی دستی همه فیلدها.

▼ مشکل:

- وابستگی به کلاس اصلی
- گاهی فقط اینترفیس شیء را می‌شناسیم، نه کلاس واقعی آن

راه حل های اشتباه



راه حل درست

واگذاری فرایند کپی به خود شنء.

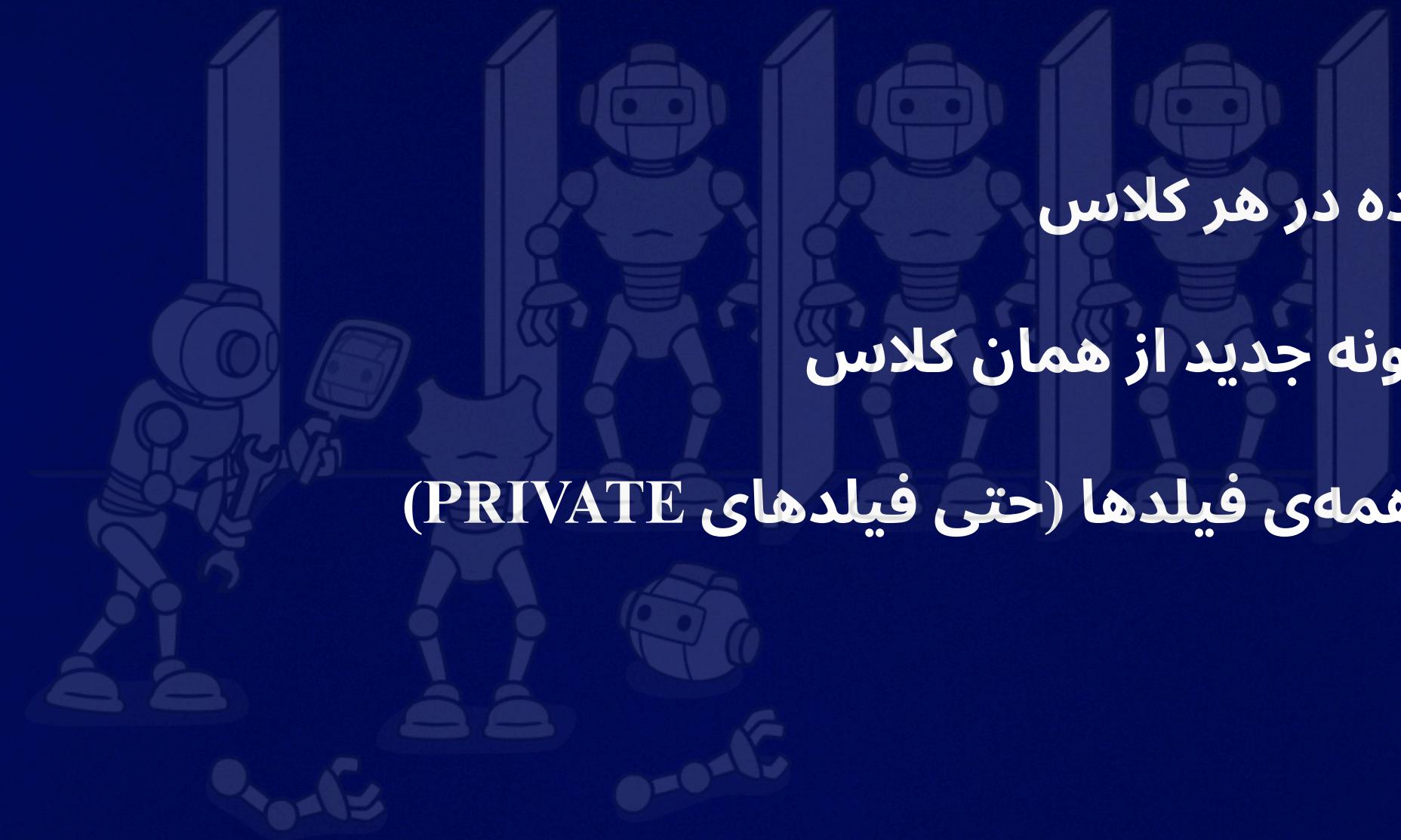
- ◆ تعریف یک اینترفیس مشترک برای همه‌ی کلاس‌ها که شامل متدهای شاملاً می‌شوند.
- ◆ این متدهای کپی را مستقل از کلاس انجام می‌دهند.



راه حل درست

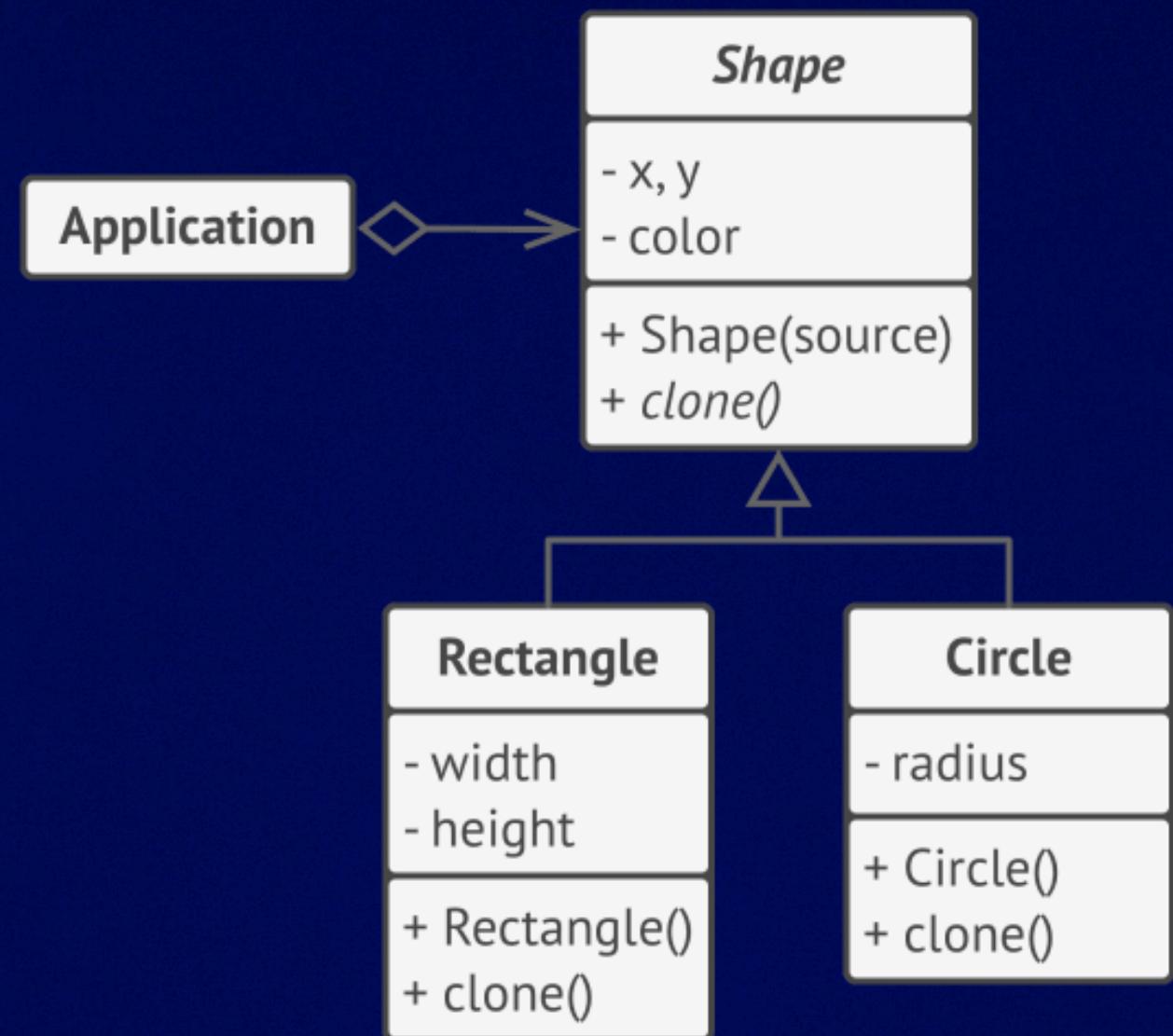


متد CLONE چیست؟



- متدهای ساده در هر کلاس ✓
- ساخت نمونه جدید از همان کلاس ✓
- کپی کردن همه فیلد ها (حتی فیلد های PRIVATE) ✓

ساختار



کی استفاده کنیم

- وقتی نباید به کلاس‌های واقعی وابسته باشید ✓
- وقتی من خواهید تعداد زیرکلاس‌ها را کم کنید ✓
- وقتی تنظیمات پیچیده دارید و من خواهید شئ از پیش آماده را کپی کنید ✓

رابطه الگو PROTOTYPE & ABSTRACT FACTORY

- ♦ معمولاً **FACTORY ABSTRACT FACTORY** روی مجموعه‌ای از **METHOD**ها ساخته می‌شود.
- ♦ به جای تعریف **FACTORY METHOD** برای هر محصول → می‌توان از **PROTOTYPE** برای ساخت محصولات استفاده کرد.
- ♦ انعطاف‌پذیری بیشتر و کد تمیزتر ➔ .

رابطه PROTOTYPE & COMMAND

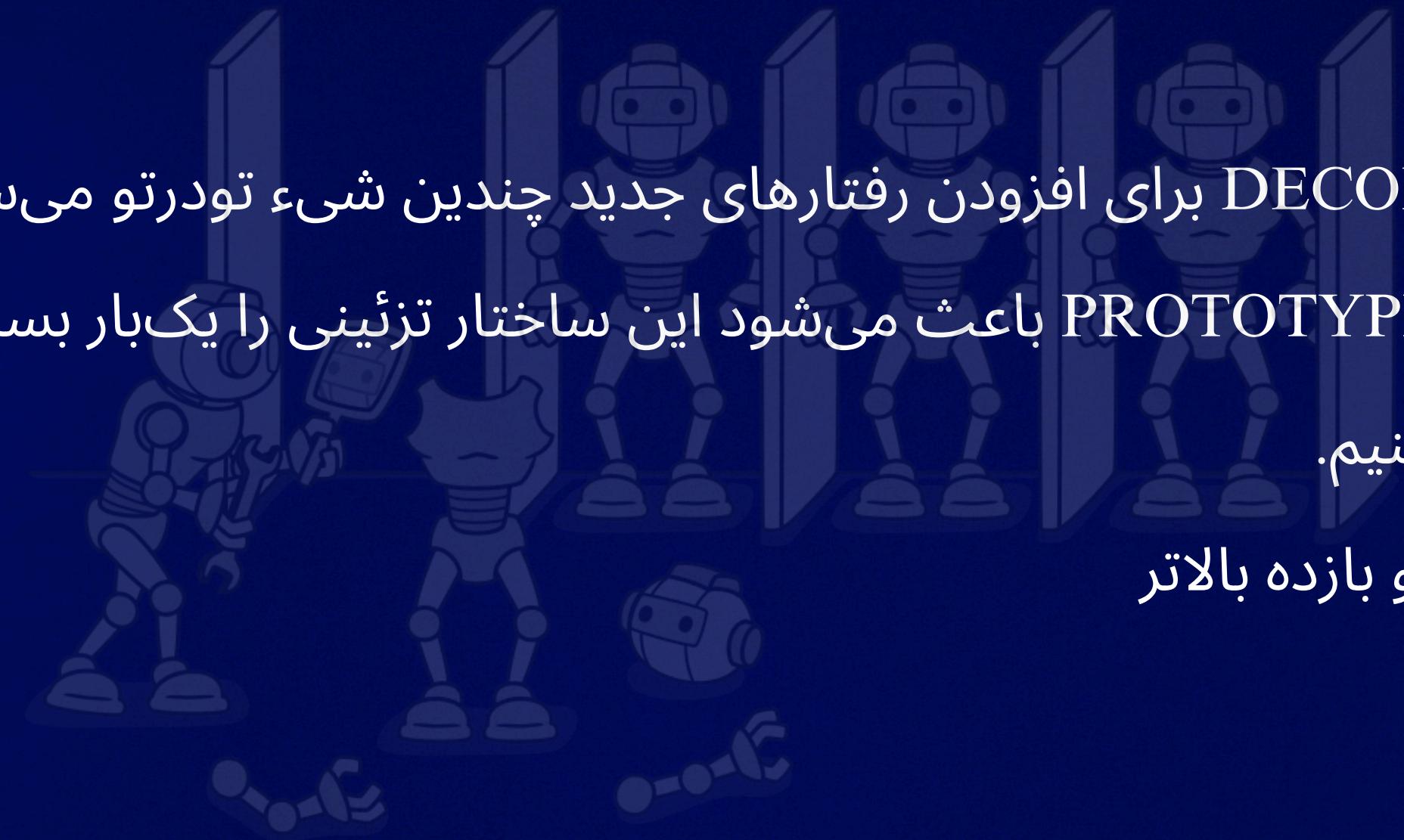
- ♦ در الگوی COMMAND نیاز داریم دستورات گذشته را ذخیره کنیم.
 - ♦ با استفاده از PROTOTYPE می‌توانیم هر COMMAND را به جای بازسازی ، فقط CLONE کنیم.
- نتیجه: مدیریت تاریخچه ساده‌تر و سریع‌تر ➔

رابطه PROTOTYPE & COMPOSITE

- ♦ در ساختار COMPOSITE ممکن است اشیاء تو و پیچیده داشته باشیم.
- ♦ به جای ایجاد مجدد این ساختار از صفر → می توانیم یک PROTOTYPE کامل را CLONE کنیم.
- ♦ صرفه جویی در زمان و کاهش پیچیدگی ➔

رابطه PROTOTYPE & DECORATOR

- ♦ در DECORATOR برای افزودن رفتارهای جدید چندین شیء تودرتو می‌سازیم.
- ♦ استفاده از PROTOTYPE باعث می‌شود این ساختار تزئینی را یک‌بار بسازیم و سپس CLONE کنیم.
- ♦ تکرار کمتر و بازده بالاتر ➔



نکات کلیدی:

هر شیء قابل کپی → 

مناسب برای اشیاء با پیکربندی زیاد 

مبتتنی بر وراثت نیست → مشکلات آن را ندارد 



جمع بندی

= سادهسازی ایجاد اشیاء 

به جای بازنویسی و ایجاد زیرکلاس‌های متعدد، از کپی کردن نمونه‌های آماده استفاده کنید. 