Factory Pattern

Factory Pattern – паттерн создания

Помогает создать объект без использования оператора New, а с помощью вызова функции

Функция конструктор возвращает новый объект

Паттерн фабрика может быть полезным в решении следующих задач:

* Система должна оставаться расширяемой путем добавления объектов новых типов.
* Заранее известно, когда нужно создавать объект, но неизвестен его тип.

|  |
| --- |
| Factory example |
| 1. function Factory() { 2. this.createEmployee = function (type) { 3. var employee; 5. if (type === "fulltime") { 6. employee = new FullTime(); 7. } else if (type === "parttime") { 8. employee = new PartTime(); 9. } else if (type === "temporary") { 10. employee = new Temporary(); 11. } else if (type === "contractor") { 12. employee = new Contractor(); 13. } 15. employee.type = type; 17. employee.say = function () { 18. log.add(this.type + ": rate " + this.hourly + "/hour"); 19. } 21. return employee; 22. } 23. } 25. var FullTime = function () { 26. this.hourly = "$12"; 27. }; 29. var PartTime = function () { 30. this.hourly = "$11"; 31. }; 33. var Temporary = function () { 34. this.hourly = "$10"; 35. }; 37. var Contractor = function () { 38. this.hourly = "$15"; 39. }; |

Для того, чтобы система оставалась независимой от различных типов объектов, паттерн Factory Method использует механизм полиморфизма - классы всех конечных типов наследуют от одного абстрактного базового класса, предназначенного для полиморфного использования. В этом базовом классе определяется единый интерфейс, через который пользователь будет оперировать объектами конечных типов.

Для обеспечения относительно простого добавления в систему новых типов паттерн Factory Method локализует создание объектов конкретных типов в специальном классе-фабрике.

A factory defines an interface for creating an object from a collection of related subclasses.