


## デザインパターンのメリット

 絶版に伴い、校正前の原稿テキストを公開したものです。基本的に原稿そのままをHTML形式に変換したものですので、誤字/脱字、説明不足の箇所もあるかも知れませんがご了承ください。初出:「PHPによるデザインパターン入門」(下岡秀幸/畑勝也/道端良 著, 秀和システム, ISBN4-7980-1516-4, 2006年11月23日発売)

本書で取り扱うGoFのデザインパターンですが、修得し実践していくことでどの様なメリットがあるのでしょうか？

簡単ですが、ここで一度整理しておきます。

### 良い設計の再利用ができる

デザインパターンは、優秀なプログラマたちの実践的なノウハウが凝縮されています。これを利用することで、開発者は、より良い設計により早く到達することができます。

### 再利用性が向上する

よく言われることですが、デザインパターンを使って設計されたクラスは、クラスどうしの関係がゆるくなります(疎結合と言います)。これは、良い設計の重要なポイントの1つです。また、クラスどうしの関係がゆるければゆるいほど、独立性が高いということになります。この結果、再利用性が高くなるのです。

### 保守性が向上する

デザインパターンを使用することで「独立性」が高まりますので、急な仕様変更や機能追加が発生した場合に対象となる影響範囲を狭める結果となります。この結果、保守性を高めることができます。

### 開発者どうしのコミュニケーションが容易になる

前節でも触れましたが、デザインパターンに名前が付けられているのは、他の開発者と「ノウハウ」を共有するためです。つまり、デザインパターンを理解している開発者どうしであれば、具体的なパターン名だけで、相手が何をしようとしているのかを理解できることになります。このため、開発者どうしのコミュニケーションがスムーズになります。たとえば、「ここはFactory Methodパターンでインスタンスを返すようにしよう」とか、「このロジックはStrategyパターンにして切り替えられるようにしよう」といった具合です。

### クラス名からクラスどうしの関係を理解できる

他の開発者が作ったクラスでも、パターンの名前が付いたクラスがあります。こういった場合、「このクラスはこういった役目で、他にこういう関係を持つクラスがあるはずだ」ということが、名前だけで分かるようになります。これは、クラス数の多いライブラリの場合、非常に役に立ちます。

逆に、自分自身で作るクラスの場合でも、クラス名に適切なパターン名を付けることも重要と言えます。

### オブジェクト指向をより深く理解できる

これは筆者が感じたメリットの一つです。

一般的に、デザインパターンを学ぶタイミングは、オブジェクト指向の初歩を学んだ後になると思います。そして、デザインパターンを修得し、オブジェクト指向設計やオブジェクト指向プログラミングを学ぶ、といった流れになると思います。しかし、オブジェクト指向とデザインパターンを同時に学ぶことは大きなメリットがあると思います。デザインパターンには、オブジェクト指向の重要なポイントが凝縮されており、かつ非常に実践的なものです。そのため、早くからデザインパターンに触れておくことで、オブジェクト指向に対する視野が広がります。結果、オブジェクト指向をより深く理解できるようになるのだと思います。