

第14講 フレームワークとMVCアーキテクチャについての基礎知識

フレームワークとは？

- オブジェクト指向における再利用
- フレームワークの定義
- フレームワークの構造
- アプリケーション 主導型 vs フレームワーク主導型
- フレームワークの技術的基礎 (抽象クラス)
- フレームワークの技術的基礎 (オブジェクトの相互作用)
- フレームワークの技術的基礎 (制御の逆転)

オブジェクト指向における再利用

- 抽象度の程度により分類
 - ツールキット (クラスライブラリ)
 - フレームワーク
 - デザインパターン

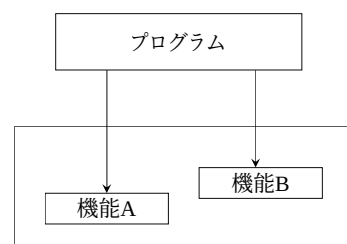
フレームワークの定義

- 抽象クラスの集合とそのインスタンス間の相互作用によって表現された、システム全体または、一部の再利用可能な設計
- 開発者がカスタマイズできる、アプリケーションの骨組み

フレームワークの構造

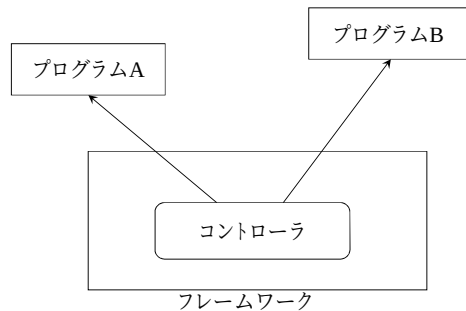
- 通常のシステムと同様に、複数の抽象クラスおよび具体的なクラスで構成
- 通常のシステムとの違い
 - 抽象的なシステムである
 - それ自体ではシステムとして実行できない
 - 差分を定義するために何らかの支援機構の実現

ライブラリの仕組み



ライブラリは、必要に応じてプログラムから呼び出して使う

フレームワークの仕組み



フレームワーク自身に制御の仕組みがあり、必要に応じてフレームワーク自身がプログラムを呼び出す

アプリケーション主導型vs フレームワーク主導型

- アプリケーション主導
 - アプリケーションが必要とする関数やクラスをコントロール
- フレームワーク主導
 - フレームワークがアプリケーションをコントロール
 - フレームワークは、アプリケーションに依存する部分の処理 (hookメソッドに) 依頼する
 - つまり、アプリケーションはアプリケーションに依存する部分 (メソッド) のみを用意すれば、フレームワークが適当な時に処理をおこなう

フレームワークの技術的基礎

- 抽象クラス
 - インスタンスが存在せず、スーパークラスとしてだけ使われる
 - 通常の抽象クラスには、実装されていないオペレーションが少なくとも一つ存在する
 - そのオペレーションはサブクラスが実装する
 - 抽象クラスにはインスタンスがないので、オブジェクトを生成するためのテンプレートとして使われるのではなく、サブクラスを作成するためのテンプレートとして使われる
 - 利用価値の高い抽象クラスの定義 → 適切な抽象度に押さえる

フレームワークの技術的基礎

- オブジェクトの相互作用
 - フレームワークは、プログラムが相互作用を行なうオブジェクトの集合にどのように分解されるのかを記述
 - フレームワークは、抽象クラスの集合に加えて、それらのインスタンスの相互作用の仕組みによって表現
 - インスタンスの相互作用の仕組み
 - オブジェクトの協調モデル
 - オブジェクトの交信パターンとも言い換えることもできる

フレームワークの技術的基礎

- 制御の逆転
 - フレームワークの特徴のひとつ
 - 従来の方法
 - コンポーネントをライブラリからもってきて、そのコンポーネントを呼び出すメインプログラムを書くことによってコンポーネントの再利用をおこなう
 - 開発者がアーキテクチャ全体とプログラムの制御の流れを決定し、その上でコンポーネントをいつ呼び出すのかを決めていた
 - フレームワークによる方法
 - メインプログラムの方が再利用される
 - 開発者はフレームワークにプラグインするものを決める
 - 場合によっては、プラグインするコンポーネントを自ら作ることもある
 - フレームワークがアーキテクチャ全体と制御の流れを決定し、開発者は書いたコードがフレームワークに呼び出される

抽象クラスの再利用

- 抽象クラス
 - インスタンスが存在せず、スーパークラスとしてだけ使われる
 - 通常の抽象クラスには、実装されていないオペレーションが少なくとも一つ存在する
 - そのオペレーションはサブクラスが実装する
- 抽象クラスがあらかじめ用意されていれば、個々の具体的なアプリケーションは、そのサブクラスとして定義する
 - メソッドの上書き
 - 差分プログラミング

抽象化と利用価値

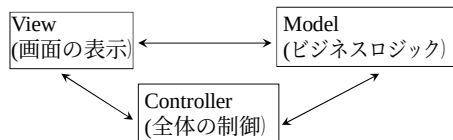
- 一般に、抽象度を上げれば、適用できる範囲は広がる
- その反面、クラスがもつ機能は少なくなる
- つまり、あまり抽象度が高いと、その抽象クラスの機能を継承してみても、再利用できるものが少ない
- 利用価値の高い抽象クラスの定義 → 適切な抽象度に押さえる

フレームワーク

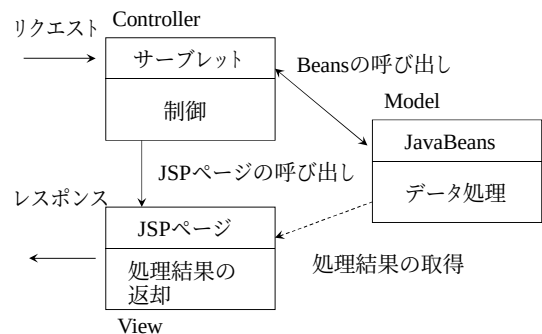
- フレームワーク (アプリケーションフレームワーク)
 - ステージのようにさまざまな目的で使える共通な枠組み
 - 半製品のアプリケーション
 - 細部を欠いている部分があるため、完全なアプリケーションとしては機能しない
 - 欠けている細部を補うようにサブクラスを定義することで、完成したアプリケーションを作成する
 - 細部の補いかたを変えることで、別のアプリケーションが作成できる

MVCモデル (アーキテクチャ) フレームワーク

- MVC (Model-View-Controller)
- 「ビジネスロジック」「表示および入出力」「全体の制御」という形に整理してアプリケーションを作成していく枠組み
- GUIフレームワークとして提案



Webアプリケーション構築のための MVCモデル (アーキテクチャ)



オブジェクト指向による生産性の向上とは？

- フレームワークの利用
 - アプリケーションの基本的な組み立てはフレームワークに任せられる
 - 異なる細部だけを考えればよい
 - その結果、アプリケーション設計の工数の大幅の減少
- 前提
 - 利用できるフレームワークが事前に存在する
- 課題
 - 分野別に共通に利用できるフレームワークを社会的な資産として整備していくこと

Struts

- Webアプリケーションの開発支援用フレームワーク
- Apache Software Foundationにより開発

Webアプリケーション開発の問題点(1)

- 全体を制御する機能がない
 - JSP/サーブレットには、Webアプリケーション全体を制御する機能がない
- GUIとロジックが分離されていない
 - JSP/サーブレットには、クライアントに表示するGUI部分とロジック部分が切り分けられていない

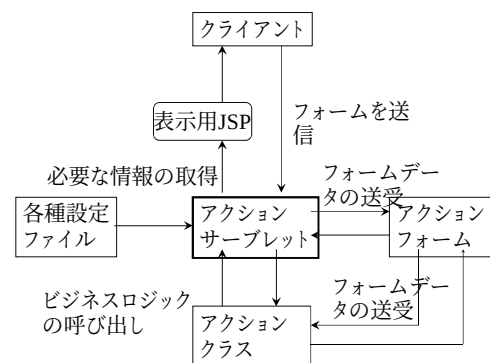
Webアプリケーション開発の問題点(2)

- データの管理機能がない
 - フォームから送信された情報を保持し管理するための機構が、JSP/サーブレットには標準で提供されていない
- 同じことの繰り返しが多い
 - 同じような処理をいくつも作成しなければならない(特に、表示関係)

Strutsの特徴

- MVCアーキテクチャに基づいた設計
- GUIとロジックの分離
- 送信されたフォーム情報の管理が容易
- 入出力情報をチェックするバリデータ機能やBeanの利用や繰り返し機能の提供
- Strutsのバージョン
 - Struts 1.X
 - Struts 2

Strutsの構成



アクションサーブレット、アクションクラス、アクションフォームを中心にそれぞれが必要に応じて呼び出される

Strutsの構成要素(1)

- アクションサーブレット
 - StrutsにおけるWebアプリケーション全体の制御をおこなう。プログラマが作成する必要なし。
- 表示用JSP
 - Strutsでもクライアント側に表示するGUI部をJSPを使って作成。Strutsで用意されている専用のカスタムタグを利用。

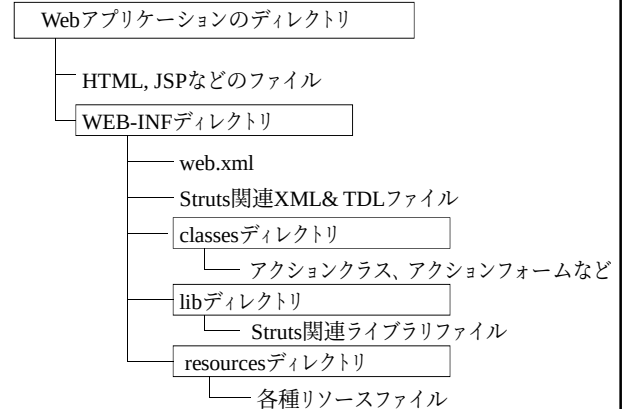
Strutsの構成要素(2)

- アクションフォーム
 - 表示用JSPから送信された情報を管理する。Strutsでは、送信された情報はすべてアクションフォームに保管される。JavaBeansにて表現される。
- アクションクラス
 - フォームが送信された際の処理(ビジネスロジック)を担当。アクションクラスには、実行する処理をまとめたメソッドが定義され、表示用JSPからフォームが送信されると対応するアクションクラス内のメソッド呼び出しがおこなわれる。

Strutsにおける処理の流れ

1. クライアントからフォームがサーバに送信
2. アクションサーブレットが受け取る
3. アクションサーブレットは、送信された情報をアクションフォームに保管する
4. 保管作業が終了したら、アクションクラスで用意されている処理を実行する。アクションクラスからは、必要に応じてアクションフォームの値を取り出したり変更したりする
5. すべての処理が終わったら、送信先のJSPと保存されているアクションフォームの値を元に、クライアント側に送信するHTML文を作成する
6. HTML文をクライアントに送信し、一連の処理を終了する

Strutsのファイル構造



開発者が作成するファイル

- 表示用JSPファイル
 - Strutsで提供されているカスタムタグ<html:○○>を使用
- アクションフォームのクラス
 - org.apache.struts.actionパッケージで用意されている「ActionForm」という抽象クラスを継承して作成。JavaBeansで表現される。
- アクションクラス
 - org.apache.struts.actionパッケージで用意されている「Action」という抽象クラスを継承して作成。アクション実行時に呼び出される「execute」メソッドを実装。
- struts-config.xml(Strutsの設定ファイル)
 - Struts実行時に必要な設定情報が記述。
- web.xml
 - アクションサーブレットを登録。