Javaのフレームワーク

1、**Spring Framework**　（スプリングフレームワーク）

変更に強い

DI(Dependency Injection）（依存性の注入）と呼ばれる仕組みが導入されています。依存性のあるプログラムは外部から取り入れるようになっているので、**1つ1つのプログラムを独立させやすくなります。**

テストが簡単

「Spring MVC Test」という専用のテストプログラムを使うことができ、バグの混入を最小限におさえ、**安全に開発をすすめることができる**。

拡張性が高い

フレームワークの基本的な機能の大部分が**「インターフェース」**として提供されており、**オブジェクト指向の基本原則を徹底**して、Javaを最大限に生かすことができるように設計されている。

保守性が高い

「AOP（Aspect Oriented Programming）」というプログラミングを用いることができます。主に必要となる処理とそうではない処理を分けることで、**コードをわかりやすくする効果**がある。
それによって、何かバグや不具合があった場合にも**原因を特定しやすくなる**。

再利用性が高い

「AOP」の仕組みにより、共通のプログラムをまとめやすくなるので、**再利用することも容易に**なる。
プログラムを再利用することで、プログラムを書く量が減り、修正する場合も最低限の修正で済むようになる。
これは、とくに長期のアプリ開発で**ものすごく効率に差がつく**部分で、再利用性の高いプログラムを、**プログラマーの腕ではなく仕組みで解決している**。

2、Play Framework（プレイフレームワーク）

肥大化したエンタープライズ Java を代替する洗練されたフレームワーク

開発者の生産性に注目し、RESTful アーキテクチャを目指している

Play は アジャイルソフトウェア開発 のための完璧なガイド

痛みの伴わない Java フレームワーク

Play は純粋な Java フレームワークであり、あなた好みのツールやライブラリで開発を続けることができます。
開発プラットフォームとして既に Java を使用しているのならば、別の言語や IDE、ライブラリに切り替える必要はありません。
もっと生産的な Java 開発環境に切り替えるだけでよい。

バグを直してリロード

開発プロセスが効率的になるよう開発サイクルを見直すことで、Java のソースコードを直接コンパイルし、サーバを再起動する必要なしに JVM へ動的にリロードします。ちょうど LAMP や Rails のように、コードを編集してリロードすれば直ちに変更点を見ることができる。

シンプルなステートレス MVC アーキテクチャ

HTTP-コードマッピング

効率的なテンプレートエンジン

強化された JPA

フルスタック・アプリケーションフレームワーク

などが特徴

3、**JSF（JavaServer Faces）**

今後J2EEの世界では標準になるであろうWebアプリケーションのインターフェイスを構築するためのフレームワーク

Webアプリケーションのビジネスロジックとプレゼンテーションデザインを完全に分離することができる

これによりビジネスロジックを構成する開発者とプレゼンテーションデザイナーの役割分担を明確にすることができ、Webアプリケーションの開発効率を上げることができる

最大の特徴は、プレゼンテーションコンポーネント（テキストフィールドやボタン、セレクトボックスなど）をJSPカスタムタグライブラリを使用して表現することができ、さらにそれらのコンポーネントに対して、ステート情報の保持や入力値のチェック（バリデータ）、型の変換（コンバータ）、イベントの制御（イベントハンドラ）、データモデルへのマッピング、といった機能を与えられること

「ステート情報の保持」機能は、各コンポーネントに保存されたアトリビュートを透過的に保持する

インターフェイスを構築するうえで必要な機能をより簡単に実現できる

JSFの記述形式は特別なスクリプトやマークアップ言語を用いるわけではなく、JSPカスタムタグライブラリが用意されている

JSF APIはServlet API上に構築されており、多様なクライアントデバイスに対して出力するために、そのほかのプレゼンテーションテクノロジを併用することも可能