ASP.NET WebForms (VB.NET) スライド教材

イベントドリブン Web アプリケーション開発

初級〜中級 VB.NET 開発者向け

2025年8月14日

目次

この ASP.NET WebForms 学習資料では、基礎から応用まで段階的に学習を進められるよう構成されています。

進捗状況: 2/11

ASP.NET WebForms 概要

ASP.NET WebForms は Microsoft が提供するイベントドリブン型の Web アプリケーション開発フレームワークです。デスクトップアプリケーションのような開発体験を Web 開発で実現します。

主な特徴

- ドラッグ & ドロップによる RAD (高速アプリケーション開発)
- 豊富なサーバーコントロールによる高機能 UI 構築
- イベントドリブンプログラミングモデル
- ViewState による自動状態管理

> **ポイント**
>
> VB.NET と組み合わせることで、Visual Basic 開発者が慣れ親しんだ文法でWebアプリケーションを開発できます。

.NET エコシステムでの位置づけ

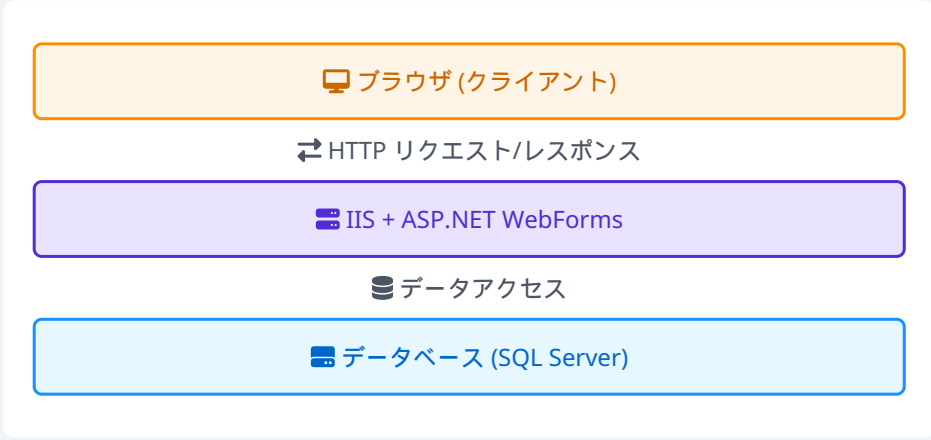
- .NET Framework 上で動作
- IIS (Internet Information Services) でホスティング
- ADO.NET によるデータベース連携

適用シーン

- 社内業務システム
- データ管理・レポート系アプリケーション
- 中小規模の Web アプリケーション

基本アーキテクチャ

クライアント・サーバーモデル



イベントドリブンモデル

- ユーザーアクション (クリック、入力) がサーバーイベントを発生
- PostBack メカニズムによるサーバー処理

ページベース構造

- **.aspx** - マークアップ (HTML + サーバーコントロール)
- **.aspx.vb** - コードビハインド (VB.NET ロジック)
- **web.config** - アプリケーション設定

> ⚠ 重要な概念
>
> **ViewState:** HTTPのステートレス性を補うため、ページの状態情報をクライアントに保存

処理フロー

1. ブラウザからリクエスト
2. ページライフサイクル実行
3. イベント処理・レンダリング
4. HTMLレスポンス送信

開発環境構築

必要ツール

Visual Studio 2019/2022 Community版で十分	必須
.NET Framework 4.7.2+ Visual Studio に含まれる	自動
IIS Express 開発用Webサーバー	自動
SQL Server Express データベース学習用 (オプション)	推奨

セットアップ手順

1. **Visual Studio インストール**
   ASP.NET と Web 開発ワークロードを選択
2. **新規プロジェクト作成**
   ASP.NET Web アプリケーション (.NET Framework) テンプレート選択
3. **Web Forms 選択**
   「Web Forms」テンプレートでプロジェクト作成
4. **動作確認**
   F5 キーでデバッグ実行、ブラウザで表示確認

成功のサイン

ブラウザに "Welcome to ASP.NET!" ページが表示されれば環境構築完了です。

ページライフサイクル

ASP.NET WebForms のページは、リクエストから レスポンスまで決まった順序でイベントが実行されます。

1. **Page_PreInit**
   ページ初期化前、テーマ設定可能
2. **Page_Init**
   コントロール初期化、動的コントロール作成
3. **Page_Load**
   ViewState復元後、最も一般的な処理
4. **Control Events**
   ボタンクリック等のイベント処理
5. **Page_PreRender**
   レンダリング前の最終処理

重要なポイント

IsPostBack プロパティ

初回アクセスとPostBackを区別

開発のコツ

- Page_Load で初期化処理
- 動的コントロールはPage_Init
- Init後 ViewState復元

Webコントロール基礎

ASP.NET WebForms では豊富なサーバーコントロールを使用して、動的な Web UI を構築できます。

主要なコントロール

Label

テキスト表示

```
<asp:Label runat="server" />
```

TextBox

テキスト入力

```
<asp:TextBox runat="server" />
```

Button

クリックイベント

```
<asp:Button runat="server" />
```

DropDownList

選択リスト

```
<asp:DropDownList runat="server" />
```

CheckBox

チェックボックス

```
<asp:CheckBox runat="server" />
```

GridView

データグリッド表示

```
<asp:GridView runat="server" />
```

コントロールの特徴

サーバーコントロール

- runat="server" 属性が必要
- VB.NET からプロパティ操作可能
- イベント処理をサーバーサイドで実行

AutoPostBack

```
<asp:DropDownList
  AutoPostBack="True"
  runat="server" />
```

使い分けのコツ

- Label: 動的テキスト表示
- Literal: HTML そのまま出力
- Repeater: 柔軟なデータ表示
- GridView: 表形式データ編集

状態管理機能

HTTP はステートレスプロトコルですが、ASP.NET WebForms では様々な状態管理機能を提供しています。

ViewState

ページのコントロール状態を自動保存

Hidden フィールドに暗号化して保存、PostBack で復元

Session

ユーザーセッション単位でのデータ保存

サーバーメモリ、セッションタイムアウトで自動削除

Application

アプリケーション全体でのグローバルデータ

全ユーザー共有、アプリケーション再起動まで保持

Cookie

クライアント側での永続データ保存

ブラウザに保存、有効期限設定可能

使い分け比較

スコープと特徴		
保存場所	容量制限	適用場面
ViewState	制限なし	ページ内状態
Session	メモリ依存	ユーザー情報
Application	メモリ依存	設定値
Cookie	4KB	設定保存

実装のコツ

- ViewState 無効化でパフォーマンス向上
- Session は適切なタイムアウト設定
- 機密データは Session が安全
- Cookie はユーザー設定に活用

データバインディング

ASP.NET WebForms では、DataSource コントロールと表示コントロールを組み合わせて、効率的なデータ操作を実現できます。

主要な表示コントロール

GridView

表形式データ表示・編集

ページング、ソート、編集機能内蔵

DetailsView

単一レコード詳細表示

挿入、更新、削除操作対応

CRUD 操作の実装

基本的な流れ

1. DataSource 設定
2. 表示コントロール配置
3. DataSourceID 関連付け
4. イベント処理実装

パラメータ化クエリ

```
SELECT * FROM Users
WHERE Age = @Age
AND City = @City
```

SQLインジェクション対策として必須

開発のコツ

- GridView は自動 CRUD 操作が簡単
- Repeater は表示性能が最適
- ObjectDataSource で N層設計
- キャッシング機能で性能向上

ベストプラクティス

パフォーマンス最適化

- 不要なコントロールで ViewState を無効化
- ページサイズの制限とページング実装
- OutputCache ディレクティブでキャッシング

重要な注意点

⚠ セキュリティ

- 入力値検証は必須
- 機密情報はセッションに保存
- ViewState 暗号化を有効化

高速化のコツ

- 不要な ViewState は無効化
- データバインドは必要最小限
- 静的リソースはキャッシング

ℹ よくある落とし穴

- ViewState の肥大化
- メモリリークの発生

推奨開発フロー

1. Master Page でレイアウト作成
2. 個別ページで機能実装
3. User Control で共通部品化
4. セキュリティ・性能チェック

参考リンク・資料

Microsoft 公式ドキュメント

- ASP.NET Web Forms
  docs.microsoft.com/ja-jp/aspnet/web-forms/
- VB.NET 言語ガイド
  docs.microsoft.com/ja-jp/dotnet/visual-basic/
- Visual Studio ドキュメント
  docs.microsoft.com/ja-jp/visualstudio/

学習リソース

- Microsoft Learn - ASP.NET 学習パス
- Channel 9 - ASP.NET 動画チュートリアル
- ASP.NET サンプルコード集 (GitHub)

よくある質問 (FAQ)

PostBack で画面がちらつく

UpdatePanel を使用した Ajax により部分更新で解決。ScriptManager も必要。

ViewState が大きくなりすぎる

EnableViewState="False" で無効化、または Session を活用して代替。

JavaScript との連携方法

ClientScript.RegisterStartupScript でサーバーからJavaScript実行。

次のステップ

- ASP.NET MVC への移行検討
- ASP.NET Core 学習
- 実践的なプロジェクト挑戦