# Sitecore CMS 6

# データ定義クックブック

CMS 管理者、アーキテクト、開発者向けのヒントとテクニック

# 目次

# Chapter 1

## イントロダクション

この Sitecore データ定義クックブックは、Sitecore ソリューションのデータ インフラストラクチャ コンポーネントを分析および実装する CMS 管理者、アーキテクトおよびデベロッパーに対してヒントとテクニックを提供します。この文書には、データ テンプレート、ブランチ テンプレート、コマンド テンプレート、挿入オプション、プロキシーおよびエイリアスに関する情報が含まれます。[1]

この文書には次の章があります。

- Chapter 1 – イントロダクション
- Chapter 2 – データ テンプレート
- Chapter 3 – ブランチ テンプレート
- Chapter 4 – コマンド テンプレート
- Chapter 5 – 挿入オプション
- Chapter 6 – エイリアスおよびプロキシー

[1] このクックブックで説明されている内容の詳細については、http://sdn.sitecore.net/Reference/ References%20in%20Japanese.aspx にある『データ定義リファレンス ガイド』を参照してください。

# Chapter 2

## データ テンプレート

この章では、データ テンプレートを処理する手順について説明します。これには、テンプレート マネージャーへのアクセス、データ テンプレートのプロパティ表示、データ テンプレートの作成および更新、フィールドおよび標準値の処理の各ステップが含まれます。

この章には次のセクションがあります。

- テンプレート マネージャーへのアクセス方法
- データ テンプレート プロパティの調査
- データ テンプレートの処理
- フィールドの処理
- フィールドの標準値

## 2.1 テンプレート マネージャーへのアクセス方法

テンプレート マネージャーを使用して、データ テンプレートを設定することができます。

テンプレート マネージャーにアクセスするには、Sitecore デスクトップで、[Sitecore] ボタンをクリックして [テンプレート マネージャー] をクリックします。Sitecore デスクトップ内にテンプレート マネージャーが表示されます。

**メモ**
データ テンプレートは、テンプレート マネージャーまたはコンテンツ エディターを使用して設定することができます。これら 2 つのアプリケーションの機能的な違いは、ユーザー インターフェースに表示されるコンテンツ ツリーのルート アイテムです。テンプレート マネージャー内のコンテンツ ツリーのルート アイテムは `/Sitecore/Templates` です。コンテンツ エディター内のコンテンツ ツリーのルート アイテムは `/Sitecore` です。コンテンツ エディター内の `/Sitecore/Templates` アイテムを表示するには、非表示アイテムの表示が必要な場合があります。[2]

[2] 非表示アイテムを表示する方法については、http://sdn.sitecore.net/Reference/ References%20in%20Japanese.aspx にある『クライアント構成クックブック』を参照してください。

## 2.2 データ テンプレート プロパティの調査

このセクションでは、アイテムに関連付けられたデータ テンプレートを確認する手順と、データ テンプレートのプロパティを表示する手順について説明します。

### 2.2.1 アイテムに関連付けられたデータ テンプレートの確認方法

アイテムに関連付けられたデータ テンプレートを確認する方法:

1. コンテンツ エディターでアイテムを選択します。

2. [表示] セクションで、テンプレート プロパティを調査します。

### 2.2.2 データ テンプレートのベース テンプレートの確認方法

データ テンプレートに関連付けられたベース テンプレートを確認する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレート定義アイテムを選択します。テンプレート ビルダーが表示されます。

2. テンプレート ビルダーで、[継承] タブをクリックします。データ テンプレートに関連付けられたベース テンプレートがリストされたレポートが表示されます。これには、これらのテンプレートのセクションおよびフィールドの継承の階層と構造が含まれます。

## 2.3 データ テンプレートの処理

このセクションでは、データ テンプレートを処理する手順について説明します。

### 2.3.1 アイテムに関連付けられたデータ テンプレートの編集方法

アイテムに関連付けられたデータ テンプレートを編集する方法:

1. コンテンツ エディターでアイテムを選択します。
2. [設定] タブをクリックします。
3. [テンプレート] グループにある [編集] をクリックします。選択したアイテムに関連付けられたデータ テンプレートとともにテンプレート マネージャーが表示されます。

### 2.3.2 新しいデータ テンプレートの作成方法

新しいデータ テンプレートを作成する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、`/System/Templates/Template Folder` データ テンプレートを使用して `/Sitecore/Templates` 内に必要な任意のプロジェクト固有のフォルダーを挿入します。
2. 適切なプロジェクト固有のフォルダーに、`/Branches/System/Templates/New Template` ブランチ テンプレートを使用して新しいデータ テンプレート定義アイテムを挿入します。新しいデータ テンプレート ダイアログが表示されます。
3. 新しいデータ テンプレート ダイアログで、新しいデータ テンプレートの名前を [名前] に入力します。
4. ベース テンプレートについては、ベース データ テンプレートを選択するか、デフォルトの `/System/Templates/Standard template` ベース データ テンプレートを選択します。[3]
5. [次へ] をクリックします。

[3] 標準のテンプレートの詳細については、http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『データ定義リファレンス マニュアル』を参照してください。

6. 新しいデータ テンプレートを格納するプロジェクト固有のフォルダーを選択して [次へ] をクリックします。

7. [完了] をクリックします。

### 2.3.3 データ テンプレートのアイコンの設定方法

データ テンプレートのアイコンを設定することにより、データ テンプレートに基づくすべてのアイテムに対して表示されるデフォルト アイコンを制御することができます。[4]

データ テンプレートのアイコンを設定する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレート定義アイテムを選択します。

2. [設定] タブをクリックします。

3. [アピアランス (外観と動作)] グループにある [アイコンを設定] コマンドをクリックします。アイコン選択ダイアログが表示されます。

4. アイコン選択ダイアログでアイコンを選択します。

**メモ**
多くのデータ テンプレート プロパティとは異なり、テンプレートの標準値内ではなく、テンプレート定義アイテム内のテンプレートにアイコンを設定します。

### 2.3.4 データ テンプレートのベース テンプレートの設定方法

データ テンプレートのベース テンプレートを設定する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレート定義アイテムを選択します。テンプレート ビルダーが表示されます。

2. テンプレート ビルダーで、[ビルダー オプション] タブをクリックします。

3. [テンプレート] グループにある [ベース テンプレート] をクリックします。[ベース テンプレート] ダイアログが表示されます。

4. [ベース テンプレート] ダイアログでベース テンプレートを選択して [OK] をクリックします。

[4] アイコンの詳細については、http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『クライアント構成リファレンス マニュアル』および『クライアント構成クックブック』を参照してください。

## テンプレート継承の例: テンプレート階層の設定

次に考慮すべき要件を簡単に示します。

- すべてのコンテンツ アイテムには Title という名前のフィールドが含まれます。
- コンテンツ アイテムの一部の種類には Title、および Body という名前のフィールドが含まれます。
- NewsArticle コンテンツ アイテムには Title、Body、および ReleaseDate という名前の別のフィールドが含まれます。
- Product コンテンツ アイテムには Title、Body、および Department という名前の別のフィールドが含まれます。
- NewsArticle と Product の両コンテンツ アイテムには Image という名前の別のフィールドが含まれます。
- JobDescription コンテンツ アイテムには Title、Body および Department が含まれますが、Image は含まれません。

次の概要は、これらの要件を満たすと想定されるいくつかのデータ テンプレート継承階層の 1 つを示します。特に指定のない限り、すべてのデータ テンプレートは標準データ テンプレートを継承します。

1. Title フィールドが含まれる Page データ テンプレートを作成します。
2. Page から継承し、Body フィールドが含まれる ContentPage データ テンプレートを作成します。
3. Department フィールドが含まれる HasDepartment データ テンプレートを作成します。
4. Image フィールドが含まれる HasImage データ テンプレートを作成します。
5. ContentPage と HasImage の両方から継承する ImageContentPage データ テンプレートを作成します。
6. ImageContentPage から継承し、ReleaseDate フィールドを定義する NewsArticle データ テンプレートを作成します。
7. ImageContentPage と HasDepartment の両方から継承する Product データ テンプレートを作成します。
8. ContentPage と HasDepartment の両方から継承する JobDescription データ テンプレートを作成します。

**メモ**
さまざまなデータ テンプレートでプロパティが異なるフィールドにはテンプレート継承を使用しないでください。たとえば、NewsArticle と Product の Image フィールドにメディア ライブラリ内の異なる場所からイメージを読み取る必要がある場合は、Image フィールドが含まれる共通データ テンプレートから継承するのではなく、データ テンプレートごとに Image フィールドを定義します。

**ヒント**
継承するテンプレートを作成する前にベース テンプレートのセクションおよびフィールドを完全に定義しない場合でも、ベース テンプレートから継承するデータ テンプレートを作成する前にベース テンプレートを作成すると便利ですが、これは必須ではありません。

## 2.3.5 データ テンプレートへのセクションの追加方法

データ テンプレートにデータ テンプレート セクションを追加する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレート定義アイテムを選択します。テンプレート ビルダーが表示されます。

2. テンプレート ビルダーで、[新しいフィールド セクションを追加する] というテキストが含まれる最初のフィールドに新しいデータ テンプレート セクションの名前を入力します。

3. [書き込み] グループにある [保存] をクリックします。

**重要**
標準のテンプレートには、[アドバンスド]、[アピアランス (外観と動作)]、[ヘルプ]、[レイアウト]、[有効期間]、[挿入オプション]、[パブリッシュ]、[セキュリティ]、[統計]、[タスク]、[検証ルール] および [ワークフロー] というセクション名が使用されています。データ テンプレートではこれらのセクション名は使用しないようにしてください。

### データ テンプレート セクションのアイコンの設定方法

コンテンツ エディターに表示されるデータ テンプレート セクションのアイコンを設定する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレート定義アイテムを展開します。

2. コンテンツ ツリーで、データ テンプレート セクション定義アイテムを選択します。

3. [設定] タブをクリックします。

4. [アピアランス (外観と動作)] グループにある [アイコンを設定] コマンドをクリックします。アイコン選択ダイアログが表示されます。

5. アイコン選択ダイアログでアイコンを選択します。

### 2.3.6 データ テンプレートへのフィールドの追加方法

データ テンプレートにデータ テンプレート フィールドを追加する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレート定義アイテムを選択します。テンプレート ビルダーが表示されます。

2. テンプレート ビルダーで、少なくとも 1 つのデータ テンプレート セクションを追加します。

3. テンプレート ビルダーのデータ テンプレート セクションの下で、[新しいフィールドを追加する] というテキストが含まれる最初のフィールドに新しいデータ テンプレート フィールドの名前を入力します。

4. [タイプ] 列で、新しいデータ テンプレート フィールドの種類を選択します。

5. [書き込み] グループにある [保存] をクリックします。

### 2.3.7 異なるデータ テンプレートへのアイテムの関連付け方法

既存のアイテムを異なるデータ テンプレートに関連付けることができます。たとえば、一般的なページをニュース記事に変更するために、アイテムに関連付けられたデータ テンプレートを変更する必要がある場合があります。アイテム内の各フィールド値については、新しいデータ テンプレートに元のデータ テンプレートと同じ ID または名前のフィールドが含まれる場合、アイテムのそのフィールド値が保持されます。

アイテムを異なるデータ テンプレートに関連付ける方法:

1. コンテンツ エディターでアイテムを選択します。

2. [設定] タブをクリックします。

3. [テンプレート] グループにある [テンプレートを変更] をクリックします。[テンプレートを選択してください] ダイアログが表示されます。

4. [テンプレートを選択してください] ダイアログで、アイテムの新しいデータ テンプレートを選択して [次へ] をクリックします。

### 2.3.8 データ テンプレート フィールド値の検証方法

データ テンプレート フィールドに入力されたデータを検証するには、データ テンプレート フィールド定義アイテムに検証ルールを設定します。[5]

データ テンプレート フィールドの検証ルールを設定する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレート フィールド定義アイテムを編集します。
2. [検証ルール] セクション内のフィールドについては、検証ルールを選択します。

### 2.3.9 データ テンプレートに基づくアイテムの検証方法

データ テンプレートに基づくすべてのアイテムに入力されたデータを検証するには、テンプレートの標準値に検証ルールを設定します。

データ テンプレートに基づくすべてのアイテムにデフォルト検証を設定する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレート定義アイテムを編集します。テンプレート ビルダーが表示されます。
2. [ビルダー オプション] タブをクリックします。
3. [テンプレート] グループにある [スタンダード バリュー] をクリックします。これが存在しない場合、データ テンプレート定義アイテムの下に標準値アイテムが挿入されます。
4. 標準テンプレート フィールドを表示します。[6]
5. [検証ルール] セクション内のフィールドについては、検証ルールを選択します。
6. 標準テンプレート フィールドを非表示にします。

[5] 検証の詳細については、http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『クライアント構成クックブック』を参照してください。
[6] 標準テンプレート フィールドを表示または非表示にする方法については、http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『クライアント構成クックブック』を参照してください。

## 2.4 フィールドの処理

このセクションでは、データ テンプレート フィールド プロパティを設定する手順について説明します。

### 2.4.1 フィールド プロパティの設定方法

データ テンプレート フィールドのプロパティを設定する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、フィールドが含まれるデータ テンプレート定義アイテムおよびデータ テンプレート セクション定義アイテムを展開してから、データ テンプレート フィールド定義アイテムを選択します。

2. データ テンプレート フィールド定義アイテムにフィールド プロパティを入力します。

### 2.4.2 選択フィールド内にあるアイテム リストの制御方法

データ テンプレート フィールドのソース プロパティは、ユーザーがゼロ以上のアイテムを選択できるフィールドの選択リスト内に表示されるアイテムを制御する 3 つのテクニックをサポートしています。

**メモ**
次の各セクションで説明するように、データ テンプレート フィールドのソース プロパティを Sitecore クエリまたは Treelist パラメーターに設定する場合、データ テンプレート フィールド定義アイテムのソース フィールドに 1 つまたは複数の破損リンクが含まれることを示すプロンプトが表示される可能性があります。この警告は無視し、データ テンプレート フィールド定義アイテムの変更を保存することができます。

#### ルート アイテム

Droplink、Droplist、Droptree、File、Grouped Droplink、Grouped Droplist、Image、Multilist、Treelist または TreelistEx フィールドのソース プロパティをルート アイテムに対して設定することができます。Checklist、Droplink、Droplist または Multilist などの単一レベル リストの場合、ユーザーはルート アイテムの子を選択することができます。Grouped Droplink および Grouped Droplist などのグループ リストの場合、ユーザーはルート アイテムの孫を選択することができます。Droptree、File、Image、Treelist および TreelistEx などのツリー リストの場合、ユーザーはルート アイテムの子孫を選択することができます。たとえば、`/Sitecore/Content/Home` アイテムの下から選択できるようにするには、次のようにソース プロパティを設定します。

```
        /Sitecore/Content/Home
```

## Sitecore クエリ

`query`: 接頭辞を使用して Checklist、Droplink、Droplist、Grouped Droplist、Grouped Droplink または Multilist フィールドのソース プロパティを Sitecore クエリに設定することができます。[7] たとえば、Section という名前のデータ テンプレートに基づくホーム アイテムの子を選択できるようにする方法は、次のとおりです。

```
query:/sitecore/content/home/*[@@templatename="Section"]
```

## Treelist パラメーター

Treelist または TreelistEx フィールドのソース プロパティを、URL エスケープした複数の key=value パラメーター (アンパサンド文字 ("`&`") で区切る) に設定することができます。たとえば、Section という名前のデータ テンプレートまたは Page という名前のデータ テンプレートに基づくホーム アイテムの子孫を選択できるようにする方法は、次のとおりです。

```
DataSource=/Sitecore/Content/Home&IncludeTemplatesForSelection=section,page
```

次のパラメーターを指定することができます。

- **`DataSource`**: ルート アイテムです。詳細については、前のセクション「ルート アイテム」を参照してください。
- **`DatabaseName`**: `DataSource` が含まれるデータベースの名前です。
- **`IncludeTemplatesForSelection`**: ユーザーはこのカンマ区切りのデータ テンプレート名のリストに基づくアイテムのみを選択することができます。
- **`ExcludeTemplatesForSelection`**: ユーザーはこのカンマ区切りのデータ テンプレート名のリストに基づくアイテムを選択することはできません。
- **`IncludeTemplatesForDisplay`**: ユーザーはこのカンマ区切りのデータ テンプレート名および ID のリストに基づくアイテムを移動することができます。
- **`ExcludeTemplatesForDisplay`**: ユーザーはこのカンマ区切りのデータ テンプレート名および ID のリストに基づくアイテムを移動することはできません。
- **`IncludeItemsForDisplay`**: ユーザーはこのカンマ区切りのアイテム名および ID のリストに基づくアイテムを移動することができます。

[7] Sitecore クエリの詳細については、http://sdn.sitecore.net/Reference/Using%20Sitecore%20Query.aspx、および http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『データ定義リファレンス マニュアル』を参照してください。

- **ExcludeItemsForDisplay**: ユーザーはこのカンマ区切りのアイテム名および ID のリストに基づくアイテムを移動することはできません。

- **AllowMultipleSelection**: `yes` である場合、ユーザーは同じアイテムを複数回選択することができます。

### 2.4.3 File または Image フィールドのルート アイテムの指定方法

File または Image のデータ テンプレート フィールドのルート アイテムを指定するには、セクション「ルート アイテム」で説明されているように、フィールド定義アイテムにソース プロパティを設定します。たとえば、Image フィールドのユーザーが `/Sitecore/Media Library/Images` の子孫を選択できるようにするには、次のように、フィールド定義アイテムにソース プロパティを設定します。

```
/Sitecore/Media Library/Images
```

### 2.4.4 Image フィールドのデフォルト フォルダーの指定方法

Image のデータ テンプレート フィールドのコンテンツ ツリーにデフォルトの場所を指定するには、データ テンプレート フィールド定義アイテム内のソース プロパティを、チルダ文字 ("~") から始まるパスに設定します。選択リストはこの場所にデフォルト設定されますが、ユーザーはメディア ライブラリのルートに移動することができます。たとえば、Image フィールドのセレクターの場所を `/Sitecore/Media Library/Images` にデフォルト設定するには、次のように、フィールド定義アイテムにソース プロパティを設定します。

```
~/Sitecore/Media Library/Images
```

**メモ**
File または Image のデータ テンプレート フィールドに対してルート アイテムとデフォルト フォルダーの両方を指定することはできません。

### 2.4.5 リッチ テキスト エディターのフィールドで使用可能な機能の設定方法

リッチ テキスト エディター (RTE) のプロファイルは、リッチ テキスト エディターのフィールドで使用可能な機能を制御します。[8] RTE フィールド定義アイテムのソース プロパティを Core データベース内の `/Sitecore/System/Settings/Html Editor Profiles` 内の RTE プロファイル定義アイテムのパスに設定します。

[8] リッチ テキスト エディターのプロファイルの詳細については、http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『クライアント構成リファレンス マニュアル』および『クライアント構成クックブック』を参照してください。

## 2.5 フィールドの標準値

このセクションでは、フィールドの標準値を処理する手順について説明します。

### 2.5.1 フィールドの標準値の設定方法

データ テンプレートのフィールドに標準値を設定する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレート定義アイテムを選択します。テンプレート ビルダーが表示されます。
2. [ビルダー オプション] タブをクリックします。
3. [テンプレート] グループにある [スタンダード バリュー] をクリックします。これが存在しない場合、データ テンプレート定義アイテムの下に標準値アイテムが挿入されます。
4. 標準値アイテムにフィールドの標準値を入力し、リボンを使用して、挿入オプション、レイアウト詳細および初期ワークフローなど、標準テンプレートのフィールドにストアされているプロパティに標準値を適用します。

### 2.5.2 標準レイアウト詳細の割り当て方法

データ テンプレートに基づくすべてのアイテムにデフォルト レイアウト詳細を割り当てる方法:[9]

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレートの標準値を設定します。
2. [プレゼンテーション] タブをクリックします。
3. [レイアウト] グループにある、[デザイン] または [詳細] をクリックします。
4. レイアウト詳細を割り当てます。

[9] レイアウト詳細については、http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『プレゼンテーション コンポーネント リファレンス ガイド』を参照してください。

### 2.5.3 標準挿入オプションの割り当て方法

挿入オプションの詳細については、Chapter 5 の「挿入オプション」を参照してください。

データ テンプレートに基づくすべてのアイテムにデフォルト挿入オプションを割り当てる方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレートの標準値を設定します。
2. [設定] タブをクリックします。
3. [挿入オプション] グループにある [割り当て] をクリックします。
4. 挿入オプションを割り当てます。

### 2.5.4 デフォルト ワークフローの割り当て方法

データ テンプレートに基づくすべてのアイテムにデフォルト ワークフローを割り当てる方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、データ テンプレートの標準値を設定します。
2. [レビュー] タブをクリックします。
3. [ワークフロー] グループにある [編集] をクリックします。
4. データ テンプレートに基づくすべてのアイテムのデフォルト ワークフローを選択します。

### 2.5.5 フィールドの標準値へのリセット方法

アイテム内のフィールドを標準値にリセットする方法:

1. コンテンツ エディターでアイテムを選択します。
2. フィールドが標準テンプレートに定義されている場合、標準テンプレート フィールドを表示します。
3. [バージョン] タブをクリックします。
4. [フィールド] グループにある [リセット] をクリックします。[フィールドをリセット] ダイアログが表示され、左側にアイテム フィールド値、右側に標準値が表示されます。

5. [フィールドをリセット] ダイアログで、標準値にリセットするフィールドのチェックボックスを選択して [リセット] をクリックします。選択したフィールドが標準値にリセットされます。

6. 標準テンプレート フィールドが表示されている場合、標準テンプレート フィールドを非表示にします。

### 2.5.6 レイアウト詳細の標準値へのリセット方法

Sitecore には、レイアウト詳細を標準値にリセットするためのショートカットがあります。

レイアウト詳細を標準値にリセットする方法:

1. コンテンツ エディターでアイテムを選択します。

2. [プレゼンテーション] タブをクリックします。

3. [レイアウト] グループにある [リセット] をクリックし、プロンプトを確認します。アイテムのレイアウト詳細が、テンプレートの標準値に定義されているレイアウト詳細にリセットされます。

### 2.5.7 挿入オプションの標準値へのリセット方法

Sitecore には、挿入オプションを標準値にリセットするためのショートカットがあります。

挿入オプションを標準値にリセットする方法:

1. コンテンツ エディターでアイテムを選択します。

2. [設定] タブをクリックします。

3. [挿入オプション] グループにある [リセット] をクリックし、プロンプトを確認します。アイテムの挿入オプションが、テンプレートの標準値に定義されている挿入オプションにリセットされます。

### 2.5.8 .NET API を使用したフィールドの標準値へのリセット方法

`Sitecore.Data.Fields.Field.Reset()` メソッドを使用して、フィールドを標準値にリセットすることができます。たとえば、コンテキスト アイテムのレイアウト詳細を標準値にリセットするには、次のようなコードを使用します。

```
        Sitecore.Data.Items.Item item = Sitecore.Context.Item;
        Sitecore.Data.Fields.Field field = item.Fields[Sitecore.FieldIDs.LayoutField];
        item.Editing.BeginEdit();
        field.Reset();
        item.Editing.EndEdit();
```

# Chapter 3

## ブランチ テンプレート

この章では、ユーザーが新しいアイテムを挿入する際にレプリケート可能なアイテム構造のプロトタイプを定義する、ブランチ テンプレートを設定する手順について説明します。この章には、新しいブランチ テンプレートを作成する手順、既存のアイテムを複製して新しいブランチ テンプレートを作成する手順、ブランチ テンプレートを使用して挿入されたアイテムのアクセス権を設定する手順、さまざまなブランチ テンプレートを使用可能な CMS ユーザーを制御するアクセス権を設定する手順が含まれます。

この章には次のセクションがあります。

- ブランチ テンプレートの概要
- ブランチ テンプレートの作成方法
- データ テンプレートのブランチ テンプレートの作成方法
- 既存のアイテムの複製によるブランチ テンプレートの作成方法
- ブランチ テンプレートのアイコンの設定方法
- ブランチ テンプレートを使用して挿入されたアイテムの明示的なアクセス権の設定方法
- ブランチ テンプレートを使用して挿入可能なアカウントの制御方法

## 3.1 ブランチ テンプレートの概要

ブランチ テンプレートは、ブランチ テンプレート定義アイテム (単一アイテムを格納可能)、アイテムの階層、またはアイテムの複数の階層で構成されています。ユーザーがブランチ テンプレートを起動すると、ブランチ テンプレート定義アイテムの下にあるアイテム (任意のフィールド値を含む) が複製された後、次のセクション「ブランチ テンプレートの作成方法」で説明されているように、アイテム名およびフィールド値に対してトークン置換が実行されます。Chapter 5 の「挿入オプション」で説明されているように、ブランチ テンプレートとともにデータ テンプレートおよびコマンド テンプレートを割り当てることができます。

**メモ**
次の各セクションで説明されているように、`/System/Branches/Branch` データ テンプレートまたは `/Branches/System/Branch/New Branch` コマンド テンプレートを使用して新しいブランチ テンプレート定義アイテムを挿入することができます。コマンド テンプレートを使用すると、指定したデータ テンプレートの名前をとって命名されるブランチ テンプレート定義アイテムが作成され、ブランチ テンプレート定義アイテム内のこのデータ テンプレートに基づく `$name` という名前のアイテムが作成されます。

ブランチ テンプレート定義アイテム内の各アイテムのフィールドに次のトークンが含まれる値を入力することができます。これらのトークンは、ブランチ テンプレートを使用して挿入された各アイテムのフィールド値内で展開されます。

- **$name**: アイテムの挿入時にユーザーによって入力された名前
- **$id**: アイテムの ID です。**$parentid**: アイテムの親の ID
- **$parentname**: アイテムの親の名前
- **$date**: システム日付 (`yyyyMMdd`)
- **$time**: システム時間 (`HHmmss`)
- **$now**: 日付および時間 (`yyyyMMddTHHmmss`)

**メモ**
すべてのデータ テンプレートに対してブランチ テンプレートを作成する必要はありません。ブランチ テンプレートを使用するのは、複数のアイテムを挿入する場合、挿入されたアイテムにアクセス権を割り当てる場合、またはブランチ テンプレートを使用してアイテムを挿入可能なアカウントを制御する場合です。

**メモ**

ブランチ テンプレート定義アイテム内には任意の数のアイテムを作成でき、各アイテムには任意の数の子孫アイテムを設定することができます。ユーザーがブランチ テンプレートを使用してアイテムを作成すると、トークンを展開する前にブランチ テンプレート定義アイテムのすべての子孫が複製されます。

**重要**

各データ テンプレートの単一標準値アイテム内にストアされ、フィールドに値がないこのテンプレートに基づくすべてのアイテムに適用される標準値とは異なり、ブランチ テンプレート定義アイテム内のフィールド値は、このブランチ テンプレートを使用して作成された各アイテムに複製されます。データが複製されないようにするには、ブランチ テンプレート内のフィールド値ではなく、標準値を使用します。

## 3.2 ブランチ テンプレートの作成方法

新しいブランチ テンプレートを作成する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、`/System/Branches/Branch Folder` データ テンプレートを使用して `/Sitecore/Tempaltes/Branches` 内に必要な任意のプロジェクト固有のフォルダーを挿入します。

2. 適切なプロジェクト固有のフォルダーに、`/System/Branches/Branch` データ テンプレートを使用して新しいブランチ テンプレート定義アイテムを挿入します。

3. ブランチ テンプレート定義アイテム内に 1 つまたは複数のアイテムまたはアイテムの階層を挿入します。

## 3.3 データ テンプレートのブランチ テンプレートの作成方法

特定のデータ テンプレートに基づく単一アイテムが含まれる新しいブランチ テンプレートを挿入することができます。

データ テンプレートの新しいブランチ テンプレートを作成する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、`/System/Branches/Branch Folder` データ テンプレートを使用して `/Sitecore/Tempaltes/Branches` 内に必要な任意のプロジェクト固有のフォルダーを挿入します。

2. 適切なプロジェクト固有のフォルダーに、`/Branches/System/Branch/New Branch` コマンド テンプレートを使用して新しいブランチ テンプレート定義アイテムを挿入します。[新しいブランチを作成します。] ダイアログが表示されます。

3. [新しいブランチを作成します。] ダイアログでデータ テンプレートを選択して [作成] を選択します。選択したデータ テンプレートに基づく `$name` という名前の単一アイテムが含まれる新しいブランチ テンプレート定義アイテムが作成されます。

**メモ**
デフォルトでは、ブランチ フォルダーの挿入オプションには `/Branches/System/Branch/New Branch` コマンド テンプレートは含まれますが、 `/System/Branches/Branch` データ テンプレートは含まれません。

## 3.4 既存のアイテムの複製によるブランチ テンプレートの作成方法

1 つまたは複数の既存のアイテムまたはアイテムのブランチを複製することにより、ブランチ テンプレートを作成することができます。

1 つまたは複数の既存のアイテムを複製してブランチ テンプレートを作成する方法:

1. コンテンツ エディターまたはテンプレート マネージャーで、`/System/Branches/Branch` データ テンプレートを使用してブランチ テンプレート定義アイテムを挿入します。

2. コンテンツ エディターで、非表示アイテムを表示します。[10]

3. 複製して新しいブランチ テンプレートに投入するアイテムを選択します。

4. [ホーム] タブをクリックします。

5. [操作] グループにある [指定の場所にコピー] をクリックします。[次の場所にアイテムをコピー] ダイアログが表示されます。

6. [次の場所にアイテムをコピー] ダイアログで、ブランチ テンプレート定義アイテムを選択して [コピー] を選択します。アイテムまたはアイテムの階層がコピーされ、ブランチ テンプレート定義アイテムの下に新しいアイテムまたはアイテムの階層が作成されます。

7. 必要な場合、非表示アイテムを非表示にします。

[10] 非表示アイテムを表示する方法については、http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『クライアント構成クックブック』を参照してください。

## 3.5 ブランチ テンプレートのアイコンの設定方法

ブランチ テンプレートに作成されたアイテムには、そのデータ テンプレートに関連付けられたアイコンが表示されます。ブランチ テンプレートのアイコンを設定することにより、挿入オプションでこのブランチ テンプレートに表示されるアイコンを制御することができます。

ブランチ テンプレートのアイコンを設定する方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、ブランチ テンプレート定義アイテムを選択します。
2. [設定] タブをクリックします。
3. [アピアランス (外観と動作)] グループにある [アイコンを設定] コマンドをクリックします。アイコン選択ダイアログが表示されます。
4. アイコン選択ダイアログでアイコンを選択します。

## 3.6 ブランチ テンプレートを使用して挿入されたアイテムの明示的なアクセス権の設定方法

デフォルトでは、ブランチ テンプレートを使用してアイテムを挿入すると、これらのアイテムはその親からアクセス権を継承します。アクセス権を継承させる代わりに、ブランチ テンプレート内のアイテムにアクセス権を設定することもできます。

ブランチ テンプレートを使用して挿入されたアイテムのアクセス権を設定するには、ブランチ テンプレート定義アイテム内のアイテムにアクセス権を設定します。これらのアクセス権はブランチ テンプレートを使用して挿入されたアイテムにコピーされます。

## 3.7 ブランチ テンプレートを使用して挿入可能なアカウントの制御方法

`item:create` アクセス権を持つすべてのユーザーは、挿入オプションを介して使用可能な任意のブランチ テンプレートを使用してアイテムを挿入することができます。ブランチ テンプレート定義アイテムにアクセス権を設定することにより、特定のブランチ テンプレートを使用してアイテムを挿入可能なユーザーを制限することができます。

ブランチ テンプレートを使用してアイテムを挿入可能なユーザーを制御するには、ブランチ テンプレート定義アイテムにアクセス権を設定します。ブランチ テンプレート定義アイテムに対する `item:read` アクセス権がないユーザーは、このブランチ テンプレートを使用してアイテムを挿入することはできません。

# Chapter 4

# コマンド テンプレート

この章では、1 つまたは複数の新しいアイテムを挿入する際に一般に使用されるロジックが含まれるコマンド テンプレートを構成するための手順について説明します。ユーザー インターフェースを表示しないコマンド テンプレート、JavaScript プロンプトを使用してアイテム名を収集するコマンド テンプレート、ASP.NET または Sitecore Sheer ユーザー インターフェースを表示するコマンド テンプレートを作成することができます。挿入オプションを使用してコマンド テンプレートとともにデータ テンプレートおよびブランチ テンプレートを割り当てることができます。

この章には次のセクションがあります。

- コマンド テンプレートの作成方法
- コマンド テンプレートの例: 親アイテムからの挿入オプションのコピー
- コマンド テンプレートの例: ユーザー インターフェースの表示

## 4.1 コマンド テンプレートの作成方法

コマンド テンプレートを作成する方法:

1. `Sitecore.Shell.Framework.Commands.Command` から継承するクラスを作成し、`Execute()` メソッドを上書きします。次のコード サンプルを使用することができます。

```
namespace Namespace.Shell.Framework.Commands
{
  public class ClassName : Sitecore.Shell.Framework.Commands.Command
  {
    public override void Execute(Sitecore.Shell.Framework.Commands.CommandContext
      context)
    {
    }
  }
}
```

2. `name` 属性内のコマンド コードを `type` 属性内のクラス シグネチャーにマップするファイル `/App_Config/Commands.config` に `/configuration/command` エレメントを追加します。次はその例です。

```
<command name="namespace:category:command"
type="Namespace.Shell.Framework.Commands.ClassName,Assembly" />
```

3. テンプレート マネージャーまたはコンテンツ エディターで、`/Templates/Common/Folder` データ テンプレートを使用して `/Sitecore/Templates` 内に必要な任意のプロジェクト固有のフォルダーを挿入します。

4. `/Sitecore/Templates` 内の適切なプロジェクト固有のフォルダーに、`/System/Branches/Command Template` データ テンプレートを使用してコマンド テンプレート定義アイテムを挿入します。

5. コマンド テンプレート定義アイテムで、[Data] セクション内の [Command] フィールドにコマンド コードを入力します。次はその例です。

```
namespace:category:command(id=$ParentID)
```

**メモ**

パラメーター `id=$ParentID` は、コマンドに渡される ID を、ユーザーによる新しいアイテムの挿入先のアイテムに設定します。これにより、ユーザーがアイテムを選択したがコマンド テンプレートを起動する前に別のアイテムを右クリックした際に、正しいアイテム ID がコマンド テンプレートに渡されるようになります。このパラメーターがなければ、ユーザーが右クリックしたアイテムではなく、選択したアイテムの ID が渡されます。

6. 挿入オプションを他のアイテムに割り当てて新しいコマンド テンプレートを使用可能にする方法については、「データ テンプレートまたは個々のアイテムへの挿入オプションの割り当て方法」のセクションを参照してください。

## 4.2 コマンド テンプレートの例: 親アイテムからの挿入オプションのコピー

ユーザーがフォルダーを作成する際に、親アイテムから新しいフォルダーに対して挿入オプションがコピーされる必要がある要件を考慮してください。

次のコマンド テンプレート コードは、この要件を満たしています。

```
namespace Namespace.Shell.Framework.Commands
{
  public class NewAssignedFolder : Sitecore.Shell.Framework.Commands.Command
  {
    public override void Execute(Sitecore.Shell.Framework.Commands.CommandContext
      context)
    {
      if (context.Items.Length == 1)
      {
        Sitecore.Data.Items.Item item = context.Items[0];
        System.Collections.Specialized.NameValueCollection parameters =
          new System.Collections.Specialized.NameValueCollection();
        parameters["id"] = item.ID.ToString();
        parameters["language"] = item.Language.ToString();
        parameters["database"] = item.Database.Name;
        Sitecore.Context.ClientPage.Start(this, "Run", parameters);
      }
    }

    protected void Run(Sitecore.Web.UI.Sheer.ClientPipelineArgs args)
    {
      if (args.IsPostBack)
      {
        if ((!String.IsNullOrEmpty(args.Result)) && args.Result != "undefined")
        {
          Sitecore.Data.Database db = Sitecore.Configuration.Factory.GetDatabase(
            args.Parameters["database"]);
          Sitecore.Data.Items.Item parent = db.GetItem(args.Parameters["id"],
            Sitecore.Globalization.Language.Parse(args.Parameters["language"]));
          Sitecore.Data.Items.TemplateItem template =
            Sitecore.Context.ContentDatabase.Templates[Sitecore.TemplateIDs.Folder];
          Sitecore.Data.Items.Item item = parent.Add(args.Result, template);

          if (!(String.IsNullOrEmpty(parent[Sitecore.FieldIDs.Branches])
            && String.IsNullOrEmpty(parent["__Insert Rules"])))
          {
            item.Editing.BeginEdit();

            if (!String.IsNullOrEmpty(parent[Sitecore.FieldIDs.Branches]))
            {
              item.Fields[Sitecore.FieldIDs.Branches].Value =
                parent[Sitecore.FieldIDs.Branches];
            }

            if (!String.IsNullOrEmpty(item["  Insert Rules"]))
            {
              item.Fields["__Insert Rules"].Value = parent["__Insert Rules"];
            }

            item.Editing.EndEdit();
```

```
                }

                Sitecore.Context.ClientPage.SendMessage(this,
                  "item:load(id=" + item.ID.ToString() + ")");
              }
            }
            else
            {
              Sitecore.Context.ClientPage.ClientResponse.Input(
                "Enter the name of the new folder:",
                Sitecore.Globalization.Translate.Text("New folder"),
                Sitecore.Configuration.Settings.ItemNameValidation,
                ""$Input" is not a valid name.",
                100);
              args.WaitForPostBack();
            }
          }
        }
      }
```

ユーザーがこのコマンド テンプレートを起動すると、`Execute()` メソッドが起動され、処理コンテキストに関する情報が準備され、`Run()` メソッドに渡されます。`Run()` メソッドは、ユーザーにアイテム名を指定するよう求めてから、この名前でフォルダーを作成し、親アイテムから新しいアイテムに対して挿入オプションをコピーします。

## 4.3 コマンド テンプレートの例: ユーザー インターフェースの表示

ユーザーがアイテムを挿入する際にオプションを選択できるユーザー インターフェースを表示するコマンド テンプレートを使用することができます。

**メモ**
場合によっては、コマンド テンプレートのユーザー インターフェースをカスタム アイテム エディターとして再利用することができます。[11]

コマンド テンプレート用の次のコード サンプルは、ユーザーがフォルダーを挿入するためのユーザー インターフェースを示します。この例は、ユーザー インターフェースを表示するコマンド テンプレートの実装に必要な最小限のコンポーネントを示すために意図的に過度に簡略化されています。

```
namespace Namespace.Shell.Framework.Commands
{
  public class NewFolder : Sitecore.Shell.Framework.Commands.Command
  {
    public override void Execute(Sitecore.Shell.Framework.Commands.CommandContext
      context)
    {
      if (context.Items.Length == 1)
      {
        Sitecore.Data.Items.Item item = context.Items[0];
        System.Collections.Specialized.NameValueCollection parameters =
          new System.Collections.Specialized.NameValueCollection();
        parameters["id"] = item.ID.ToString();
        parameters["database"] = item.Database.Name;
        Sitecore.Context.ClientPage.Start(this, "Run", parameters);
      }
    }

    protected void Run(Sitecore.Web.UI.Sheer.ClientPipelineArgs args)
    {
      if (args.IsPostBack)
      {
        if ((!String.IsNullOrEmpty(args.Result)) && args.Result != "undefined")
        {
          Sitecore.Context.ClientPage.SendMessage(this,"item:load(id="+args.Result +")");
        }
      }
      else
      {
        Sitecore.Text.UrlString url = new Sitecore.Text.UrlString(
          "/sitecore modules/shell/Namespace/NewFolder.aspx");
        url.Append("id", args.Parameters["id"]);
        url.Append("database", args.Parameters["database"]);
        Sitecore.Context.ClientPage.ClientResponse.ShowModalDialog(url.ToString(),true);
        args.WaitForPostBack(true);
      }
    }
```

[11] カスタム アイテム エディターの詳細については、http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『クライアント構成リファレンス』を参照してください。

```
        }
      }
```

Execute() メソッドに渡される唯一のパラメーターは Sitecore.Shell.Framework.Commands.CommandContext オブジェクトです。このオブジェクトの Items コレクション内の最初のエレメントは、ユーザーがコマンド テンプレートを起動した際に選択した Sitecore.Data.Items.Item を示します。Execute メソッドが起動し、処理環境に関する情報を準備してから、コマンド テンプレートのロジックが含まれる Run() メソッドを起動します。

Run() メソッドは、コマンド テンプレートのユーザー インターフェース ダイアログを起動し、ユーザーがこのダイアログを完了すると、作成されたアイテムを編集用のユーザー インターフェースにロードします。Run() メソッドでは、args.IsPostBack が true である場合、ユーザーはこのインターフェースを完了しており、args.Result に値が含まれる場合、これは作成されたアイテムの ID であり、Run() メソッドはユーザー インターフェースにこのアイテムをロードさせます。args.IsPostBack が false である場合、Run() メソッドはコマンド テンプレートのユーザー インターフェースを起動します。

次のコードは、前に示した Run() メソッドによって参照されるユーザー インターフェース コンポーネント ファイル /sitecore modules/shell/Namespace/NewFolder.aspx のコンテンツを示します。

```
      <%@ Page Language="C#" AutoEventWireup="true" CodeBehind="NewFolder.aspx.cs"
Inherits="Namespace.Web.UI.NewFolder" %>
      <html>
        <head>
          <script language="javascript" type="text/javascript">
      function onClose()
      {
        window.returnValue = form1.hidCreatedId.value;
        window.close();
      }

      function onCancel()
      {
        if(confirm( "Are you sure you want to cancel?"))
        {
          window.close();
        }

        window.event.cancelBubble = true;
        return false;
      }
          </script>
          <base target=" self" />
        </head>
        <body>
          <form id="form1" runat="server">
            <div>
              <asp:textbox ID="txtName" runat="server" /><br />
              <asp:checkbox id="chkCopyInsertOptions" checked="true" runat="server"
                Text="Copy Insert Options" />
              <asp:checkbox id="chkCopyInsertRules" checked="true" runat="server"
                Text="Copy Insert Rules" /><br />
              <asp:button runat="server" id="btnCreate" text="Create" />
              <asp:button runat="server" id="btnCancel" text="Cancel"
                onclientclick="onCancel();" />
              <asp:button runat="server" id="btnDone" text="Done" onclientclick="onClose();"
                visible="false"/><br />
              <input type="hidden" runat="server" id="hidCreatedId" /><br />
```

```
            </div>
          </form>
        </body>
      </html>
```

**メモ**

理解しやすいように、このコードでは ASP.NET ページが使用されています。UI の一貫性とデベロッパーの生産性を高めるために、Sitecore Sheer UI ページを使用してコマンド テンプレートを実装することができます。ただし、Sitecore Sheer UI の実装はこの文書の範囲外です。

<base> エレメントとその属性に注意してください。これらにより、ユーザーがページ内のボタンをクリックした際にブラウザーで新しいウィンドウが開かないようにしています。[12]

次のコードは、/sitecore modules/shell/Namespace/NewFolder.aspx のコードビハインドのコンテンツを示します。

```
      namespace Namespace.Web.UI
      {
        public partial class NewFolder : System.Web.UI.Page
        {
          protected void Page Load(object sender, EventArgs e)
          {
            string dbName = Sitecore.Web.WebUtil.GetQueryString("database");
            Sitecore.Data.Database db = Sitecore.Configuration.Factory.GetDatabase(dbName);
            string parentID = Sitecore.Web.WebUtil.GetQueryString("id");
            Sitecore.Data.Items.Item parent = db.GetItem(parentID);

            if (String.IsNullOrEmpty(hidCreatedId.Value)
              && (!String.IsNullOrEmpty(txtName.Text) && txtName.Text != "undefined"))
            {
              chkCopyInsertOptions.Enabled = false;
              chkCopyInsertRules.Enabled = false;
              Sitecore.Data.Items.TemplateItem template =
                Sitecore.Context.ContentDatabase.Templates[Sitecore.TemplateIDs.Folder];
              Sitecore.Data.Items.Item item = parent.Add(txtName.Text, template);
              txtName.Enabled = false;
              btnDone.Visible = true;
              btnCreate.Visible = false;
              btnCancel.Visible = false;
              hidCreatedId.Value = item.ID.ToString();

              if(chkCopyInsertOptions.Checked||chkCopyInsertRules.Checked)
              {
                item.Editing.BeginEdit();

                if (chkCopyInsertOptions.Checked)
                {
                  item.Fields[Sitecore.FieldIDs.Branches].Value =
                    parent[Sitecore.FieldIDs.Branches];
                }

                if (chkCopyInsertRules.Checked)
                {
                  item.Fields["__insert rules"].Value = parent["__insert rules"];
```

[12] モーダル ウィンドウ内の HTML <base> エレメントの使用の詳細については、http://msdn.microsoft.com/en-us/library/ms536759(VS.85).aspx を参照してください。

```
                }

                item.Editing.EndEdit();
              }
            }
            else
            {
              if(String.IsNullOrEmpty(parent[Sitecore.FieldIDs.Branches]))
              {
                chkCopyInsertOptions.Enabled = false;
                chkCopyInsertOptions.Checked = false;
              }

              if(String.IsNullOrEmpty(parent["  insert rules"]))
              {
                chkCopyInsertRules.Enabled = false;
                chkCopyInsertRules.Checked = false;
              }
            }
          }
        }
      }
```

# Chapter 5

## 挿入オプション

この章では、挿入オプションを設定する手順について説明します。効率的な挿入オプションは、ユーザーが既存のアイテムの下に挿入できるアイテムの種類を制御します。Sitecore では、割り当てられている挿入オプションからアイテムに対して効率的な挿入オプションが決定されます。アイテムまたはデータ テンプレートの標準値に挿入ルールを割り当てることにより、データ テンプレートに基づく個別アイテムまたはすべてのアイテムに対して効率的な挿入オプションをプログラムから操作することができます。挿入オプションのパイプライン内のプロセッサーを削除、置換または追加することにより、すべてのアイテムに対して効率的な挿入オプションを操作することができます。

この章には次のセクションがあります。

- 割り当てられた挿入オプションと効率的な挿入オプション
- データ テンプレートまたは個々のアイテムへの挿入オプションの割り当て方法
- アイテム間での挿入オプションのコピー方法
- 挿入ルール
- 挿入オプションのパイプライン

## 5.1 割り当てられた挿入オプションと効率的な挿入オプション

ユーザーが既存のアイテムの下にある新しいアイテムへの挿入を可能にする機能を起動すると、ユーザーに対して効率的な挿入オプションを決定するための挿入オプションのパイプラインが起動します。効率的な挿入オプションには、ユーザーが選択したアイテムの下に新しいアイテムを挿入するために使用可能なデータ テンプレート、ブランチ テンプレートおよびコマンド テンプレートのリストが含まれます。挿入ルールまたは挿入オプションのパイプライン プロセッサーを実装しない場合、ユーザーに対して効率的な挿入オプションは、ユーザーが使用するためのアクセス権を持つ割り当て済み挿入オプションになります。挿入ルールおよび挿入オプションのパイプライン プロセッサーは、効率的な挿入オプションのリストを動的に操作することができます。

**注意**
適切なアクセス権を持つユーザーは、任意のデータ テンプレートまたはブランチ テンプレートからアイテムを移動、アイテムを複製、アイテムを挿入することができます。それ以外の場合は、挿入オプションで許可されないアイテムを作成することができます。インフォメーション アーキテクチャを適切に実現するには、CMS 機能に対するアクセス権を制限するか、挿入オプションと矛盾する特定の操作を阻止するロジックを実装する必要があります。

## 5.2 データ テンプレートまたは個々のアイテムへの挿入オプションの割り当て方法

データ テンプレートまたは個々のアイテムに挿入オプションを割り当てる方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、標準値アイテムまたは個々のアイテムを選択します。
1. [設定] タブをクリックします。
2. [挿入オプション] グループにある [割り当て] をクリックします。[オプションを挿入] ダイアログが表示されます。
3. [オプションを挿入] ダイアログで、[オプション] の下にある [テンプレート] をクリックしてから、割り当て対象のデータ テンプレート、ブランチ テンプレートおよびコマンド テンプレートを選択します。
4. [オプションを挿入] ダイアログで、[オプション] の下にある [ルールを挿入] をクリックしてから、挿入するルールを選択します。
5. [OK] をクリックします。

**重要**
データの重複と長期の管理を最小限に抑えるには、可能な場合、個々のアイテムではなく標準値に挿入オプションを割り当てます。

## 5.3 アイテム間での挿入オプションのコピー方法

ソース アイテムからターゲット アイテムに挿入オプションをコピーする方法:

1. コンテンツ エディターでソース アイテムを選択します。
2. 標準テンプレート フィールドおよびフィールドの RAW 値を表示します。[13]
3. ソース アイテムの [挿入オプション] セクション内の [オプションを挿入] および [ルールを挿入] フィールドの値を、対応するターゲット アイテムのフィールドにコピーします。
4. 標準テンプレート フィールドおよび RAW 値を非表示にします。

[13] フィールドの RAW 値を表示または非表示にする方法については、http://sdn.sitecore.net/Reference/References%20in%20Japanese.aspx にある『クライアント構成クックブック』を参照してください。

## 5.4 挿入ルール

挿入ルールは、アイテムに対して効率的な挿入オプションのリストを動的に変更することができます。

### 5.4.1 挿入ルールの作成方法

挿入ルールを作成する方法:

1. 単一の `System.Int32` パラメーターを受け入れるコンストラクターとともに `Sitecore.Data.Masters.InsertRule` を継承するクラスを作成し、`Expand()` メソッドを上書きします。次のコード サンプルを使用することができます。

```
namespace Namespace.Data.InsertRules
{
  public class ClassName : Sitecore.Data.Masters.InsertRule
  {
    public ClassName(System.Int32 count)
    {
    }

    public override void Expand(
      System.Collections.Generic.List<Sitecore.Data.Items.Item> insertOptions,
      Sitecore.Data.Items.Item item)
    {
      //TODO: manipulate insertOptions
    }
  }
}
```

2. コンテンツ エディターで、`/Templates/Common/Folder` データ テンプレートを使用して `/Sitecore/System/Settings/Insert Rules` の下に必要な任意のプロジェクト固有のフォルダーを挿入します。

3. 適切なプロジェクト固有のフォルダーに、`/System/Branches/Insert Rule` データ テンプレートを使用して挿入ルール定義アイテムを挿入します。

4. 挿入ルール定義アイテムで、Data セクション内の Type フィールドにクラスのシグネチャーを入力します。

5. 挿入ルールを他のアイテムに割り当てるには、セクション「データ テンプレートまたは個々のアイテムへの挿入オプションの割り当て方法」を参照してください。

### 5.4.2 挿入ルールの割り当て方法

データ テンプレートに基づくすべてのアイテムまたは個々のアイテムに挿入ルールを割り当てる方法:

1. テンプレート マネージャーまたはコンテンツ エディターで、標準値アイテムまたは個々のアイテムを選択します。
2. [設定] タブをクリックします。
3. [挿入オプション] グループにある [割り当て] をクリックします。[オプションを挿入] ダイアログが表示されます。
4. [オプションを挿入] ダイアログで、[オプション] の下にある [ルールを挿入] をクリックしてから、挿入ルールを選択します。

### 5.4.3 挿入ルールの例: 単一データ テンプレートに基づく複数の子の禁止

次の要件を考慮してください:

- ホーム アイテムの下で、ユーザーはさまざまなデータ テンプレートのそれぞれに基づく子アイテムを作成することができます。
- ホーム アイテムの下で、ユーザーは任意のデータ テンプレートに基づく子アイテムを 2 つ作成することはできません。

次の挿入ルールのコードは、これらの要件を満たしています。

```
namespace Namespace.Data.InsertRules
{
  public class PreventChildrenWithCommonTemplate : Sitecore.Data.Masters.InsertRule
  {
    public PreventChildrenWithCommonTemplate(System.Int32 count)
    {
    }

    public override void Expand(
      System.Collections.Generic.List<Sitecore.Data.Items.Item> insertOptions,
      Sitecore.Data.Items.Item item)
    {
      foreach (Sitecore.Data.Items.Item insertOption in insertOptions.ToArray())
      {
        if(item.Axes.SelectSingleItem("./*[@@templateid=""+insertOption.ID + ""]")!=null)
        {
          insertOptions.Remove(insertOption);
        }
      }
    }
  }
}
```

## 5.5 挿入オプションのパイプライン

挿入オプションのパイプラインには、効率的な挿入オプションを定義するためのプロセッサーが含まれます。`web.config` 内の `/configuration/sitecore/pipelines/uiGetMasters` エレメントは、挿入オプションのパイプラインを定義します。この文書の執筆時点では、デフォルトの挿入オプションのパイプラインは、それぞれが効率的な挿入オプションを操作する 3 つのプロセッサーで構成されています。これらのプロセッサーは、次のとおりです。

- **`Sitecore.Pipelines.GetMasters.GetItemMasters`**: アイテムまたはデータ テンプレートの標準値に割り当てられた挿入オプションをリストに投入します。
- **`Sitecore.Pipelines.GetMasters.GetInsertRules`**: アイテムまたはデータ テンプレートの標準値に割り当てられた挿入オプションに定義されている挿入ルールを起動します。
- **`Sitecore.Pipelines.GetMasters.CheckSecurity`**: コンテキスト ユーザーに使用アクセス権がない挿入オプションを削除します。

### 5.5.1 挿入オプションのパイプライン プロセッサー

挿入オプションのパイプライン内のプロセッサーを使用することにより、すべてのアイテムに対して効率的な挿入オプションを操作することができます。

#### 挿入オプションのパイプライン プロセッサーの実装方法

挿入オプションのパイプライン プロセッサーを実装する方法:

1. `Sitecore.Pipelines.GetMasters.GetMastersArgs` の引数を受け入れる `Process()` メソッドを実装するクラスを作成します。次のコード サンプルを使用することができます。

```
namespace Namespace.Pipelines.GetMasters
{
  public class ClassName
  {
    public void Process(Sitecore.Pipelines.GetMasters.GetMastersArgs args)
    {
      //TODO: manipulate args.Masters
    }
  }
}
```

2. `web.config` 内の `/configuration/sitecore/pipelines/uiGetMasters` にある挿入オプションのパイプライン定義エレメント内の適切な位置にプロセッサーを追加します。

**重要**
挿入オプションのパイプライン内の各パイプライン プロセッサーの位置はその機能に影響します。たとえば、デフォルトの `Sitecore.Pipelines.GetMasters.GetInsertRules` プロセッサーの前に設定されたプロセッサーの結果は挿入ルールの対象になりますが、`Sitecore.Pipelines.GetMasters.GetInsertRules` の後に設定されたプロセッサーの結果は対象になりません。コンテキスト ユーザーに使用アクセス権がない挿入オプションを削除するために、デフォルトの `CheckSecurity` プロセッサーは常にパイプライン内の最後のプロセッサーになります。

## 挿入オプションのパイプライン プロセッサーの例: 任意の場所でのフォルダーの使用許可

ユーザーが任意のアイテムの下にフォルダーを作成できるという要件を考慮してください。

次の挿入オプションのパイプライン プロセッサーのコードは、この要件を満たしています。

```
        namespace Namespace.Pipelines.GetMasters
        {
          public class AllowFolderAnywhere
          {
            public void Process(Sitecore.Pipelines.GetMasters.GetMastersArgs args)
            {
              Sitecore.Data.Items.Item folder =
args.Item.Database.Items[Sitecore.TemplateIDs.Folder];

              if(!args.Masters.Contains(folder))
              {
                args.Masters.Add(folder);
              }
            }
          }
        }
```

既存の `GetItemMasters` プロセッサーの後、かつ既存の `GetInsertRules` プロセッサーの前で、挿入オプションのパイプライン定義にプロセッサーを追加します。

```
        <uiGetMasters argsType="Sitecore.Pipelines.GetMasters.GetMastersArgs">
         <processor mode="on"
type="Sitecore.Pipelines.GetMasters.GetItemMasters,Sitecore.Kernel"/>
          <processor mode="on" type="Namespace.Pipelines.GetMasters.AllowFolderAnywhere,Assembly"
/>
          <processor mode="on" type="Sitecore.Pipelines.GetMasters.GetInsertRules,
Sitecore.Kernel"/>
          <processor mode="on" type="Sitecore.Pipelines.GetMasters.CheckSecurity,
Sitecore.Kernel"/>
        </uiGetMasters>
```

**メモ**
この例を更新して、「コマンド テンプレートの例: 親アイテムからの挿入オプションのコピー」のセクションで説明されているように、挿入オプションを親からコピーするロジックを組み込むことができます。

## 挿入オプションのパイプライン プロセッサーの例: 子の制限

ユーザーは任意のアイテムの下に指定された数の子アイテムより多くの子アイテムを作成できないという要件を考慮してください。

次の挿入オプションのパイプライン プロセッサーのコードは、この要件を満たしています。

```
namespace Namespace.Pipelines.GetMasters
{
  public class LimitChildren
  {
    private int  maxChildren = 25;

    public int MaxChildren
    {
      set
      {
         maxChildren = value;
      }
      get
      {
        return  maxChildren;
      }
    }

    public void Process(Sitecore.Pipelines.GetMasters.GetMastersArgs args)
    {
      if(MaxChildren>-1 && args.Item.Children.Count>=MaxChildren)
      {
        args.Masters.Clear();
      }
    }
  }
}
```

web.config 内の既存の Namespace.Pipelines.GetMasters.GetInsertRules プロセッサーの後、かつ既存の Namespace.Pipelines.GetMasters.CheckSecurity プロセッサーの前で、挿入オプションのパイプライン定義にプロセッサーを追加します。

```
<uiGetMasters argsType="Sitecore.Pipelines.GetMasters.GetMastersArgs">
  <processor mode="on"
type="Sitecore.Pipelines.GetMasters.GetItemMasters,Sitecore.Kernel"/>
  <processor mode="on" type="Sitecore.Pipelines.GetMasters.GetInsertRules,
Sitecore.Kernel"/>
  <processor mode="on" type="Namespace.Pipelines.GetMasters.LimitChildren,Assembly">
    <maxChildren>10</maxChildren>
  </processor>
  <processor mode="on" type="Sitecore.Pipelines.GetMasters.CheckSecurity,
Sitecore.Kernel"/>
</uiGetMasters>
```

# Chapter 6

## エイリアスおよびプロキシー

この章では、エイリアスおよびプロキシーを設定する手順について説明します。エイリアスを使用すると、1 つのコンテンツ アイテムに対して複数の URL を使用することができます。プロキシーを使用すると、共通アイテムまたはブランチを複数の場所に表示することができます。

この章には次のセクションがあります。

- エイリアス
- プロキシー

## 6.1 エイリアス

エイリアスを使用すると、単一のコンテンツ アイテムに対して任意の数の URL を使用することができます。たとえば、デフォルトでは URL /hr/jobs.aspx はコンテンツ アイテム `/Sitecore/Content/Home/HR/Jobs` を起動します。エイリアスを作成することにより、デフォルトの URL /hr/jobs.aspx に加えて URL /jobs.aspx によって `/Sitecore/Content/Home/HR/Jobs.` を起動することができます。

### 6.1.1 エイリアスの作成方法

エイリアスを作成する方法:

1. コンテンツ エディターで、`/Sitecore/Content/Home/HR/Jobs` などのターゲット アイテムを選択します。
2. [プレゼンテーション] タブをクリックします。
3. [URL] グループにある [エイリアス] をクリックします。
4. `/jobs` や `jobs` (両方とも同等) などのエイリアス名を入力します。
5. [追加] をクリックします。

**メモ**
統合および管理されているパイプライン モードに設定されているアプリケーション プールとともに IIS7 を使用する場合、`web.config` 内の `/configuration/sitecore/linkManager` エレメントに `addAspxExtension` 属性の値を設定することにより、`.aspx` 拡張子を取り除くことができます。この変更を実装するためにエイリアスを使用する必要はありません。

**注意**
エイリアスを使用すると、単一のコンテンツ アイテムに対して複数の URL が存在することになり、検索エンジンが混乱する可能性があります。エイリアスを使用する場合はその存在する場所に一貫性を保つようにするか、この問題に対処するために別の解決策を実行してください。

## 6.2 プロキシー

このセクションでは、プロキシーを処理する手順について説明します。プロキシーを使用すると、コンテンツ ツリーのソースの場所にストアされているアイテムまたはブランチがターゲットの場所で複製されて表示されます。プロキシーはプロキシー定義アイテムによって定義されます。複数の場所で単一ソース アイテムをプロキシーしたり、その子孫アイテムを各プロキシーの場所に組み込んだりすることができます。

### 6.2.1 プロキシーの有効化方法

プロキシーは、プロキシーを使用するデータベースごとに有効にする必要があります。データベースに対してプロキシーを有効にする方法は、次のとおりです。

1. データベース名に相当する `name` 属性を持つ `web.config` 内の `/configuration/sitecore/databases` エレメントを見つけます。

2. `<proxiesEnabled>` エレメントの値を `true` に設定します。

```
        <databases>
          ...
          <database id="master" singleInstance="true" type="Sitecore.Data.Database,
Sitecore.Kernel">
            ...
            <proxiesEnabled>true</proxiesEnabled>
            ...
          </database>
          ...
          <database id="web" singleInstance="true" type="Sitecore.Data.Database,Sitecore.Kernel">
            ...
            <proxiesEnabled>true</proxiesEnabled>
            ...
          </database>
          ...
        </databases>
```

### 6.2.2 プロキシーの作成方法

プロキシーを作成する方法:

1. コンテンツ エディターで、`/Sitecore/System/Proxies` に移動します。

2. `/System/Proxy` データ テンプレートを使用してプロキシー定義アイテムを挿入します。

3. プロキシー定義アイテムで、[Data] セクション内の [Source アイテム] フィールドで、プロキシーするソース アイテムを選択します。

4. Target アイテム フィールドについては、プロキシー アイテムの親として機能するアイテムを選択します。

5. Proxy タイプ フィールドについては、個々のアイテムをプロキシーする場合は Root アイテムのみを選択し、アイテムの階層全体をプロキシーする場合は Entire サブツリーを選択します。

6. Source データベース フィールドについては、同じデータベースからアイテムをプロキシーする場合は値を入力せず、別のデータベースからアイテムをプロキシーする場合はデータベース名を入力します。