**アプリケーション開発者向け**

**Microsoft® Windows 7   
対応アプリケーションの互換性**

**免責**

このドキュメントに記載されている情報は、このドキュメントの発行時点におけるマイクロソフトの見解を反映したものです。マイクロソフトは市場の変化に対応する必要があるため、このドキュメントの内容に関する責任をマイクロソフトは問われないものとします。

このホワイト ペーパーに記載された内容は情報提供のみを目的としており、明示、黙示、または法令に基づく規定に関わらず、これらの情報についてマイクロソフトはいかなる責任も負わないものとします。

この文書およびソフトウェアを使用する場合は、適用されるすべての著作権関連の法律に従っていただくものとします。著作権法による制限に関係なく、マイクロソフトの書面による許可なしに、この文書の一部または全部を複製したり、検索システムに保存または登録したり、別の形式に変換したりすることは、手段、目的を問わず禁じられています。ここでいう手段とは、複写や記録など、電子的、または物理的なすべての手段を含みます。

マイクロソフトは、このドキュメントに記載されている内容に関し、特許、特許申請、商標、著作権、またはその他の知的所有権を有する場合があります。マイクロソフトから提供される使用許諾書に明記されていない限り、この文書の配布によりこれらの特許、商標、著作権、またはその他の知的財産権がお客様に譲渡されることはありません。

***2009 Microsoft Corporation.All rights reserved.***

Microsoft、Windows 7™は、米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

この文書およびソフトウェアで使用されている実在の会社名および製品名は、該当各社の商標です。



このドキュメントは、Windows 7 RC 版における情報をもとに構成しています。仕様および機能は変更される可能性があります。

|  |
| --- |
| ***目次*** |

[1. はじめに 1](#_Toc229470662)

[1.1 アプリケーション互換性へのMicrosoft の取り組み 1](#_Toc229470663)

[1.2 アプリケーション互換性テクノロジ 3](#_Toc229470664)

[2. 検証と対応のポイント 6](#_Toc229470665)

[2.1 アプリケーション互換性問題の傾向 6](#_Toc229470667)

[3. 一般的な互換性問題 7](#_Toc229470668)

[3.1 非公開情報に依存しない 7](#_Toc229470670)

[3.2 OS のバージョンチェック 8](#_Toc229470671)

[Windows の内部バージョン番号 8](#_Toc229470672)

[OS のバージョンチェックの検証 8](#_Toc229470673)

[適切なバージョンチェック 9](#_Toc229470674)

[3.3 ファイルやフォルダーパスの取得 12](#_Toc229470675)

[ファイルやフォルダーパスの変更 12](#_Toc229470676)

[フォルダーやファイルパスの取得方法 13](#_Toc229470677)

[3.4 ファイルへの署名 15](#_Toc229470678)

[デジタル署名の必要性 15](#_Toc229470679)

[デジタル署名による改ざんの防止 15](#_Toc229470680)

[Authenticode 証明書とは 16](#_Toc229470681)

[Authenticode の利用場面 17](#_Toc229470682)

[他の機能との連動 18](#_Toc229470683)

[ソフトウェアへのデジタル署名 19](#_Toc229470684)

[3.5 x64 バージョンのサポート 24](#_Toc229470685)

[サポートされないソフトウェア 24](#_Toc229470686)

[64ビットバージョンのアプリケーションの作成 24](#_Toc229470687)

[4. 新機能や機能変更に伴う互換性問題 26](#_Toc229470688)

[4.1 リソースの管理 26](#_Toc229470690)

[フォルダー構造の変更点 26](#_Toc229470691)

[ディレクトリの接合点 27](#_Toc229470692)

[環境変数の変更点 30](#_Toc229470693)

[リソース管理への対応 31](#_Toc229470694)

[4.2 整合性レベル ( IL ) によるオブジェクトへのアクセス制御 33](#_Toc229470695)

[4.3 ユーザーアカウント制御 ( UAC：User Account Control ) 34](#_Toc229470696)

[セキュリティに関する問題点 34](#_Toc229470697)

[Windows XPでの対策と問題点 34](#_Toc229470698)

[UAC の目的と特徴 35](#_Toc229470699)

[UAC のしくみ 35](#_Toc229470700)

**Windows 7 Only**

[Windows 7 での変更点 42](#_Toc229470701)

[UAC 対応アプリケーションを新規に作成するには 43](#_Toc229470702)

[Windows 7での既存アプリケーションへの対応 51](#_Toc229470703)

[互換性を検証するには 52](#_Toc229470704)

[UAC事例 53](#_Toc229470705)

[4.4 ユーザーインターフェイス特権の分離 ( UIPI ) 57](#_Toc229470706)

[UIPI とは 57](#_Toc229470707)

[UIPI に対応するには 57](#_Toc229470708)

[UIPIによる問題を検証するには 58](#_Toc229470709)

[4.5 IE のセキュリティ機能 60](#_Toc229470710)

[Internet Explorer ( IE ) の保護モード 60](#_Toc229470711)

[デジタル証明書とHTTPS 63](#_Toc229470712)

[4.6 Windows リソース保護 ( WRP ) 65](#_Toc229470713)

[WRP とは 65](#_Toc229470714)

[既存アプリケーションへの影響と回避策 65](#_Toc229470715)

[WRP対応アプリケーションの作成 66](#_Toc229470716)

[4.7 セッション 0 の分離 68](#_Toc229470717)

[セッション0の分離とは 68](#_Toc229470718)

[互換性への影響と検証のポイント 70](#_Toc229470719)

[セッション0の分離への対応 71](#_Toc229470720)

[4.8 ファイルマッピングオブジェクト 72](#_Toc229470721)

[SeCreateGlobalPrivilege 権限が必要 72](#_Toc229470722)

[回避方法および対応方法 73](#_Toc229470723)

[4.9 新フォント環境 74](#_Toc229470724)

[JIS2004対応 74](#_Toc229470725)

[メイリオフォント 77](#_Toc229470726)

[4.10 プログラムの互換性アシスタント ( PCA ) 78](#_Toc229470727)

[PCA とは 78](#_Toc229470728)

[インストール失敗の検出 78](#_Toc229470729)

[プロセス作成の失敗の検出 79](#_Toc229470730)

[既知の問題に対するメッセージプロセス作成 79](#_Toc229470731)

[4.11 Internet Explorer 8 の標準準拠 80](#_Toc229470732)

[これまでの IE の各バージョンの描画モード 82](#_Toc229470733)

[これまでの IE のDTD 判別ロジック 83](#_Toc229470734)

[IE8 の描画モード 84](#_Toc229470735)

[X-UA-CompatibleによるIE7 互換モードへの切り替え 85](#_Toc229470736)

[X-UA-Compatibleの設定値 87](#_Toc229470737)

[段階的なサイト互換性の実現 89](#_Toc229470738)

[4.12 DEP/NXの既定値 90](#_Toc229470739)

**Windows 7 Only**

**Windows 7 Only**

**Windows 7 Only**

[DEP/NX とは 90](#_Toc229470740)

[発生する互換性問題 90](#_Toc229470741)

[問題の回避 91](#_Toc229470742)

[問題の解決 91](#_Toc229470743)

**Windows 7 Only**

[4.13 Windows メールの削除 92](#_Toc229470744)

[Windows メールの削除 92](#_Toc229470745)

[発生する互換性問題 92](#_Toc229470746)

[問題の回避と対応 93](#_Toc229470747)

**Windows 7 Only**

[4.14 容易な High DPI 設定 94](#_Toc229470748)

[High DPI とは 94](#_Toc229470749)

[発生する互換性問題 94](#_Toc229470750)

[問題の回避と対応 95](#_Toc229470751)

[4.15 長音表記の変更 97](#_Toc229470752)

[長音表記の変更 97](#_Toc229470753)

[発生する互換性問題と問題の解決 97](#_Toc229470754)

[5. Application Compatibility Toolkit ( ACT ) 98](#_Toc229470755)

[5.1 ACTとは 98](#_Toc229470757)

[5.2 Compatibility Administratorの利用 99](#_Toc229470758)

[**Compatibility Administrator** の起動 99](#_Toc229470759)

[アプリケーションへ適用する互換フィックスの定義 101](#_Toc229470760)

[カスタムデータベースファイルへの保存 104](#_Toc229470761)

[コンピューターへの展開 105](#_Toc229470762)

[参考情報 106](#_Toc229470763)

[付録 107](#_Toc229470764)

[描画モード切り替え表 107](#_Toc229470765)

# はじめに

Windows 7 は、ユーザーエクスペリエンスやセキュリティの向上のため、革新的な機能が多数導入され、既存の機能も改善されています。同時に、全体的には高い互換性も実現しており、Windows XP や Windows Vista など、以前のWindows 用に作成されたアプリケーションの多くは、今までと同じように利用することができます。しかし、Windows VistaおよびWindows 7では優れた新機能、高いセキュリティ、信頼性などを優先させたことで、互換性が保てない部分があることも確かです。

このドキュメントでは、互換性に関する既知の問題点について検証・回避・対応する方法を紹介します。

## アプリケーション互換性へのMicrosoft の取り組み

新しい OS を導入する際には、これまでのアプリケーションやハードウェアが使用できるかどうかが重要な決定要因の一つになります。Microsoft では、Windows 7 をリリースするにあたり、ユーザーや開発者が互換性問題の解決にかかる時間やコストを削減できるよう、さまざまなツールや情報、リソースを提供しています。

* 問題回避のためのアプリケーション互換性テクノロジ

Windows 7 では、既存のアプリケーションが動作するようWindows 95 やWindows XP、Windows Vistaなど、これまでの OS のエミュレーション機能を用意しています。どの OS のエミュレータを使用してアプリケーションを実行するかは、各ユーザーが設定することもできますし、管理者が組織のコンピューターにまとめて設定するためのツールを使用することもできます。

詳細は「1.2 アプリケーション互換性テクノロジ」を参照してください。

* ベータプログラムの実施と検証ガイドの提供

ベータプログラムや、RC (Release Candidate) をはじめとするさまざまなソフトウェアや情報を開発者向けに提供することで、Windows 7 の早期採用を支援しています。

* アプリケーション開発ガイドラインの提供

Windows SDKやMSDN、Windowsデベロッパーセンター (http://msdn.microsoft.com/ja-jp/windows/) などを通じ、アプリケーション開発ガイドラインを提供しています。また、互換性に関しては「Windows 互換性情報サイト」 (http://www.microsoft.com/japan/windowsvista/compatible/) において、アプリケーションの互換性テストを実施する際のノウハウや、アプリケーション開発者向けの情報などを公開しています。

* 互換性の高いアプリケーションを認定

Microsoft では、技術要件を満たしたアプリケーションを認定しています。このロゴプログラムに合格することで、製品にロゴを付けることができます。これにより、他製品との差別化が図れ、互換性や信頼性、セキュリティなどの品質を顧客に提示することができます。デバイスやソフトウェアは「Compatible with Windows 7」ロゴを、PC は「Windows 7」ロゴを取得することができます。

[](http://news.softpedia.com/images/news2/Windows-7-Client-Software-Logo-Program-Looking-to-Windows-8-3.png)

* 1. Compatible wirh Windows 7 ロゴ
* OS 展開時の互換性問題の回避や軽減をサポート

Windows 7 でのアプリケーションの互換性に関する検証やテストをスムーズにおこなえるよう、Application Compatibility Toolkit ( ACT ) を提供しています。ACT については、「5. Application Compatibility Toolkit」を参照してください。

## アプリケーション互換性テクノロジ

「アプリケーション互換性テクノロジ」とは、Windows のバージョンアップに伴い発生するアプリケーションの互換性問題を、回避したり、軽減したりするために開発されたテクノロジです。具体的には、以下のようなソリューションを提供しています。

* 互換フィックス

アプリケーションが問題のある関数を呼び出している場合、以前のOS と互換性のある関数の呼び出しに置き換えて、アプリケーションが期待する結果を OS が返すようにします。たとえば、古いアプリケーションに対し、以前のOS のバージョン番号を返すことで、そのアプリケーションが正しく実行できるようにしたりします。

* 互換モード

特定のOS 環境をエミュレートするための互換フィックスの集まりです。たとえば、「Windows 95 互換モード」には、69個の互換フィックスが含まれ、Windows 95 用に設計されたアプリケーションが Windows 7 上で動作できるようにします。

* アプリケーションヘルプ

互換フィックスや互換モードが利用できない状況で、互換性のないアプリケーションを実行・インストールしようとした時に、表示されるヘルプメッセージです。このヘルプメッセージダイアログで、アプリケーションの実行を許可もしくは、禁止することができます。

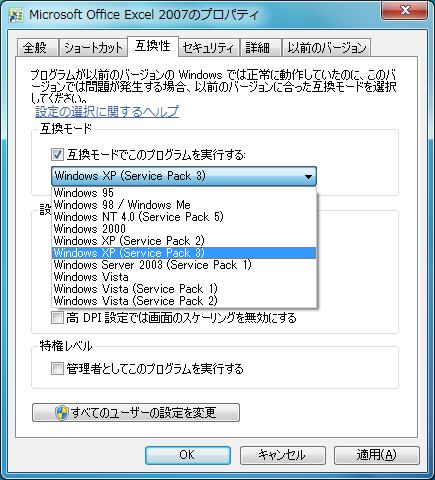
また、これらのソリューションをより簡単に適用するためのツールも提供されています。

* [ 互換性 ] タブ

プログラムのプロパティで、「互換性」タブをクリックすると、アプリケーションの実行環境を変更することができます。この設定はユーザーが各コンピューターで操作することができるため、手軽に設定することができます。

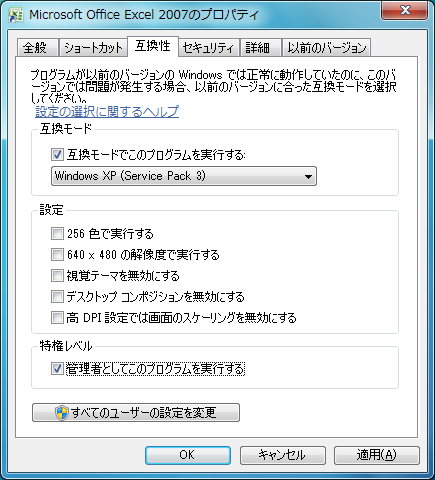
* 「互換モード」

特定の OS 環境をエミュレートできます。互換モードは、互換性フィックスの集合体であるため、OS を選択すると、内部的には複数の互換性フィックスが適用されることになります。



* 1. 互換性タブ
* 「管理者としてこのプログラムを実行する」

Windows 7 では、既定でアプリケーションは標準ユーザーで実行されます。管理者権限が必要なアプリケーションは、このオプションを有効にすることで、常にAdministrators 権限で実行することができます。

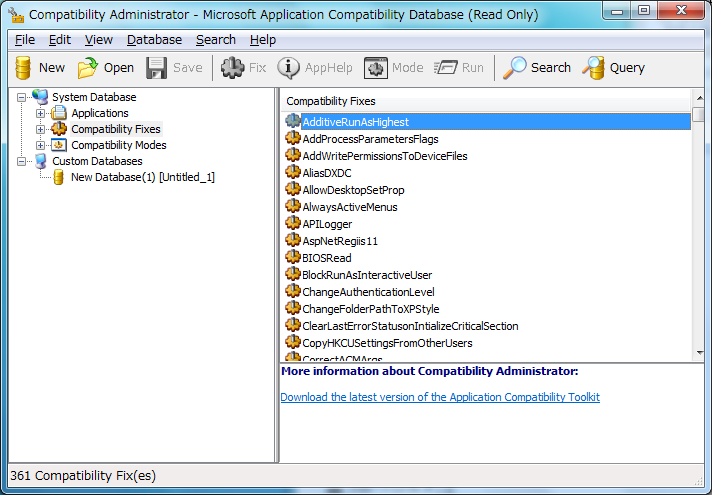


* 1. 管理者として実行
* Compatibility Administrator

組織内の多数のコンピューターで、同じアプリケーションを動作させたい場合、各コンピューターで [ 互換性 ] タブを使用して同じ設定をおこなうのは手間がかかります。また、互換モードは、いくつもの互換性フィックスの組み合わせであるため、実際はアプリケーションに必要ないものまで適用している可能性もあります。

そこで、Application Compatibility Toolkit に含まれるCompatibility Administratorツールを使用すると、アプリケーションに必要な互換性フィックスの組み合わせを１つずつ定義することができるので、互換モードを使用するよりも、細やかな設定をすることができます。また、その組み合わせをデータベースとして保存しておいて、組織内のすべてのコンピューターに適用することも簡単にできます。

Compatibility Administratorの使用方法は、「5.2 Compatibility Administratorの利用」を参照してください。



* 1. Compatibility Administrator

# 検証と対応のポイント

## アプリケーション互換性問題の傾向

アプリケーションの互換性問題には、以下の2種類の傾向があります。

* 一般的な互換性問題

Windows 7に限らず、Windows OS がバージョンアップする際には、いつも発生する問題です。つまり、Windows 7以前の OS のバージョンアップの際にも発生してきた問題であり、将来のバージョンアップの際にも発生する可能性があります。このタイプの問題は、アプリケーションの実装時に配慮することで、問題を回避できるため、アプリケーション開発時に気をつけなければなりません。

また、これらの問題の発生原因を理解したうえでアプリケーションを作成することで、Windows 7 だけでなく、それ以降の OS にもスムーズに移行することができます。

本ドキュメントでは、3章で一般的な互換性問題について扱います。

* Windows Vista / Windows 7の互換性問題

Windows XP からWindows Vista へのバージョンアップでは、多くの機能が追加・変更されたため、多くの互換性問題が発生する可能性があります。特にセキュリティが大幅に強化されているため、脆弱なアプリケーションでは互換性問題が発生します。たとえば、システムのリソースに無制限にアクセスしているようなアプリケーションは、正しく動作しません。

一方、Windows Vista から Windows 7 のバージョンアップでは、大きなアーキテクチャの変更はありません。また、Windows 7 は、Windows Vista で利用されてきたアプリケーションやドライバーはすべて動作させるという理念のもと開発されたため、互換性問題もあまり起こりません。本ドキュメントでは、Windows 7 でのみ発生する互換性問題に加え、Windows Vista と Windows 7に共通で発生する問題も紹介します (4章) 。Windows 7 でのみ発生する問題に関しては、以下のようなマークを示しています。すでに Windows Vista への対応が完了している場合は、この部分のみご確認ください。

**Windows 7 Only**

# 一般的な互換性問題

この章では、以下のような一般的な互換性問題について紹介します。

* 非公開情報に依存しない
* OS のバージョンチェック
* ファイルやフォルダーパスの取得
* ファイルへの署名
* x64 バージョンのサポート

## 非公開情報に依存しない

Microsoft が推奨・公開している方法を使用して、アプリケーションを開発します。SDK では公開されていない関数・データ構造体・レジストリなどは、次のバージョンアップの際に変更されたり、なくなったりする可能性があります。そのため、その時の OS では動作しても、次のバージョンの OS で問題が発生する可能性があります。

## OS のバージョンチェック

### Windows の内部バージョン番号

Windows 7 の内部バージョン番号は、「6.1」です。内部バージョン番号は、OS のバージョンアップに伴い更新されます。これまでの Windows の内部バージョンは以下のとおりです。

|  |  |
| --- | --- |
| Windows の種類 | 内部バージョン |
| Windows 2000 | 5.0 |
| 32ビット版 Windows XP 各エディション | 5.1 |
| Windows Server 2003  Windows XP x64 Edition | 5.2 |
| Windows Vista  Windows Server 2008 | 6.0 |
| Windows 7  Windows Server 2008 R2 | 6.1 |

アプリケーションが「OS の特定のバージョンかどうか」をチェックしている場合、正しく動作しない可能性があります。たとえば、バージョンが「5.1かどうか」をチェックしている場合には、Windows 7 は「6.1」を返すため、正しく動作しません。

特にWindows Vista と Windows 7 では、Windows 2000 以来のメジャーバージョンアップがおこなわれています。そのため、メジャーバージョンのみをチェックしているアプリケーションは、Windows XPや Windows Server 2003 までは正しく動作しても、Windows Vista や Windows 7 では正しく動作しない可能性があります。たとえば、この場合はバージョンが「5以上かどうか」をチェックすることで正しく動作させることができます。

### OS のバージョンチェックの検証

バージョンチェック問題による実行の失敗は、アプリケーションによって発生するタイミングが異なります。一般的にはインストール時やアンインストール時、起動時にバージョンチェックをすることが多いため、これらのタイミングで失敗することが多いといえます。

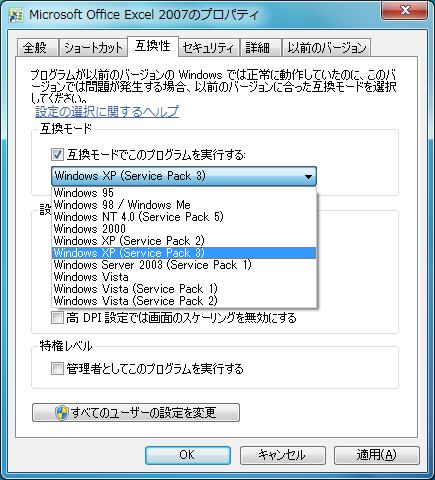


* 1. バージョンエラー

#### 互換性テクノロジを使用した問題の回避

* OSの互換モードの使用

アプリケーションのバージョンチェック問題が原因で、インストールや起動ができないなら、ファイルのプロパティで「Windows XP ( Service Pack 3 ) 」互換モードなどのオプション有効にします。これにより、以前の OS に近い環境でアプリケーションを実行することができます。



* 1. [互換性] タブ
* VersionLie 互換フィックスの使用

「VersionLie」は、アプリケーションに対して OS のバージョン番号を偽ることができる互換フィックスです。これにより、アプリケーションからバージョン番号が要求されたときに、以前の Windows のバージョン番号を回答することができます。この互換フィックスはさまざまなOS 用のものが提供されており、たとえば、WinXPSP3VersionLie 互換フィックスでは、アプリケーションに対して Windows XP SP3 のバージョン番号を返します。これにより、バージョン番号が一致していないだけの理由で、実行が失敗するといった問題に対処することができます。

ただし、VersionLie は、アプリケーションのバージョン番号のチェックをすり抜けているにすぎません。Windows 7 では許可されない処理が呼び出されると、そのタイミングで失敗します。

### 適切なバージョンチェック

既存のアプリケーションを、新しい OS にインストールできない最大の理由は、アプリケーションがバージョン番号を正しく扱わないことです。このようなアプリケーションでは、バージョンチェックの部分さえ取り除いてしまえば、新しい Windows でも問題なく動作することがよくあります。ここでは、正しいバージョンチェックの方法を紹介します。

#### 将来のバージョンでのアプリケーションの使用を許可する

* 目的のバージョン以上かどうかをチェックする

将来の Windows でもアプリケーションの使用を許可するなら、バージョンをチェックする際に、「目的のバージョンと一致するかどうか」ではなく、「目的のバージョン以上かどうか」をチェックします。

* レジストリの CurrentVersion をチェックしない

レジストリに直接アクセスして、OS のバージョンを取得しないようにします。

* 他のバージョンへのインストールを制限する

低レベルのユーティリティで、特定の OS に密接に関連している場合や、EULA (使用許諾契約書) で将来の OS での使用が禁止されている場合は、故意に他の OS へのインストールを拒否することもできます。この場合には、以下のようなメッセージを出すなどして、システムに悪影響を及ぼさないように失敗させる必要があります。



* 1. インストール制限

#### バージョンではなく機能をチェックする

OS に特定の機能があるかどうかを検証したい場合、バージョン番号だけをチェックするのは最善の方法ではありません。新しい OS がリリースされたり、サービスパックなどの提供により、新しい機能が追加されたりすることもあるからです。

特定の機能の有無を確認するには、バージョン番号に依存するのではなく、その機能をサポートしているかどうかをチェックすることが重要です。これは、Windows 開発チームの推奨です。

* GetVersionEx()

GetVersionExは、現在の OS のバージョン番号を返す API です。たとえば、以下のコードは、OS のメジャーバージョンが「6」以上かどうかを確認しているので、 Windows Vista 以降かどうかを確認することができます。

OSVERSIONINFO osvi;

osvi.dwOSVersionInfoSize = sizeof(OSVERSIONINFO);

GetVersionEx(&osvi);

BOOL bIsWindowsVistaOrLater =

(osvi.dwPlatformId == VER\_PLATFORM\_WIN32\_NT) &&

(osvi.dwMajorVersion >= 6);

* VerifyVersionInfo()

VerifyVersionInfo() APIは、単にバージョンを返すのではなく、現在の OS がアプリケーションの要件に沿っているかどうかをtrue もしくはfalse で答えます。要件には、OS のメジャーバージョン、マイナーバージョンおよびサービスパックを指定することができ、現在のシステムがそのバージョンかそれ以上であれば、true を返します。つまり、「SP3 を適用した Windows XP かどうか」といった条件を与えると、Windows 7 がインストールされている場合にもtrue を返します。ただし、この API はWindows 2000 以降でのみサポートされます。

* Environment.OSVersion プロパティ

.NET Framework では、System.Environment.OSVersionプロパティを使用して、OS のバージョンチェックをすることができます。

## ファイルやフォルダーパスの取得

### ファイルやフォルダーパスの変更

Windows では、各リリースでいくつかのフォルダーの位置が変更されています。たとえばマイドキュメントフォルダーは、Windows 95 からWindows 7 までの間に以下のようにパスが変わっています。

|  |  |
| --- | --- |
| OSの種類 | マイドキュメントのパス |
| Windows 9x | C:\My Documents |
| Windows NT 4.0 | C:\Windows\Profiles\<username>\Personal |
| Windows 2000  Windows XP  Windows Server 2003 | C:\Documents and Settings\<username>\My Documets |
| Windows Vista  Windows 7  Windows Server 2008 | C:\Users\<username>\Documents |

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** **Windows Vista と Windows 7のマイドキュメント** |
| Windows Vista では、マイドキュメントフォルダーの表示名 (エクスプローラで開いたときに表示される名前) は、「ドキュメント」でした。しかし、Windows ７ では、「マイドキュメント」に戻っています。これは、Windows 7 では、図3-4のように、「ライブラリ」フォルダー内に「ドキュメント」フォルダーが作成され、その中に、「マイドキュメント」と「パブリックのドキュメント」が含まれるようになったからです。この「マイドキュメント」が「Document」フォルダーです。     * 1. ドキュメントとマイドキュメント   ただ、Windows 7でも、「Documents」というフォルダー名自体は変更されていないので、Windows Vista への対応が完了しているアプリケーションはそのまま使うことができます。 |

※他にも変更されているフォルダーがあります。詳細は、「4.1 リソースの管理」を参照してください。

### フォルダーやファイルパスの取得方法

#### やってはいけないこと

* パスのハードコード

マイドキュメントのフォルダーパスは Windows のバージョンによって変更されているため、特定の場所へのパスがハードコードされているアプリケーションは、ほかのバージョンの Windows では動作しない可能性があります。実際、この問題が多くのアプリケーション非互換の原因になっています。

また、ドメインのようなエンタープライズ環境で運用している場合、ユーザーのマイドキュメントフォルダーは、ローカルディスクではなく、ネットワークフォルダーに移動されることもあります。これにより、ユーザーはドメイン内のどのコンピューターからでも、移動ユーザープロファイルを使用してアクセスすることができますが、パスをハードコードしていると、ローカル システム上でフォルダーを検索するアプリケーションの場合は、場所が変更されたときに マイドキュメントフォルダーを見つけることができません。

#### すべきこと

* 適切なAPI によるフォルダーパスの取得

API を使用して、特定のフォルダーのパスを取得します。これにより、たとえOS が変更されても、アプリケーションには新しいパスが返されるため、パスの変更による互換性の問題が発生しません。以下にAPIの例を示します。

|  |  |
| --- | --- |
| API | 説明 |
| GetWindowsDirectory() | Windows フォルダーのパスを取得 |
| GetSystemDirectory() | Windows システムフォルダーのパスを取得 |
| GetTempPath() | システムの一時フォルダーのパスを取得 |
| SHGetFolderPath() | CSIDL 値によって識別されたWindows 特殊フォルダーへのパスを取得 |

たとえば、マイドキュメントなどのフォルダーを見つけるには、SHGetFolderPath() を使用してパスを取得します。以下にこの API を使ったコード例を示します。

HMODULE hModSHFolder = LoadLibrary("shfolder.dll");

if ( hModSHFolder != NULL )

{

(\*(FARPROC\*)&g\_pfnSHGetFolderPath =

GetProcAddress(hModSHFolder, "SHGetFolderPathA"));

}

Else

{

g\_pfnSHGetFolderPath = NULL;

}

if (g\_pfnSHGetFolderPath != NULL )

g\_pfnSHGetFolderPath(NULL, CSIDL\_SYSTEM, NULL, NULL, szSystem32);

else

szSystem32[0] = '\0';

OpenFileName.lpstrInitialDir = szSystem32;

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| このコードでは、「CSIDL\_SYSTEM」CSLID値を使用して、Windowsのシステムフォルダーへのパスを取得しています。以下に、よく使用される CSIDL 値を示します。   |  |  | | --- | --- | | CSIDL 値 | 説明 | | CSIDL\_SYSTEM | System フォルダーへのパスを取得 | | CSIDL\_PROGRAM\_FILES | Program Filesフォルダーへのパスを取得 | | CSIDL\_PERSONAL | マイドキュメントフォルダーへのパスを取得 | | CSIDL\_COMMON\_APPDATA | アプリケーションデータへのパスを取得 | |

* 適切なレジストリの使用

レジストリからパスを取得する際は、データ型を確認しなければなりません。データ型が「REG\_EXPAND\_SZ」の場合には、ExpandEnvironmentStrings関数を使用することができます。「REG\_EXPAND\_SZ」型は、展開可能な文字列型のことで、環境変数を含むことができます。環境変数を含むレジストリ値を読み込むときは、ExpandEnvironmentStrings 関数を使用することで、「%WinDir%」のような環境変数を現在の値に置き換えることができます。

## ファイルへの署名

### デジタル署名の必要性

インターネットからソフトウェアをダウンロードする場合、そのソフトウェアが適切な発行元で、適切に作成されたことが保証されないと、信用できないソフトウェアや悪意のあるソフトウェアをインストールしてしまう可能性があります。これらをインストールしてしまうと、システムの安定性やセキュリティに悪影響を及ぼします。

そこで、ソフトウェアが正規の発行元から提供されていて、しかも改ざんされていないことを証明するために、デジタル署名を使用することができます。デジタル署名により、管理者やユーザーはソフトウェアをインストールする前に、信頼できるソフトウェアかどうかを確認することができます。

### デジタル署名による改ざんの防止

デジタル署名は、公開鍵暗号基盤( PKI ) を使用しており、改ざんを防ぐことができます。たとえば、ソフトウェアの発行元がデジタル署名すると、以下のような手順で安全にソフトウェアを送信できます。

1. ソフトウェアの発行元は、コードの作成後、ハッシュアルゴリズムを使用して、メッセージダイジェスト ( MD ) を生成します。
2. メッセージダイジェストを、ソフトウェアの発行元の秘密キーを使用して暗号化します。これがデジタル署名です。
3. ユーザーは、コードとともに、デジタル署名もダウンロードします。
4. ユーザーは、発行元から送信されたコードをハッシュし、メッセージダイジェストを生成します。
5. ユーザーは、発行元の公開キーでデジタル署名を復号し、メッセージダイジェストを取り出します。
6. ユーザーは、④でハッシュした結果と、⑤で復号したメッセージダイジェストを比較し、同じであれば、送信時に改ざんされていないことが保証されます。

**ソフトウェアの発行元**

コード



MD

①ハッシュ

**ユーザー**

コード



MD

⑥比較

MD

④ハッシュ

③送信

デジタル

署名

②発行元の

秘密キー

で暗号化

⑤発行元の

公開キー

で復号

③送信

デジタル

署名

* 1. デジタル署名を利用したソフトウェアの送信

デジタル署名では、ユーザーが取得した、「発行元の公開キーが正当なものである」ということが前提になっています。公開キーが改ざんされた場合には、デジタル署名は意味を持ちません。そこで、公開キーの正当性を確保するために「デジタル証明書」を使用します。デジタル証明書はインターネットにおける電子的な身分証明書で、証明機関 ( CA : Certification Authority ) により発行されます。デジタル証明書には発行元の「公開キー」が含まれているため、送信先のユーザーに提示すれば、自分の公開キーを安全に送信することができます。

### Authenticode 証明書とは

Windows では、デジタル証明書を利用したコードへの署名の手法として、「Authenticode」といわれる技術を提供しています。Authenticode で使用される証明書のことを「Authenticode 証明書」といいます。以下に、Authenticode 証明書がどのように使用されているのかを示します。

1. ソフトウェアの発行元は、証明機関 ( CA ) に「Authenticode証明書」を要求します。証明機関の審査を通過すると、証明書を取得することができます。Authenticode 証明書は、複数のベンダーから入手可能です。取得するためには、一定の条件が必要となります。
2. ユーザーがソフトウェアのダウンロードを要求した際に、コードとともに発行元の「Authenticode 証明書」も一緒に送信します。
3. ユーザーは証明書の中から、公開キーを取り出します。

これにより、図3-5の手順⑤で使用する公開キーを取得したことになるため、受け取ったデジタル署名を復号することができます。

**ソフトウェアの発行元**

**ユーザー**

①Authenticode証明書の要求・発行・取得

②送信

**証明機関(CA)**

発行元の証明書

公開キー

発行元の証明書

公開キー

公開キー

③取り出し

* 1. 証明書を利用した公開キーの送信

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** **証明機関 ( CA:Certification Authority )** |
| 証明書を発行する証明機関は、ユーザーのコンピューター上に「信頼する証明機関」として登録されている必要があります。ユーザーが信頼する証明機関は、Internet Explorer で [ ツール ] メニューから[ インターネットオプション ] を選択し、「コンテンツ」タブから「証明書」ボタンをクリックして確認することができます。この画面で、信頼する証明機関を追加することもできます。     * 1. 信頼された証明機関 |

### Authenticode の利用場面

Windows XPまでは、インターネットから発行されたActive Xコントロールや Java アプレットなどのダウンロードパッケージや実行可能ファイル、およびドライバーをインストールする際に、デジタル署名をチェックし、ユーザーが信頼していない発行元のソフトウェアには、警告を表示することでインストールを阻止することができました。

Windows Vista や Windows 7 ではこれに加えて、いくつかの新機能がデジタル署名と連動しているため、デジタル署名の使用が義務付けられているコードが増えています。対象となるソフトウェアは以下の通りです。

* Internet Explorer を使用してダウンロード、インストールされるパッケージや自己解凍形式の実行可能ファイルは、実行・インストールするためにデジタル署名が必要です。
* 未署名のカーネルモードコンポーネントをインストールするには、管理者特権が必要です。これには、デバイスドライバー、フィルタドライバー、サービスなどが含まれます。
* ハードウェア関連のドライバーは、Windows ロゴプログラムを取得するためにデジタル署名が必要です。
* Windows Vista や Windows 7 の 64 ビットバージョンでは、署名済みのドライバーが必要です。
* 起動時に読み込まれるドライバーのバイナリには、埋め込み署名を含める必要があります。
* カーネルモードのドライバーには、WHQL や DRS プログラムを通じて入手するMicrosoft による署名が必要です。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** **WHQL (Windows Hardware Quality Labs ) とは** |
| コンピューターや周辺機器などのハードウェアやドライバーが、Windows 対応製品としてふさわしいかどうかをテストする組織です。WHQL で認定されると、Windows 対応ロゴを取得することができます。 |

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote: DRS ( Driver Reliability Signature ) とは** |
| Windows Vista から導入されているプログラムで、デバイスのロゴを取得するためのベースとなるテストを提供します。DRS に合格すると、Microsoft からデジタル署名ファイルを入手することができます。 |

Authenticode 証明書は、バージョンに依存しないので、Windows XP で利用していたものが使えます。また、証明書には有効期限がありますが、有効期限内であれば何度でも利用することができます。つまり、1つの証明書で複数のソフトウェアを署名することができます。

### 他の機能との連動

Windows VistaやWindows 7 では、以下の新機能がデジタル署名と連動しています。

* 添付ファイル 実行サービス ( AES )

AES ( Attachment Execution Service ) はWindows XP SP 2 以降でサポートされている機能です。メールに添付された圧縮ファイル ( .zip や.lzh ) に含まれる .exe や .com の実行をブロックすることができます。

* ユーザーアカウント制御 ( UAC )

権限昇格ダイアログが表示される際に、発行元の情報を表示することができます。UACに関しては、「4.3 ユーザーアカウント制御 ( UAC：User Account Control )」を参照してください。

* Internet Explorer 8

既定では未署名の Active X コントロールのインストールは制限されています。

### ソフトウェアへのデジタル署名

* デジタル署名の方法

Windows VistaやWindows 7 では、すべての実行可能ファイルに Authenticode 証明書を使用して署名すべきです。拡張子.exe、.dll、.ocx、.sys、.cpl、.drvおよび.scr のついたファイルが対象です。Authenticode 証明書による署名のための手順を以下に示します。

1. Authenticode 証明書をサードパーティの証明機関から入手します。信頼されるサードパーティの商用証明機関一覧は、Microsoft の以下の Web サイトで確認できます。

http://msdn.microsoft.com/en-us/library/ms995347.aspx

1. このような証明機関は、申請者の資格を確認し、その結果によって申請者にAuthenticode 証明書を発行しています。
2. ソフトウェアの発行元は証明書を入手後、Windows SDK ( ソフトウェア開発キット ) に付属する署名ツール ( Signtool.exeなど) を使用して署名します。Signtool.exe コマンドラインツールの使用例は、以下の通りです。

signtool sign /a MyFile.exe

各オプションの意味は、以下の通りです。

|  |  |
| --- | --- |
| オプション | 説明 |
| sign | ファイルにデジタル署名する際に指定します。 |
| /a | 最適な証明書を自動的に選択する際に指定します。 |
| MyFile.exe | 署名するファイルへのパスを指定します。 |

Signtool.exe の詳細は、Microsoft の以下の Webサイトを参照してください。

http://msdn.microsoft.com/ja-jp/library/8s9b9yaz.aspx

* テスト証明書を使用したソフトウェアのテスト

通常、Authenticode 証明書は企業内で管理されています。そのため、開発やテストの段階では使用できないことがあります。MakeCert コマンドを使用すると、テスト用の証明書を作成することができます。使用例は、次の通りです。

MakeCert -r -pe -ss TestCertStoreName -n "CN=CertName"   
CertFileName.cer

各オプションの意味は、次の通りです。

|  |  |
| --- | --- |
| オプション | 説明 |
| -r | 証明書が自己署名されていることを指定します。つまり、これによりルート証明書が作成されます。 |
| -pe | 証明書に関連する秘密キーのエクスポートを可能とみなします。 |
| -ss TestCertStoreName | テスト証明書を保持する証明書ストアの名前を指定します。 |
| -n "CN=CertName" | 証明書を特定する際に、コマンド ライン署名ツールで使用できる証明書の名前を指定します。テスト証明書として明確に特定できる証明書名を使用することをお勧めします。 |
| CertFilename.cer | テスト証明書のコピーを含むファイル名を指定します。証明書ファイルは、信頼されたルート証明機関および信頼された発行元の証明書ストアに証明書を追加する際に使用します。 |

MakeCertの詳細については、Microsoft 以下のWebサイトを参照してください。

http://msdn.microsoft.com/ja-jp/library/bfsktky3.aspx

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| ソフトウェアをテストするコンピューターでは、対応する MakeCert テスト証明書をそのコンピューターの信頼されたルート証明機関の証明書ストアと、信頼された発行元の証明書ストアにインストールする必要があります。発行元が信頼されていないと、インストールの際に、信頼されていないことを通知する、図3-8のようなダイアログが表示されます。     * 1. 不明な発行元を告げるダイアログボックス   あらかじめ、MakeCert テスト証明書をそのコンピューターの信頼されたルート証明機関の証明書ストアなどにインストールしておくと、ソフトウェアパッケージは、署名付きとして扱われるため、図3-9のようなダイアログボックスが表示されます。     * 1. 発行者の確認   証明書をインストールするためには、Internet Explorer で [ ツール ]メニューから、[ インターネットオプション ] を選択し、「コンテンツ」タブから「証明書」ボタンを選択します。 |

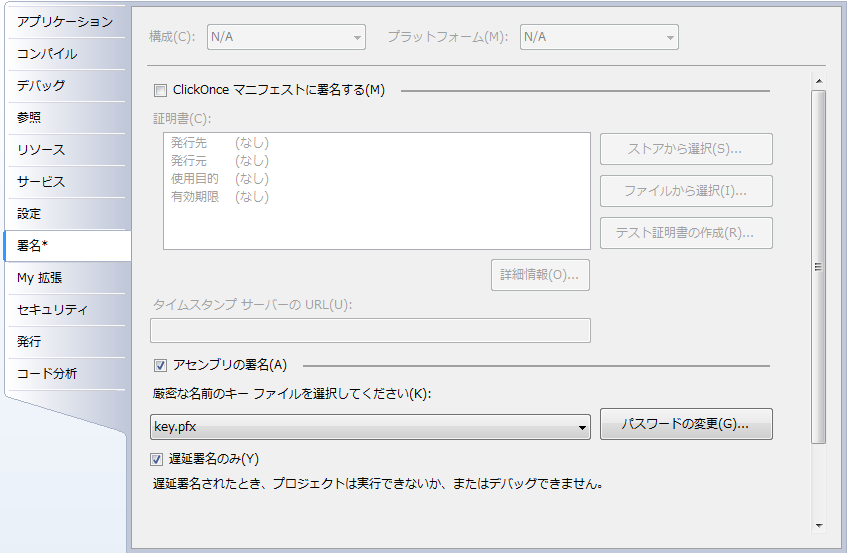
* 遅延署名

厳密名付きのアセンブリを作成するときには、秘密キーと公開キーが必要です。秘密キーはデジタル署名のために使われます。また、公開キーは厳密名の一部となるため、他のアセンブリがそのアセンブリを参照するときなどには必須となります。

しかし、企業においては、公開キーや証明書は容易に取得できても、秘密キーは厳重に管理されているため、開発者が日常的にアクセスすることは難しいといえます。このような場面では、遅延署名を使用します。遅延署名を利用すると、ビルド時は、デジタル署名用のスペースを予約するだけで、署名しないため、秘密キーがなくてもビルドすることができます。

Visual Studio を使用して、遅延署名を設定する手順は、以下の通りです。

1. プロジェクトのプロパティで、[ 署名 ] タブを選択します。
2. [ アセンブリの署名 ] チェックボックスをオンにします。
3. 公開キーが保存されたキーファイルを指定します。
4. [ 遅延署名のみ ] チェックボックスをオンにします。
5. アプリケーションをビルドします。



* 1. [署名]タブ

遅延署名を設定したプロジェクトは、ビルドはできますが実行はできません。また、グローバルアセンブリキャッシュにも登録できません。これは現段階では署名がないからです。しかし、以下のコマンドを実行すると、一時的にそのアセンブリの署名を確認しないようにすることができるため、アセンブリが実行できるようになります。

sn.exe -Vr Application.dll

|  |  |
| --- | --- |
| オプション | 説明 |
| -Vr アセンブリ名 | 署名の検査をスキップするアセンブリ名を指定します。 |

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| このオプションは、開発時にのみ設定します。署名の検証を省略したままにしておくと、悪意のあるアセンブリによって、偽装される可能性があります。解除方法は、後述します。 |

出荷直前に企業の秘密キーでデジタル署名するために、以下のコマンドを実行します。

sn.exe –R Application.dll

|  |  |
| --- | --- |
| オプション | 説明 |
| -R アセンブリ名 | 遅延署名を有効にしたアセンブリに、デジタル署名を追加します。 |

前の手順で「sn.exe -Vr」を実行しているため、このままでは署名の検証がおこなわれません。

署名の検証を有効にするため、必ず以下のコマンドを実行します。

sn.exe -Vu Application.dll

|  |  |
| --- | --- |
| オプション | 説明 |
| -Vu アセンブリ名 | 署名検査のスキップを解除します。 |

## x64 バージョンのサポート

Windows VistaやWindows 7 では、AMD や Intel の64ビットアーキテクチャプロセッサを完全にサポートしています。64 ビット版 Windows 7 では、64ビットのネイティブコードのほか、WoW64 ( Windows on Windows 64 ) サブシステムにより32ビットアプリケーションの動作環境が提供されます。

### サポートされないソフトウェア

以下のソフトウェアはWoW64 上で実行することはできません。

* 16ビットの実行可能プログラムやインストーラ
* 32ビットのカーネルモードドライバー
* 64ビットのデジタル署名のないドライバー

これらの問題点に対する回避策はないため、以下の方法で対処します。

* 16ビットの実行可能プログラムはすべて削除し、32または64ビットの同等のプログラムに置き換える。
* 16ビットのインストーラを32または64ビットのインストーラに変換する。
* アプリケーションでカーネルモードのドライバーを使用する場合、64ビットのドライバーを用意する。
* すべての64ビットドライバーにデジタル署名する。

#### ファイルリダイレクションとレジストリリダイレクション

WoW64は、32ビットアプリケーションと、64 ビットカーネルを仲介します。そして、Windows VistaやWindows 7 では互換性のために32ビット用と64 ビット用の2つのシステムファイルが用意されていますが、32ビットのアプリケーションからのアクセスの場合には、32ビット用のシステムディレクトリに自動的に切り替えられるようになっています。このファイルリダイレクション機能は、WoW64により提供されます。

同様に、ソフトウェア用のレジストリも32ビット用と64ビット用が用意されていて、WoW64のレジストリリダイレクション機能により自動的にリダイレクトされます。

### 64ビットバージョンのアプリケーションの作成

.NET Framework 1.0や1.1で作成されたアプリケーションは、すべて32ビットアプリケーションとして作成されます。一方、.NET Framework 2.0 以降では、32ビットに加え、64ビットのアプリケーションも作成できるようになっています。どのタイプのアプリケーションを作成するかは、コンパイラのオプションにより設定できます。以下に主要な言語での設定方法を紹介します。

|  |  |
| --- | --- |
| 言語 | 設定方法 |
| Visual Basic | Visual Studio 2008でプロジェクトのプロパティを開き、[ コンパイル ] タブから「詳細コンパイルオプション」ボタンをクリックし、「ビルドの詳細設定」ダイアログボックスから「ターゲットCPU」を指定します。各オプションは以下の通りです。   * AnyCPU   任意のプラットフォームで実行するアセンブリをコンパイルします。   * x86   32ビットのCLR上で実行するアセンブリをコンパイルします。   * x64   64ビットのCLR上で実行するアセンブリをコンパイルします。   * Itanium   Itaniumプロセッサ搭載コンピューター上の64ビットCLR上で実行するアセンブリをコンパイルします。(このオプションは、Visual Studio Team System でのみサポートされます) |
| C# | Visual Studio 2008でプロジェクトのプロパティを開き、[ ビルド ] タブから「プラットフォームターゲット」を選択します。設定できるオプションは Visual Basic と同じです。 |
| C++ | Visual Studio 2008でプロジェクトのプロパティを開き、[ 構成 ] －[ 全般 ] を選択し、「共通言語ランタイムサポート」で「安全な MSIL 共通言語ランタイム サポート (/clr:safe)」を選択します。これにより、任意のプラットフォームで実行するアセンブリをコンパイルします。  ネイティブ64ビットアプリケーションを作成する方法は、Microsoftの Web サイト (http://msdn.microsoft.com/ja-jp/library/h2k70f3s.aspx ) を参照してください。 |

# 新機能や機能変更に伴う互換性問題

この章では、以下のような新機能や機能変更に伴う、互換性問題について紹介します。

* リソースの管理
* 整合性レベル ( IL ) によるオブジェクトへのアクセス制御
* ユーザーアカウント制御 ( UAC：User Account Control )
* ユーザーインターフェイス特権の分離 ( UIPI )
* Internet Explorer のセキュリティ機能
* Windows リソース保護 ( WRP )
* セッション 0 の分離
* ファイルマッピングオブジェクト
* 新フォント環境
* プログラムの互換性アシスタント ( PCA )
* Internet Explorer 8 の標準準拠
* DEP/NXの既定値
* Windows メールの削除
* 容易な High DPI 設定
* 長音表記の変更

## リソースの管理

### フォルダー構造の変更点

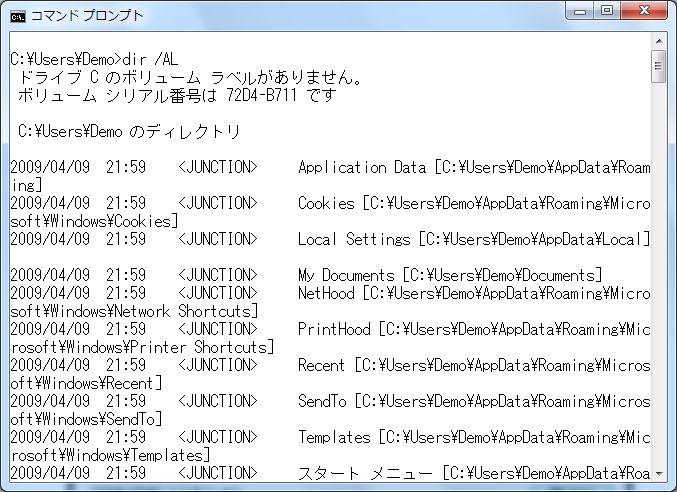
Windows Vista とWindows 7では、ユーザーデータやアプリケーションデータを格納するためのフォルダーの構造が大幅に変更されました。おもな変更点は、以下の通りです。

|  |  |  |
| --- | --- | --- |
| 項目 | Windows XP | Windows Vista / 7 |
| ユーザープロファイルの ルートフォルダー名を変更 | Documents and Settings | Users |
| ユーザーごとのアプリケーション  データ(ローミングあり/なし)を  一元化 | *<username>*\App…  *<username>*\LocalSettings \App… | *<username>*\AppData |
| ユーザーごとのアプリケーション  データフォルダーの場所を変更 | *<username>* \Application Data | *<username>*\AppData \Roaming |
| *<username>* \Local Settings \Application Data | *<username>*\AppData \Local |
| All Users プロファイルのフォルダー名を変更 | All Users | Public |
| All Users プロファイルの  アプリケーションデータフォルダーの場所を変更 | All Users\Application Data | C:\ProgramData |
| ごみ箱を各ユーザープロファイル  フォルダーへ作成 | X:Recycler\*<SID>* | Users\*<username>* \$Recycle.bin |

### ディレクトリの接合点

フォルダー構造が変更されると、既存アプリケーションが正しくフォルダーにアクセスできないために、動作しない可能性がでてきます。そこで、救済策として「接合点」が用意されています。つまり、変更前のフォルダーの位置には、接合点が用意されていて、アプリケーションが以前のフォルダーやファイルへの書き込みアクセスを試みると、新しいフォルダーへリダイレクトされます。たとえば、「C:\Documents and Settings」は「C:\Users」 を示す接合点として設定されています。

接合点は、System 属性が設定され、ACLは 「Everyone　読み取り拒否」に設定されています。そのため、既定では エクスプローラには表示されません。しかし、コマンドプロンプトで「dir /AL」を実行すると表示することができます。図4-1は「dir /AL」の実行結果です。接合点は「JUNCTION」と表示されており、リダイレクト先も表示されています。



* 1. dir /AL の実行

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote: dir /AL とは** |
| dir コマンドで、 /A オプションを指定すると、さまざまな属性のファイルを表示することができます。中でも、「L」オプションは、再解析ポイント (接合点) を表示するために使用します。 |

接合点が作成されているパスと、リダイレクト先は、次の通りです。

#### ユーザーのデータフォルダーの接合点

|  |  |
| --- | --- |
| 接合点の位置 | リダイレクト先 |
| Documents and Settings | Users |

#### ユーザーごとのーデータフォルダーの接合点

|  |  |
| --- | --- |
| 接合点の位置 | リダイレクト先 |
| Documents and Settings\*<username>* \My Documents | Users\*<username>*\Documents |
| Documents and Settings\*<username>* \My Documents\My Music | Users\*<username>*\Music |
| Documents and Settings\*<username>* \My Documents\My Pictures | Users\*<username>*\Pictures |
| Documents and Settings\*<username>* \My Documents\My Videos | Users\*<username>*\Videos |

#### ユーザーごとのアプリケーションデータフォルダーの接合点

|  |  |
| --- | --- |
| 接合点の位置 | リダイレクト先 |
| Documents and Settings\*<username>* \Local Settings | Users\*<username>*\AppData\Local |
| Documents and Settings\*<username>* \Local Settings\Application Data | Users\*<username>*\AppData\Local |
| Documents and Settings\*<username>* \Local Settings\Temporary Internet Files | Users\*<username>*\AppData\Local \Microsoft\Windows\Temporary Internet Files |
| Documents and Settings\*<username>* \Local Settings\History | Users\*<username>*\AppData\Local \Microsoft\Windows\History |
| Documents and Settings\*<username>* \Application Data | Users\*<username>*\AppData\Roaming |

#### ユーザーごとの OS 設定フォルダーの接合点

|  |  |
| --- | --- |
| 接合点の位置 | リダイレクト先 |
| Documents and Settings\*<username>*\Cookies | Users\*<username>*\AppData\Roaming \Microsoft\Windows\Cookies |
| Documents and Settings\*<username>*\Recent | Users\*<username>*\AppData\Roaming \Microsoft\Windows\Recent |
| Documents and Settings\*<username>*\Nethood | Users\*<username>*\AppData\Roaming \Microsoft\Windows\Network Shortcuts |
| Documents and Settings\*<username>*\Printhood | Users\*<username>*\AppData\Roaming \Microsoft\Windows\Printer Shortcuts |
| Documents and Settings\*<username>*\SendTo | Users\*<username>*\AppData\Roaming \Microsoft\Windows\Send To |
| Documents and Settings\*<username>* \スタート メニュー | Users\*<username>*\AppData\Roaming \Microsoft\Windows\StartMenu |
| Documents and Settings\*<username>*\Templates | Users\*<username>*\AppData\Roaming \Microsoft\Windows\Templates |

#### All Users フォルダーの接合点

|  |  |
| --- | --- |
| 接合点の位置 | リダイレクト先 |
| Users\All Users | ProgramData |

#### Default User フォルダーの接合点

|  |  |
| --- | --- |
| 接合点の位置 | リダイレクト先 |
| Documents and Settings\Default User | Users\Default |
| Documents and Settings\Default User\Desktop | Users\Default \Desktop |
| Documents and Settings\Default User \My Documents | Users\Default \Documents |
| Documents and Settings\Default User\Favorites | Users\Default \Favorites |
| Documents and Settings\Default User \My Documents\My Music | Users\Default \Music |
| Documents and Settings\Default User  \My Documents\My Pictures | Users\Default\Pictures |
| Documents and Settings\Default User  \My Documents\My Videos | Users\Default\Videos |
| Documents and Settings\Default User \Application Data\ | Users\Default\AppData\Roaming |
| Documents and Settings\Default User \Start Menu\ | Users\Default\AppData\Roaming\Microsoft  \Windows\Start Menu |
| Documents and Settings\Default User \Templates\ | Users\Default\AppData\Roaming  \Microsoft\Windows\Templates |

### 環境変数の変更点

フォルダーの構造が変更されたことに伴い、環境変数も追加・変更されています。

新規に追加された環境変数は、Windows Vista 以降の Windows をターゲットにするときにのみ使用します。つまり、Windows Vista やWindows 7 で追加された環境変数は、Windows XPでは認識されずエラーが発生してしまいます。

#### 新しい環境変数

|  |  |
| --- | --- |
| 環境変数 | 値 |
| %AppData% | AppData\Roaming |
| %LocalAppData% | AppData\Local |
| %ProgramData% | ProgramData |
| %Public% | Public |

#### パスが変更された環境変数

|  |  |
| --- | --- |
| 環境変数 | 値 |
| %AllUsersProfile% | ProgramData |
| %Temp% または %TMP% | AppData\Local\Temp |
| %UserProfile% | Users\<username> |

#### 変更されない環境変数

|  |  |
| --- | --- |
| 環境変数 | 値 |
| %CommonProgramFiles% | Program Files\Common Files |
| %ProgramFiles% | Program Files |
| %WinDir% | Windows |

### リソース管理への対応

#### 適切なリソースを使用する

フォルダーやレジストリにアクセスするときには、適切なフォルダーを選択する必要があります。標準ユーザーにはProgram Files フォルダーなどへの書き込み権がありません。そのため、これらのフォルダーにアプリケーションから一時的に利用するファイルを追加していると、エラーが起こってしまいます。実行に必要な情報は、できるだけユーザーごと、つまり「Users\*<username>*」などに保存するようにします。

なお、推奨される保存先の一覧を以下に示します。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| 「標準ユーザー」に関しては、「4.3 ユーザーアカウント制御 ( UAC：User Account Control)」を参照してください。 |

##### マシン全体のデータの保存先

すべてのユーザーに共通する設定などを保存するときに使います。ただし、既定ではこれらのフォルダーへの書き込み権は、標準ユーザーにはありません。管理者権限でのアクセスが必要となるため、インストール時などに設定を保存するようにします。

|  |  |
| --- | --- |
| プログラムファイル | %ProgramFiles% |
| システムファイル | %WinDir% |
| 共有データ | %ProgramData%  ( %AllUsersProfile% ) |
| レジストリ | HKLM |

##### ユーザーごとのデータの保存先

各ユーザーに関する設定を保存するときに使います。これらのフォルダーへの書き込み権は、標準ユーザーにあります。そのため、ユーザーがアプリケーションを実行している時に書き込みが可能です。

|  |  |
| --- | --- |
| ユーザーのファイルの保存先 | ドキュメントフォルダーなど |
| ローミングなしのデータ | %LocalAppData% |
| ローミングありのデータ | %AppData% |
| レジストリ | HKCU |

#### 適切な API を使用する

リソースへのアクセスは、ハードコードしないようにし、SHGetFolderPath() などの適切な関数を使用するようにします。関数などの詳細は、「3.3 ファイルやフォルダーパスの取得」を参照してください。

## 整合性レベル ( IL ) によるオブジェクトへのアクセス制御

整合性レベル ( IL ) は、プロセスへの権限を制限するための新しいメカニズムです。Windows VistaやWindows 7 では、他のプロセスやファイル、レジストリといったオブジェクトへのアクセス権限を、3つのレベルに分けています。一番高いレベルを「高 IL 」、中間を「中IL 」、一番低いレベルを「低 IL 」といいます。たとえば、信頼できないプロセスなら、「低IL 」で実行することで、アクセスできるリソースが少なくなるため、システムへの影響を最小限に抑えることができます。各レベルでアクセスできるリソースは、以下の通りです。

|  |  |  |
| --- | --- | --- |
| ILの種類 | 権限 | アクセスできる フォルダーやレジストリの例 |
| 高IL | 管理者権限  システム用の領域への書き込み可 | %ProgramFiles% や %WinDir%  HKLM |
| 中IL | 標準権限  ユーザー用の領域に書き込み可 | %UserProfile%  HKCU |
| 低IL | 信頼できない権限  安全な領域への書き込みのみ可 | %UserProfile%\AppData\LocalLow  HKCU\Software\AppDataLow |

これから紹介する以下の機能は、いずれも整合性レベルを使用することで、リソースへのアクセスを制御しています。

* UAC ( User Account Control ：ユーザーアカウント制御 )
* UIPI ( User Interface Privilege Isolation： ユーザーインターフェイス特権の分離 )
* 保護モードの IE

## ユーザーアカウント制御 ( UAC：User Account Control )

### セキュリティに関する問題点

ここ数年の間に、Windowsを狙ったウイルスやワームが増加しました。こうした攻撃のために、企業や家庭のコンピューターを使用するユーザーは、修復に膨大な時間や費用を費やしました。スパイウェアなどの悪意のあるソフトウェア (マルウェア)は 、疑いを持たないユーザーによってインストール、起動されることもあります。これらは、有用で一見害のないソフトウェアに潜んでいることがあるからです。

管理者としてログオンしているときに悪意のあるコードが実行されると、被害が大きくなります。これは、管理者特権で悪意のあるコードが実行されてしまうからです。たとえば、管理者としてログオンしていると、ブラウザーや電子メールクライアントなどのアプリケーションも、管理者権限を持つことになります。これらのアプリケーションが悪意のあるソフトウェアを起動すると、悪意のあるソフトウェアは、それ自体をインストールし、ウイルス対策ソフトウェアを操作して OS から見えなくすることさえできてしまいます。そして、セキュリティ侵害されたサイトにアクセスしたり、電子メールのリンクをクリックしたりすることで、ユーザーが意図していないにもかかわらず、悪意のあるソフトウェアが実行されてしまう可能性があります。

### Windows XPでの対策と問題点

この脅威に対抗するため、Windows XPでは、LUA ( Least-privileged User Account ) を採用しました。LUA アプローチでは、管理者は2つのユーザーアカウントを使い分けます。つまり、管理者といえども、ブラウザーや電子メールクライアントなどを利用するときには、「制限付きユーザーアカウント」を使います。そして、ディスク操作やユーザー作成などの管理作業をおこなうときには、「管理者ユーザーアカウント」を利用します。

この方法は、管理者が管理タスクをおこなうときに、ログオンしなおさなければなりません。そのわずらわしさのため、結局は「常に管理者ユーザーでログオンする」という状況を生む結果となりました。また、管理者権限でないと正常に動作しないアプリケーションが多いことも、この状況に拍車をかけてしまいました。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote: 制限付きユーザーアカウントとは** |
| 制限付きユーザーアカウントとは、権限が制限されていて、偶発的または故意によるシステム全体への変更が許可されていないユーザーアカウントです。最小限の権限しか持っていないため、攻撃によるシステムへの影響が少ないアカウントといえます。具体的には、ローカルコンピュータのUsers グループのメンバーです。また、ドメインに参加しているコンピューターでは、Domain Users グループのユーザーアカウントは、既定でローカルコンピュータの Users グループのメンバーに含まれます。 |

### UAC の目的と特徴

#### UAC とは

Windows Vista や Windows 7 では、LUA の代わりに、UAC (User Account Control：ユーザーアカウント制御) 権限モデルを採用しています。UAC は、LUA 同様、ユーザーが意図しないコードが実行されないようにします。しかし、使い勝手が大幅に改善されています。以下にUAC の特徴を挙げます。

* 管理者は2つのユーザーアカウントを使い分ける必要がない

1つのユーザーアカウントに、通常のユーザーとしての役割と、管理者としての役割を与えることができます。そのため、ログイン、ログアウトを繰り返す必要はありません。管理者は、常に管理権限のある1人のユーザーでログオンでき、しかも安全にアプリケーションを実行することができます。

* プロセスを分離可能

UACでは、プロセス単位で実行するユーザーの権限を変えることができます。管理権限が必要なコードをメインのアプリケーションと切り分ければ、アプリケーション全体を管理者で実行しなくてもよいため、より安全な環境を提供することができます。

* シームレスな遷移が実現される

管理者特権が必要になると、管理者として実行するかどうかを確認する画面が表示されます。この画面で承認すれば、自動的に管理者として実行されます。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| UAC はコントロールパネルの設定により、無効にすることもできます。しかし無効にすることでセキュリティ上のメリットがなくなってしまうため、推奨されません。 |

### UAC のしくみ

#### 権限の昇格が可能

Windows Vista やWindows 7 では、管理者を含む全てのユーザーは、既定で「標準ユーザー」としてアプリケーションを実行します。「標準ユーザー」とはUsers グループのユーザーです。標準ユーザーが実行するプロセスは「中 IL 」の整合性レベルが使用されるため、システム領域のリソースへのアクセスが制限されます。

管理者権限が必要なアプリケーションやタスクを実行する際は、標準ユーザーでは権限が足りません。そこで、管理権限が必要な場合には、図4-2のような「承認ユーザーインターフェイス」を表示します。このダイアログボックスで「はい」ボタンをクリックすると、管理者に「昇格」され、そのアプリケーションやタスクは、管理者権限で実行されることになります。昇格により実行されるプロセスは「高 IL 」の整合性レベルが使用されるため、システム領域のリソースへのアクセスも可能です。



* 1. 承認ユーザーインターフェイス

#### ログインは2つのセキュリティトークンを提供する

権限の昇格を実現するために、ログオン時にユーザーのトークンを「フルトークン」と「フィルタ済みトークン」に分割します。そして、いずれかのトークンがアプリケーションに渡されます。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote: トークンとは** |
| ユーザーや所属するグループ、保持している特権などの情報により、ユーザーをあらわしたものです。システムはトークンを使用して、セキュリティで保護されたオブジェクトへのアクセスや、システム関連の操作を実行する権限を制御します。 |

「フルトークン」は、ユーザーが保持している全ての権限が定義されているトークンです。一方、「フィルタ済みトークン」は、ユーザーが保持している全ての権限のうち、標準ユーザーが持つものだけを定義したトークンです。「フィルタ済みトークン」では管理作業はおこなえません。

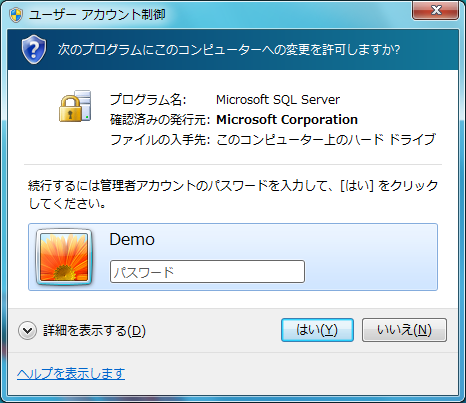
既定では、全てのユーザーは、「フィルタ済みトークン」でアプリケーションを実行します。管理者としてログオンしている場合も例外ではありません。そして、管理権限が必要になった場合には、図4-2のような「承認ユーザーインターフェイス」が表示され、「はい」ボタンをクリックすると、権限が昇格されフルトークンで動作します。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote: フィルタ済みトークンの例外** |
| ビルトインの Administrator ユーザーは、はじめから管理者権限で全てのアプリケーションが実行されます。そのため、「昇格」の必要はありません。ただし、既定では、Administrator ユーザーアカウントは無効になっています。     * 1. Administrator は既定では無効 |

#### 承認ユーザーインターフェイスと資格情報ユーザーインターフェイス

管理者としてログオンしていて、権限の昇格が必要になると、図4-2のような「承認ユーザーインターフェイス」が表示されます。ここで「はい」ボタンをクリックすることで、そのアプリケーションやタスクがフルトークンで実行されることになります。「キャンセル」ボタンをクリックした場合には、昇格はおこなわれません。

一方、標準ユーザーなどでログオンしている場合には、昇格が必要になると、図4-4のような「資格情報ユーザーインターフェイス」が表示されます。このダイアログボックスでは、管理者の資格情報を入力できます。これにより、標準ユーザーでログオンしていても、特定のタスクだけ管理者で実行することができます。



* 1. 資格情報ユーザーインターフェイス

また、どちらのダイアログボックスの場合にも、タイトルバーのすぐ下の帯の色に注目する必要があります。この色は、アプリケーションの発行元によって色が変わります。

|  |  |
| --- | --- |
| 帯の色 | アプリケーションの発行元 |
| 青 | OS が提供するアプリケーション |
| 青 | 署名つきのアプリケーション (Windows Vista ではグレー) |
| 黄 | 署名なしのアプリケーション |
| 赤 | ブロックされている発行元から提供されたアプリケーション |

#### 盾 ( シールド ) アイコン

##### 盾アイコンとは

昇格が必要なアプリケーションやタスクには、図4-5のような「盾」が表示されます。



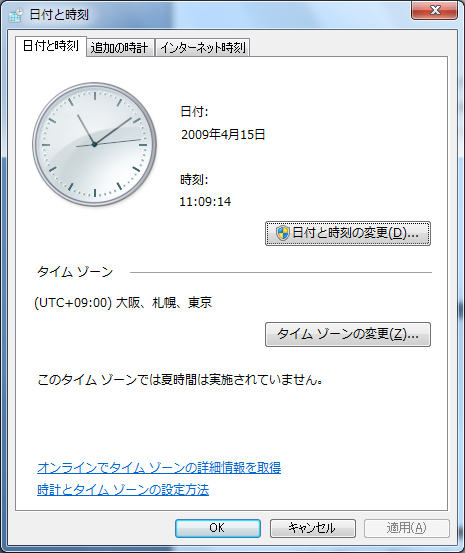
* 1. 盾アイコン

図4-6のようにファイルのアイコン自体に盾が含まれる場合は、アプリケーションを起動しようとしたタイミングで、昇格するための「承認ユーザーインターフェイス」ダイアログが表示されます。そして、アプリケーション全体が管理者権限で実行されます。



* 1. 盾が含まれているファイルのアイコン

これに対し、一部のタスクでのみ昇格を必要とする場合は、図4-7のように、ボタンなどに盾アイコンが表示されます。ここで盾アイコンの付いたボタンをクリックすると、「承認ユーザーインターフェイス」ダイアログが表示され、そのタスクのみが管理者権限で実行されます。



* 1. 盾が含まれているボタン

いずれの場合にも、管理者権限で実行されるのは、そのアプリケーションやタスクのみです。それ以外のアプリケーションやタスクは、標準ユーザーのまま実行されます。

盾アイコンの目的は、盾アイコンの付いた操作を実行すると特権の昇格が必要になることを、一目でわかるようにすることです。ユーザーは、盾アイコンの付いたアイテムをクリックすると、権限の昇格が求められることを予測できます。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote: 盾アイコンは常に表示される** |
| UAC を無効にしていたり、ユーザーが組み込みの Administrator アカウントを使用していたりする場合は、昇格の必要はありませんが、盾アイコンは表示されます。また、盾アイコンには、1つの状態しかありません。盾アイコンが無効になったり、カーソルを合わせたときに外観が変わったり、選択された状態になったりすることはありません。 |

##### 盾アイコンを表示するプログラムを作成する方法

開発者は、クリックするとその操作が特権の昇格を必要とするコントロールに、容易に盾アイコンを付けることができます。詳細はこの節の「UAC 対応アプリケーションを新規に作成するには」を参照してください。

#### 昇格してアプリケーションを実行する方法

##### 昇格しないでアプリケーションを実行できるようにする

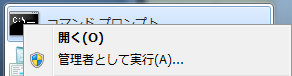
できる限り、標準ユーザーが昇格なしで実行できるようにプログラムを設計します。標準ユーザーの生産性を十分に確保し、初期配置後に管理操作をおこなう必要がないように製品を設計します。

管理権限が必要な既存のアプリケーションを実行する場合は、アプリケーションの起動時に昇格する必要があります。そのためには、以下の4つの方法がありますが、これらはいずれもWindows VistaやWindows 7 での設定を必要とするだけで、アプリケーション自体を変更する必要がありません。

##### 「管理者として実行」メニューの利用

アプリケーションを起動する際、アイコンやスタートメニューを右クリックし、コンテキストメニューから「管理者として実行」を選択します。

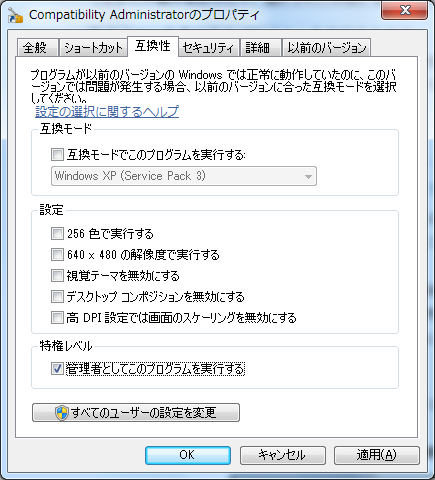
これにより昇格し、そのアプリケーションは、管理者権限で実行されます。



* 1. [ 管理者として実行 ] コンテキストメニュー

##### [ 互換性 ] タブの利用

アプリケーションのプロパティを開き、[ 互換性 ] タブで、「管理者としてこのプログラムを実行する」を選択します。これにより、常に管理者権限で実行することができます。この方法は、各コンピューターで設定できるので、ユーザー向けといえます。



* 1. [ 互換性 ] タブ

##### 互換フィックスの利用

企業内の多数のコンピューターに適用するためには、Compatibility Administrator ツールを使用して、互換フィックスを適用します。利用可能な互換フィックスは以下のとおりです。

|  |  |
| --- | --- |
| 互換フィックス | 説明 |
| RunAsUAC | 親プロセスと同じ権限で動作 |
| RunAsHighest | ユーザーが取得できる最高レベルの権限で動作 |
| RunAsAdmin | 管理者権限で動作 |
| ElevateCreateProcess | ERROR\_ELEVATION\_REQUIREDを処理 |

Compatibility Administrator についての詳細は、「5.2 Compatibility Administrator」を参照してください。

##### アプリケーションマニフェストの利用

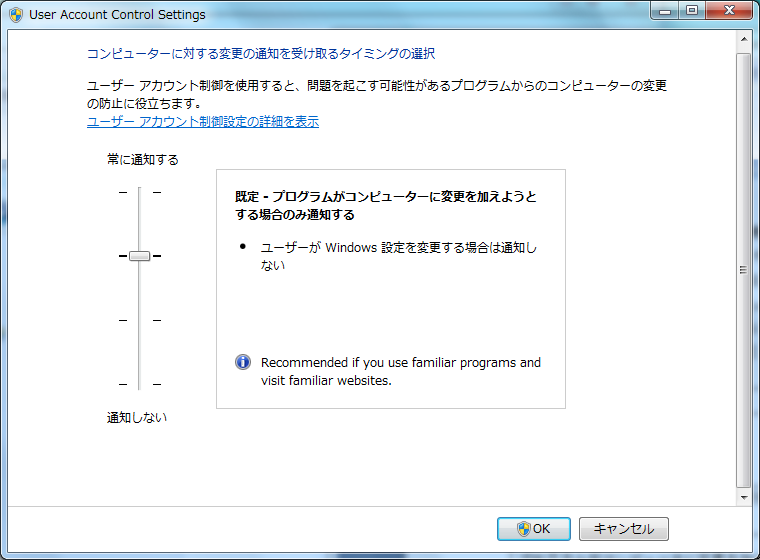
アプリケーションマニフェストを作成することで、アプリケーション開発者が昇格を要求することができます。アプリケーションマニフェストの詳細については、この節の「アプリケーションマニフェストの作成」を参照してください。

### Windows 7 での変更点

**Windows 7 Only**

UAC はセキュリティを高めるための機能なので、その機能を無効にしてしまうことは推奨されません。ただ、Windows Vista では、すべての管理作業で権限昇格ダイアログが表示されるため、煩わしさを感じるユーザーが多かったことも確かです。

そこで、Windows 7 では、UAC のセキュリティレベルを4段階の中から選択できるようになっています。これは、コントロールパネルで　[システムとセキュリティ]－[アクションセンター]－[ユーザーアカウント制御設定の変更]　で設定できます。



* 1. [ 互換性 ] タブ

|  |  |
| --- | --- |
| 設定 | 説明 |
| 常に通知する | 常に UAC の通知をおこないます。これは、Windows Vista でUAC が有効な時と同じ動作です。 |
| プログラムがコンピューターに変更を加えようとする場合のみ通知する | アプリケーションがソフトウェアをインストールしようとしたり、コンピューターに変更を加えようとしたりした場合に、UAC のダイアログを表示します。  ただし、Windows が提供するアプリケーションでは、UAC のダイアログを表示することなく昇格します。  UAC のダイアログの表示時には、デスクトップ全体を薄暗くし、他のアプリケーションを操作することができなくなります。Windows 7はこれが既定値です。 |
| プログラムがコンピューターに変更を加えようとする場合のみ通知する (デスクトップを暗転しない) | アプリケーションがソフトウェアをインストールしようとしたり、コンピューターに変更を加えようとしたりした場合に、UAC のダイアログを表示します。  ただし、Windows が提供するアプリケーションでは、UAC のダイアログを表示することなく昇格します。  UAC のダイアログの表示時には、デスクトップ全体を薄暗くしないため、他のアプリケーションを引き続き操作することができます。 |
| 通知しない | 常に UAC の通知をおこないません。これは、Windows Vista で UAC が無効の時と同じです。  セキュリティ上の理由から、この設定は推奨されません。 |

### UAC 対応アプリケーションを新規に作成するには

新規に作成するアプリケーションは、UAC を考慮して開発します。そして、ユーザーが特別な操作をしなくても、必要に応じて権限を昇格できるようにします。

#### 標準権限でアプリケーションを実行できるようにする

まずは、標準権限で動作できるようにすることを優先します。標準権限で実行できるようにするためには、特に以下の点に注意します。

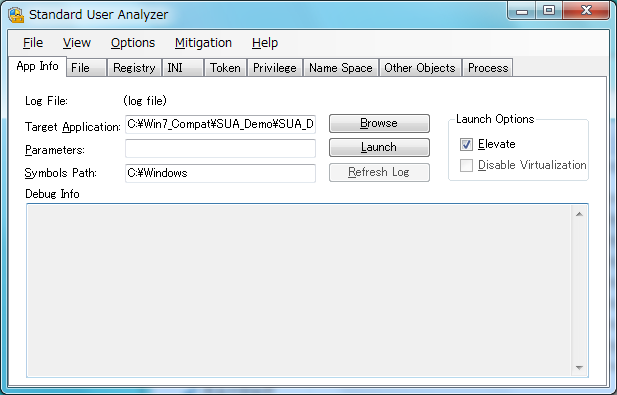
* アプリケーションの実行に必要な情報をファイルに保存するときは、適切な場所を指定します。Windows VistaやWindows 7 では、マシン全体のデータと、ユーザーごとのデータの保存先が明確に区別されています。標準ユーザーは、システム用の領域には書き込みできません。また、他のユーザーのプロファイルにもアクセスすることはできません。詳細は、4.1「リソース管理」を参照してください。
* レジストリへの書き込みをおこなっている場合には、それが適切な場所かどうかを確認します。「HKEY\_LOCAL\_MACHINE」への書き込みは、管理者権限が必要です。
* ユーザーごとの処理は、インストールが終了した後におこなうようにします。インストールは権限の昇格のために、別のユーザーでおこなわれている可能性があります。ユーザーごとの設定が必要なときは、各ユーザーの初回起動時におこないます。
* Microsoft Windows Installer 3.1以降に対応したインストーラを使用します。MSI 3.1 以降のバージョンは、UAC を意識したものになっています。標準権限でのアプリケーションの更新も可能です。

##### Standard User Analyzer

アプリケーションを検証し、昇格が必要かどうかを確認します。これを確認するための最も分かりやすい方法は、標準ユーザーでアプリケーションを実行してみることです。ただ、このやり方は、バグを1つずつ見つけて修正していく必要があるため、時間がかかる可能性があります。

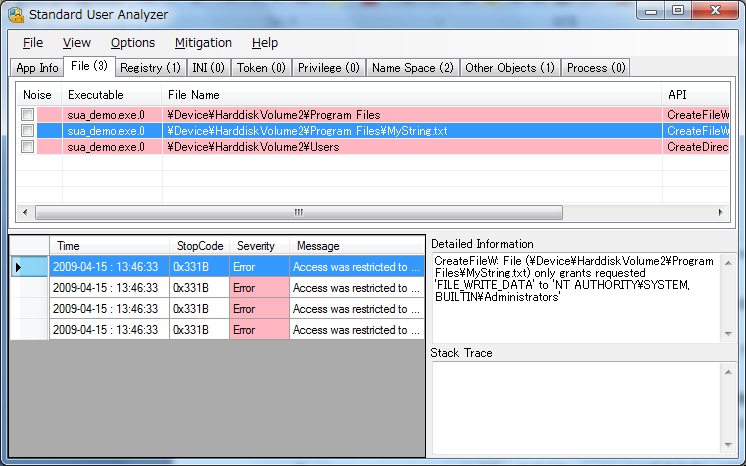
そこで、「Standard User Analyzer 」ツールを使用すると、特権に関する問題を予測・診断することができます。このツールは、ACT ( Application Compatibility Toolkit ) に含まれています。Standard User Analyzer の使用方法は、以下のとおりです。

1. スタートメニューから、[ Standard User Analyzer ] を起動します。
2. [ App Info ] タブの [ Target Application ] で、分析したいアプリケーションのディレクトリを指定します。
3. アプリケーションのパラメータが必要な場合は、[ Parameters ] を設定します。
4. [ Launch Elevated ] チェックボックスを選択し、[ Launch ] ボタンをクリックします。



* 1. Standard User Analyzer の設定画面

1. 「ユーザーアカウント制御」ダイアログボックスが表示されたら、 [はい] をクリックします。
2. ターゲットアプリケーションに対する「ユーザーアカウント制御」ダイアログボックスで、管理者の資格情報を指定し、[はい] ボタンをクリックします。
3. Standard User Analyzerは、アプリケーションを起動し、処理の監視を開始します。ここで、テスト項目に従い、アプリケーションを操作します。操作が終了したらアプリケーションを終了します。
4. Standard User Analyzer により、アプリケーションのログを生成して解析します。(完了には時間がかかる場合があります)   
   ログが生成され、解析されたら、各タブをクリックして発見された問題を確認します。



* 1. Standard User Analyzer の解析結果

#### 権限の昇格への対応

##### 管理者権限が必要なコードを分離する

作成するプログラムが、どんなユーザーをターゲットにしているのかを考えます。アプリケーションが一般ユーザー向けのものであるならば、全ての機能が標準ユーザーで動作するよう、作られていなければなりません。一方、管理者を対象としたアプリケーションで、ほぼ全ての操作に管理者権限が必要ならば、全てのコードが管理者権限で実行されるようにすることが考えられます。標準ユーザーで実行するコードと、管理者権限で実行するコードが混在する場合には、プログラムを分割することを考えます。そして、管理者権限を必要とするプログラムのみを昇格するようにすれば、一部だけ管理者権限で実行するアプリケーションを作成することができます。

ただし、プロセスのトークンは起動時に決定し、変更することはできません。そのため、管理者権限で実行するためには、分離したプロセスを、以下のいずれかの方法を使用して、メインのアプリケーションから呼び出します。

* ShellExecute() もしく は ShellExecuteEx() を使用して、親プログラムから起動する

ShellExecute()もしくは ShellExecuteEx() を使用して分割したプログラムを呼び出します。CreateProcess() では権限の昇格ができないため、「ERROR\_ELEVATION\_REQUIRED」が返されます。

* COM オブジェクトとして分離する

COM オブジェクトを作成するときに、COM昇格モニカを使用して実行するオブジェクトのインスタンスを作成します。これにより、作成したオブジェクトは昇格したプロセスでホストされます。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| COM 昇格モニカについては、以下のMSDN サイトに「CoCreateInstanceAsAdmin()」という名前でサンプルコードが紹介されています。  http://msdn.microsoft.com/en-us/library/ms679687.aspx |

##### 盾 ( シールド) アイコンの配置

盾アイコンは、特権の昇格が必要となることをユーザーに知らせることを目的としています。この盾アイコンは、以下の4種類のアイテム上に表示することができます。

###### アイコン

ファイルのアイコンに盾を表示することができます。アプリケーションマニフェストの「requestedExecutionLevel」の設定が、「requireAdministrator」となっている場合に自動的に盾が表示されます。そのため、アプリケーション開発者がコードから制御したりすることはできません。



* 1. 盾アイコンの表示されたアイコン

###### Button / Hyperlink / Commandlink

Windows VistaやWindows 7 では、Button や Hyperlink、Commandlinkなどのコントロールが拡張されており、盾アイコンを表示することができるようになっています。そのためには、以下のコードのように、BCM\_SETSHIELDメッセージをボタンに送信します。その際にLPARAM を TRUEにするとアイコンが表示され、FALSEにするとアイコンが非表示になります。

SendMessage(hwndButton, BCM\_SETSHIELD, 0, TRUE);

また、commctrl.h にマクロが定義されているため、以下のコードでも盾アイコンを表示することができます。

Button\_SetElevationRequiredState(hwndButton, true);

いずれの場合にも 盾アイコンの表示は、単にアイコンがついているにすぎないため、権限を昇格して実行するためには、呼び出し先のexe などでマニフェストを設定する必要があります。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| **Button**  **Hyperlink**  **Commandlink**   * 1. 盾アイコンの表示されたコントロール |

##### アプリケーションマニフェストの作成

昇格を必要とするアプリケーションでは、アプリケーションマニフェストに管理者権限が必要であることを定義します。Windows VistaやWindows 7では、アプリケーションマニフェストの属性が拡張されていて、「requestedExecutionLevel」要素を使用することで、アプリケーションの実行に必要な権限レベルを設定することができます。「level」属性で設定できる値は以下のとおりです。

<requestedExecutionLevel level="asInvoker" />

|  |  |
| --- | --- |
| 値 | 説明 |
| asInvoker | 親プロセスと同じ権限で動作 |
| highestAvailable | ユーザーが取得できる最高レベルの権限で動作 |
| requireAdministrator | 管理者権限で動作 |

次に、以下の3種類のうちのいずれかの方法でアプリケーションマニフェストを作成します。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| Windows XP SP2 で、「requestedExecutionLevel」などのWindows Vista の拡張機能を持つマニフェストを .exe ファイルまたは .dll ファイルに追加すると、コンピューターが再起動することがあります。この問題に関しては、修正プログラムが提供されています。詳細は、以下のサイトを参照してください。  http://support.microsoft.com/kb/921337/ja |

###### 埋め込みマニフェストを作成する

アプリケーションマニフェストを、アプリケーションファイル ( exeや dll など ) とは別に、Xml形式のファイルで提供する方法です。アプリケーションと同じディレクトリに、「アプリケーション名.exe.manifest」という名前で保存すると、アプリケーションの起動時に、マニフェストファイルが自動的に読み込まれます。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| ただし、この情報はキャッシュされているため、一度実行した後でマニフェストファイルを配置すると、マニフェストが有効にならないことがあります。更新されない場合には、以下のいずれかの対応をとる必要があります。  ① 不定期のキャッシュの更新を待つ  ② プログラムをリビルドする  ③ プログラムのパスを変える (別のフォルダーにコピーするなど) |

以下に具体的な手順を示します。

1. テキストエディタなどでマニフェストを作成します。ファイル名は「*アプリケーション名*.exe.manifest」にしなければなりません。以下はマニフェストの例です。「level="requireAdministrator"」を設定することで、管理者権限を要求しているのが分かります。

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<assembly xmlns="urn:schemas-microsoft-com:asm.v1"

manifestVersion="1.0">

<trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">

<security>

<requestedPrivileges>

<requestedExecutionLevel

level="requireAdministrator"/>

</requestedPrivileges>

</security>

</trustInfo>

</assembly>

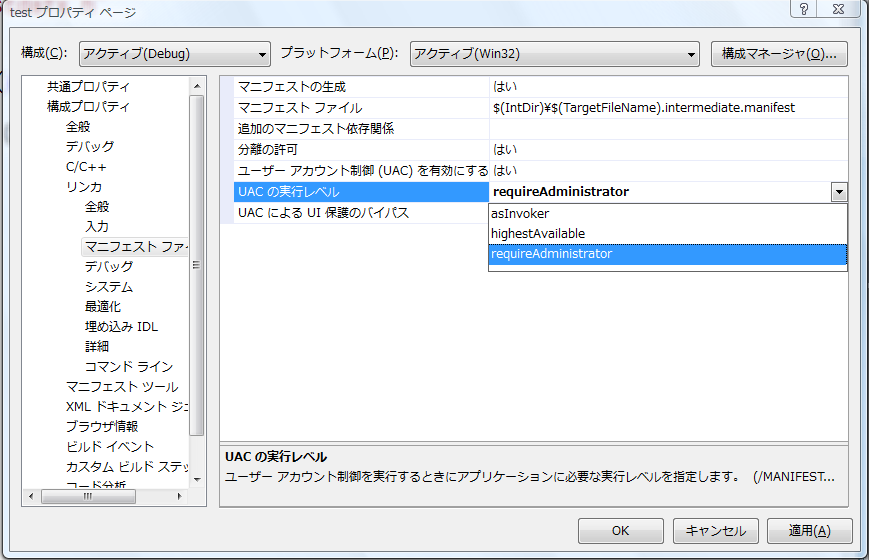
1. 手順①で作成したファイルを、アプリケーションと同じディレクトリに保存します。
2. アプリケーションを実行します。実行時にPE ( Portable Executable )イメージにマニフェストが自動的に埋め込まれます。その結果、起動時に権限の昇格が求められます。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| mt.exeコマンドを使用すると、アプリケーションをビルドした後に、アプリケーションマニフェストを埋め込むことができます。使用方法は以下の通りです。  mt.exe –manifest *マニフェストファイルパス* –outputresource:*アプリケーションパス* |

###### C / C++プロジェクトでマニフェストを作成するには

アプリケーションマニフェストを、アプリケーションファイル ( exeなど) に埋め込む方法です。Visual Studio 2008 のGUI 環境を利用して設定することができます。

1. Visual Studio 2008 でプロジェクトを作成します。
2. プロジェクトのプロパティを開きます。
3. [ 構成プロパティ ]－[ リンカ ] － [ マニフェストファイル ] を選択し、[ UAC の実行レベル ] で、「requireAdministrator」を選択します。



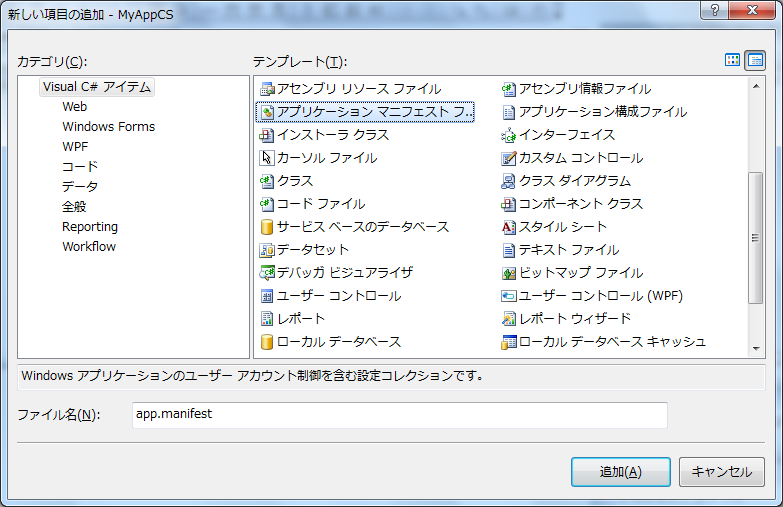
* 1. マニフェストファイルパスの設定

1. アプリケーションをビルドします。

###### マネージコード (C#、Visual Basic ) でマニフェストを埋め込むには

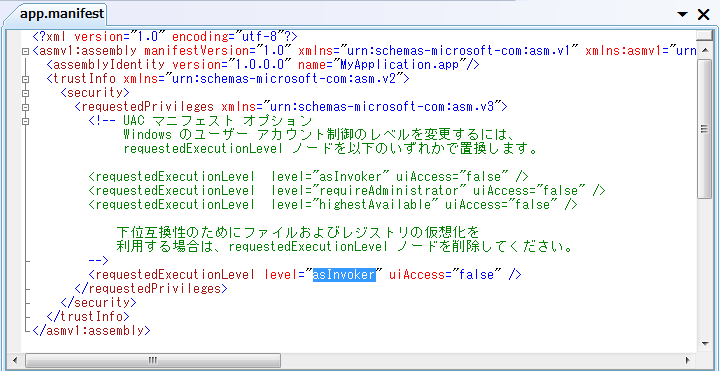
アプリケーションマニフェストを、マネージコードに埋め込む方法です。Visual Studio 2008でマニフェストを追加することができます。

1. Visual Studio 2008 でプロジェクトを作成します。
2. [プロジェクト]－[新しい項目の追加]で、「アプリケーションマニフェスト」を選択します。



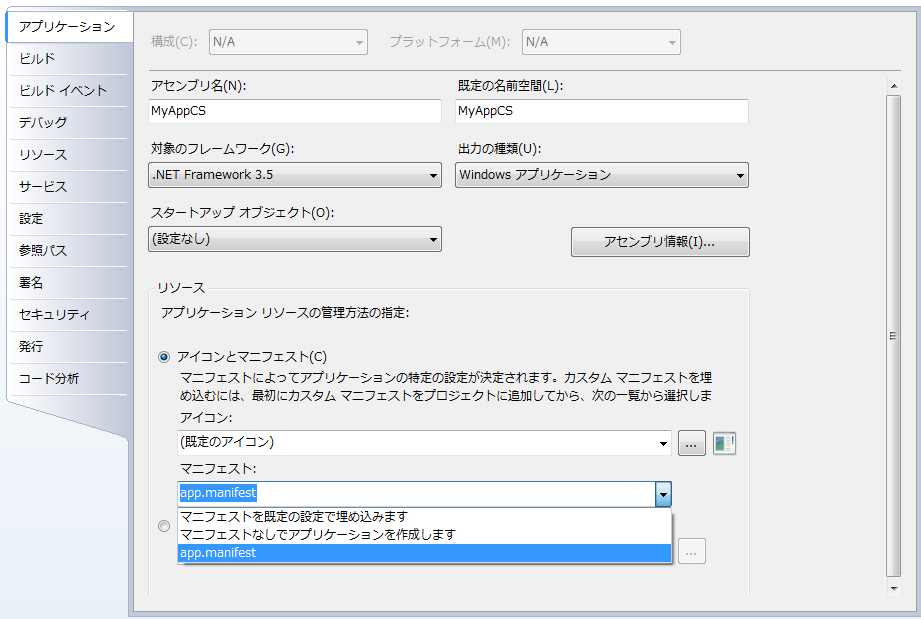
* 1. アプリケーションマニフェストテンプレート

1. 追加された「app.manifest」ファイルで、「level="requireAdministrator"」を設定します。



* 1. app.manifestファイル

1. プロジェクトのプロパティを開きます。
2. [プロジェクト] タブで、追加したマニフェストファイルが設定されていることを確認します。



* 1. マニフェストファイルの設定

1. アプリケーションをビルドし、実行します。その結果、起動時に権限の昇格が求められます。

##### バイナリへの署名の追加

アプリケーションに署名を追加することで、権限の昇格ダイアログが表示されたときに、帯の色を黄色ではなく、青で表示させることができます。

署名については、「3.4 ファイルへの署名」を参照してください。

### Windows 7での既存アプリケーションへの対応

Windows VistaやWindows 7 では、既存のアプリケーションが標準ユーザーで動作できるように、いくつかの機能を提供しています。この機能により、多くの既存のアプリケーションは、コードを書き換えることなくWindows VistaやWindows 7上で実行できるようになっています。

#### ファイルとレジストリの書き込み

アプリケーションから、以下のような標準ユーザーに書き込み権限のない領域 ( システム領域など ) への書き込みを要求すると、ユーザーごとの領域へリダイレクトされます。

|  |  |
| --- | --- |
| 書き込み制限のある マシンごとのストア | リダイレクト先のユーザーごとのストア |
| %ProgramFiles% | %LocalAppData%\VirtualStore\Program Files |
| %WinDir% | %LocalAppData%\VirtualStore\Windows |
| HKLM\Software | HKCU\Software\Classes\VirtualStore\Machine\Software |

#### インストーラの自動検出

Windows VistaやWindows 7 では、アプリケーションをインストールするには管理者権限が必要です。そのため、インストールプログラムを実行するときには、昇格が必要となります。そこでWindows VistaやWindows 7 では、経験則に基づき特定のパターンに合致するプログラムを、インストーラとして検出する機能が追加されています。インストーラとして検出されると、自動的に権限昇格ダイアログが表示され、管理者権限で実行することができます。たとえば、ファイル名やバージョン情報に「setup」、「update」、「install」という文字列が含まれている場合、Windows VistaやWindows 7はアプリケーションの実行時にインストーラとみなし、実行には昇格が必要であると認識します。そこで自動的に権限昇格用のダイアログボックスが表示されます。ここで昇格すれば、問題なくアプリケーションをインストールすることができます。ただし、自動検出するにはインストーラが以下の要件を満たしている必要があります。

* 32ビットプログラムである
* 対話型のプログラムである
* マニフェストにより実行権限が指定されていないプログラムである

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| ユーザーごとの領域へのリダイレクト機能や、インストーラの自動検出の機能は、既存のアプリケーション向けの一時的な互換性機能です。将来はサポートされなくなる可能性もあるため、新規にWindows 7 用のアプリケーションを作成する場合には、適切なストアパスを指定したり、Windows インストーラを使用したりすべきです。 |

### 互換性を検証するには

#### 標準ユーザーで検証

昇格が必要かどうかを確認するため、標準ユーザーで実行しテストします。そして、すべての機能が昇格なしに実行できることを確認します。しかし、この方法ではアプリケーションの一部で問題が発生した場合、そのアプリケーションが終了することがあります。よって、バグを1つずつ見つけて修正していかなければならないため、時間がかかります。「Standard User Analyzer 」ツールを使用することで、効率よく特権に関する問題を発見することができます。

#### 問題が発生しやすい場面

以下のような現象が起こった場合、管理者権限がないために実行が失敗している可能性があります。

* アクセス拒否のエラーが表示され、操作を続行できない
* 「管理者権限が必要」というエラーが表示され、操作を続行できない
* 何も起こらず、操作に失敗する
* 権限昇格ダイアログが起動のたびに表示される

このような問題が発生する場合には、そのアプリケーションを管理者として実行するか、エラーが起こるコードの前で、UAC の権限昇格ダイアログを表示し、管理者で実行するなどの修正が必要です。

#### 問題の切り分け

以下のユーザーを使用し、①～④の順番で、アプリケーションの互換性をテストします。

1. 標準ユーザー
2. 管理者ユーザー(標準権限)
3. アプリケーションのコンテキストメニューで、「管理者として実行」
4. ビルトインの Administrator

テストの結果、① や ②で問題が起こり、③ や ④では起こらないという場合は、UAC に起因している可能性が高いといえます。また、別のウィンドウとのやり取りにのみ問題があるという場合は、UIPI の問題である可能性があります。さらに、IE ( Internet Explorer ) がアプリケーションに関連している場合、保護モードのIE による問題の可能性もあります。UIPI については「4.4 ユーザーインターフェイス特権の分離 ( UIPI )」、保護モードの IE については「4.5 IE のセキュリティ機能」を参照してください。

### UAC事例

ここでは、UAC を使用する上で発生する可能性の高い問題と対応策を5つ紹介します。

|  |  |  |
| --- | --- | --- |
| 1. 管理者権限のチェック | | |
| 対象 | ユーザーの権限をチェックしているアプリケーション | |
| シナリオ | アプリケーションの起動時 ( 多 ) やインストール時 ( 少 ) | |
| 現象 | 管理者グループのユーザーであるにもかかわらず、「管理者権限が必要です」というエラーが表示される | |
| 原因 | UAC の「トークンフィルタ」機能により、プログラムが標準権限で動作している | |
| 解決方法 | ユーザー・ システム管理者向けの回避方法 | 権限を昇格して実行する  「ForceAdminAccess」や「ProtectedAdminCheck」互換フィックスを使用してアプリケーションを実行する   |  | | --- | | **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** | | これらの互換フィックスを使用すると、権限は昇格しませんが、権限チェックに対して「管理者として実行している」と偽ることができます。このため、権限チェックを通過することができます。ただし、実際に管理者権限が必要な処理を呼び出すとエラーが発生します。  なお、「ForceAdminAccess」互換フィックスに関する詳細は、以下のサイトを参考にしてください。  http://technet.microsoft.com/en-us/library/cc766024.aspx | |
| 開発者向けの対応方法 | 以下の手順で既存のアプリケーションを変更します。   1. 管理権限が必要な処理は別プログラムとして分離し、元のプログラムは標準ユーザーで動作させる 2. 分離したプログラムの実行権限を、アプリケーションマニフェストで指定(level = “requireAdministrator”) する 3. 必要なときにShellExecute() を使用して、分離したプログラムを呼び出す |

|  |  |  |
| --- | --- | --- |
| 1. 自動更新の失敗 ( インプロセス ) | | |
| 対象 | メインプログラム内に更新機能を持つアプリケーション | |
| シナリオ | アプリケーションの自動更新 | |
| 現象 | アプリケーションの自動更新が失敗する | |
| 原因 | UAC の「トークンフィルタ」機能により、自動更新機能を含むプログラム全体が標準ユーザーで動作している | |
| 解決方法 | ユーザー・ システム管理者向けの回避方法 | 権限を昇格してメインアプリケーションを実行する |
| 開発者向けの対応方法 | 以下のいずれかの対策をとります。   * 管理権限が必要なアップデート機能は、別プログラムとして分離し、マニフェストにより実行権限を指定 (level = “requireAdministrator” ) し、元のプログラムからShellExecute() で呼び出す * MSI を使用する |

|  |  |  |
| --- | --- | --- |
| 1. 自動更新の失敗 ( アウトプロセス ) | | |
| 対象 | メインのプログラムとは別に、更新を行うプログラムを持つアプリケーション | |
| シナリオ | アプリケーションの自動更新 | |
| 現象 | アプリケーションの自動更新に失敗する | |
| 原因 | CreateProcess() を使用してアップデータを呼び出している | |
| 解決方法 | ユーザー・ システム管理者向けの回避方法 | * メインのプログラムを管理者権限で実行する * 「ElevateCreateProcess」互換フィックスをメイン側に適用する  |  | | --- | | note**Note**: | | 「ElevateCreateProcess」互換フィックスを使用すると、メインから呼び出すアップデータのみが権限昇格します。詳細は以下のサイトを参考にしてください。  http://technet.microsoft.com/en-us/library/cc722422.aspx |  * プログラム互換性アシスタントを利用し、自動的に「ElevateCreateProcess」を適用する |
| 開発者向けの対応方法 | 以下のいずれかの対策をとります。   * アップデータの実行権限をマニフェストで指定 (level = “requireAdministrator”)し、メインのプログラムから ShellExecute()で呼び出す * MSI を使用する |

|  |  |  |
| --- | --- | --- |
| 1. インストーラとして検出しない | | |
| 対象 | すべてのアプリケーション | |
| シナリオ | アプリケーションのインストール | |
| 現象 | インストーラが自動的に権限昇格されず、処理が失敗する | |
| 原因 | インストーラが UAC のインストーラ検出機能により検出されていない | |
| 解決方法 | ユーザー・ システム管理者向けの回避方法 | インストーラを権限昇格して実行する |
| 開発者向けの対応方法 | 以下のいずれかの対策をとります。   * インストーラの実行権限をマニフェストで指定 (level = “requireAdministrator”)する。 * MSI を使用する |

|  |  |  |
| --- | --- | --- |
| 1. 誤ってインストーラとして検出 | | |
| 対象 | すべてのアプリケーション | |
| シナリオ | アプリケーションの起動時 | |
| 現象 | アプリケーションの起動時に、意図せず自動的に権限昇格される | |
| 原因 | プログラムが UAC のインストーラ検出機能により、インストーラとして検出されている | |
| 解決方法 | ユーザー・ システム管理者向けの回避方法 | 「SpecificNonInstaller」互換フィックスを使用し、インストーラとして検出されないようにする   |  | | --- | | **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** | | 「SpecificNonInstaller」互換フィックスに関する詳細は、以下のサイトを参考にしてください。  http://technet.microsoft.com/en-us/library/dd638326.aspx | |
| 開発者向けの対応方法 | マニフェストで実行権限を指定 (level = “asInvoker”)する。 |

## ユーザーインターフェイス特権の分離 ( UIPI )

### UIPI (User Interface Privilege Isolation)とは

整合性レベルにより、プロセスごとにアクセスできるリソースのレベルを決定することができるため、攻撃によるデータの変更や破壊、悪意のあるコードのインストールの可能性を軽減することができます。

しかし、プロセス間でWindows Messaging ( SendMessage() )などを利用して、自由に通信できてしまうと問題が起こります。低い整合性レベルのプロセスが、高い整合性レベルのプロセスにメッセージを送信することができるとなると、低いほうのプロセスから、高いほうのプロセスに対して、悪意のあるDLLが注入され、そのコードが管理者権限で実行される危険性があるからです。

そこで、Windows VistaやWindows 7 では、UIPI が既定で有効になっています。UIPIは 上位のプロセスを下位のプロセスから保護する機能です。UIPI により、下位のプロセスから上位のプロセスへの通信は許可されません。たとえば、「低IL」のプロセスから「中IL」や「高IL」への通信は失敗しますし、「中IL」から「高IL」への通信も失敗します。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| 許可されていない通信は、下位のプロセスから上位のプロセスへの通信に限ります。同レベルのプロセス間通信は許可されています。たとえば、「中IL」のプロセスから、他の「中IL」のプロセスへの通信は可能です。また、上位から下位のレベルへの通信も問題ありません。たとえば、「高IL」のプロセスから「中IL」のプロセスへの通信も問題なくおこなえます。 |

### UIPI に対応するには

#### 上位レベルのプロセスと通信しない

下位レベルのプロセスから、上位IL のプロセスを呼び出さないようにするのが基本です。

しかし、下位レベルのプロセスから、上位レベルのプロセスをどうしても呼び出さなければならないときには、以下の2つの方法が考えられます。

#### 権限を昇格し、自身も上位権限になる

下位レベルのプロセスを、通信したいほうのレベルに合わせて実行します。たとえば、「高IL」のプロセスと通信したいなら、自身も「高IL」で実行します。

「中IL」や「高IL」で実行する方法は、UAC と同じです。つまり、標準ユーザーとして実行しているときには「中IL」が使われ、昇格ダイアログボックスを使って、管理者に昇格すると「高IL」で実行します。自身を昇格し、「高IL」で実行すれば、他のプロセスと通信することができます。

#### 実行権限を昇格せずに、上位の権限のプロセスとやりとりする

昇格せずに、上位権限のプロセスと通信することもできます。このためには、アプリケーションマニフェストでの設定が必要です。以下にサンプルを示します。ポイントは、「<requestedExecutionLevel level="asInvoker" uiAccess="true"/>」の設定です。「level="asInvoker」の設定により、このプロセスは標準権限で実行されることが宣言されています。しかし、「uiAccess="true"」の設定により、他のプロセスと通信するときには、「高IL」として振舞うことができます。そのため、上位権限のプロセスとも通信をすることができるようになります。

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<assembly xmlns="urn:schemas-microsoft-com:asm.v1"

manifestVersion="1.0">

<trustInfo xmlns="urn:schemas-microsoft-com:asm.v2">

<security>

<requestedPrivileges>

<requestedExecutionLevel

level="asInvoker" uiAccess="true"/>

</requestedPrivileges>

</security>

</trustInfo>

</assembly>

ただし、「uiAccess="true"」を使用して上位権限のプロセスとやり取りするアプリケーションは、以下の2つの条件を満たす必要があります。

* 署名付きのプログラムである
* %WinDir% または %ProgramFiles% フォルダーに保存されている

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| 高IL のプロセスと通信するためには、Administrators グループのメンバーでログオンしていなければなりません。一般ユーザーとしてログオンしている場合には、これらの設定をしても 高IL のプロセスと通信することはできません。 |

### UIPIによる問題を検証するには

下位のプロセスから、上位権限のプロセスへの通信で失敗している場合には、以下のような現象が起こります。

* 上位権限のプロセスのハンドルの検証に失敗する
* 上位権限のウィンドウへのSendMessage()やPostMessage()に失敗する

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| この場合、エラーは発生しません。下位のプロセスからは送信されたように見えますが、上位のプロセスで破棄されます。ただし、GetLastError() には ERROR\_ACCESS\_DENIED(5) が返ります。 |

* 上位権限のプロセスへのスレッドフックやジャーナルフックに失敗する
* 上位権限のプロセスへのDLLの注入に失敗する

このような問題が発生し、しかも以下に該当する場合、UIPIの問題である可能性が高いといえます。

* 権限が同じプロセス間では問題が発生せず、上位権限との間で問題が発生する。
* 「管理者として実行」やビルトインの Administrator で起動した場合には問題が発生しない。

以上のような問題に対応するためには、「権限を昇格し、自身も上位権限になる」や、「実行権限を昇格せずに、上位の権限のプロセスとやりとりする」ので紹介した手順を実行します。

## IE のセキュリティ機能

### Internet Explorer ( IE ) の保護モード

#### Windows XP およびIE6

Windows XP と IE (Internet Explorer) 6 までの環境では、さまざまなコードが IE から実行される可能性がありました。特に管理者権限でアクセスしている場合には、攻撃によるデータの書き込みや変更、破壊がおこなわれたり、悪性のプログラムがインストールされたりする可能性があります。

画像のダウンロード

設定の保存と変更

*悪意のあるプログラム*

*のインストール*

ActiveX やツール

バーのインストール

Internet Explorer 6

管理者権限で動作

ユーザー権限

HKCU

%UserProfile%

管理者権限

HKLM

%ProgramFiles%

％WinDir％

Temporary Internet Files

悪意のあるプログラム

のインストール

Web コンテンツの

キャッシュ

* 1. Windows + IE 6

#### Windows 7 および IE8

##### 悪意のある攻撃からシステムを保護

Windows Vista の IE 7またはIE8、Windows 7 のIE 8では、IE は「保護モード」で実行されます。これは IE が「低 IL」で実行されるということです。つまり、厳しく制限された権限で実行されるため、攻撃によるデータの書き込み、変更、または破壊や、悪意のあるコードのインストールの可能性を大幅に軽減できます。また、認証なしに自己インストールをおこなう悪意のあるコードからシステムを保護できます。保護モードの IE は、UAC、UIPI、整合性レベルと連動します。保護モードの IE は、低 IL で動作しているため、ユーザーのプロファイルやシステムファイル、レジストリキーへの書き込みは許可されません。安全な領域であるインターネット一時ファイルフォルダー、履歴フォルダー、Cookie フォルダー、お気に入りフォルダー、一時ファイルフォルダーなどへの書き込みのみが可能です。

#### 互換性レイヤ

アドオンなどによる標準権限の領域 ( ユーザープロファイルの ドキュメントフォルダーや HKCUレジストリハイブ ) への書き込み処理は、以下の安全な領域へリダイレクトされます。これにより、互換性を保っています。

|  |  |
| --- | --- |
| ファイル | %LocalAppData%\Microsoft\Windows\Temporary Internet Files\Virtualized |
| レジストリ | HKCU\Software\Microsoft\Internet Explorer\InternetRegistry |

#### アクセス制御メカニズム

保護モードの IE は、制限された権限で動作しています。しかし、ブローカープロセスを使用することで、上位権限を取得することもできます。ブローカープロセスには、以下の2種類があります。

|  |  |  |  |
| --- | --- | --- | --- |
| ブローカーの種類 | 権限 | 整合性レベル | プロセス |
| 管理者ブローカー | 管理者権限 | 高IL | IEInstal.exe |
| ユーザーブローカー | 標準権限 | 中 IL | IExplore.exe |

ツールバーの  
インストール

ActiveXの  
インストール

保護モードのInternet Explorer ７

(低 IL)

ユーザー権限

HKCU

%UserProfile%

管理者権限

HKLM

%ProgramFiles%

％WinDir％

信頼性の低い権限

Temporary Internet Files  
\Virtualized

アドオン設定の保存

アプリケーション

情報サービス

**管理者ブローカー**

**IEInstal.exe(高IL)**

画像のダウンロード  
設定の保存と変更

整合性のメカニズム

**ユーザーブローカー**

**IExplorer.exe(中IL)**

保護モードのInternet Explorer 8

(低 IL)

互換性レイヤ

* 1. Windows Vista / Windows 7 + 保護モードのIE 8

図4-20のように、ユーザープロファイルにアクセスしたい場合には、標準権限が必要です。そのため、「ユーザーブローカー」を使用します。また、Active Xコントロールをインストールしたい場合は、管理者権限が必要です。そこで、「管理者ブローカー」を使用します。図4-21のように、上位にある 2 つの特権ブローカーによって、IE はユーザーの同意を得て上位処理を実行できます。

中IL(ユーザー権限)

高IL(管理者権限)

低IL(信頼性の低い権限)

※下位レベルへの書き込みアクセスは可能

**プロセス**

インストーラなど

権限昇格の確認(UAC)

ノートパッドなど

ユーザーブローカー

保護モードのIE

**オブジェクト**



管理者ブローカー

* 1. 保護モードのIEのアクセス制御メカニズム

#### IEの保護モードによる問題を検証するには

##### 互換性への影響

IE の保護モードにより、既存アプリケーションでは、以下のような問題が発生することがあります。

* アドオンなどで上位権限のリソースにアクセスできない
* アドオンなど、IE にロードされたモジュールの設定情報を、IE 以外のプロセスと共有できない

これらの問題が発生する場合、以下の回避方法が考えられます。

##### 問題の回避方法

* 「管理者として実行」

IE を管理者として実行します。スタートメニューで IE を選択する際に、マウスを右クリックして、「管理者として実行」を選びます。

* 保護モードを無効化

IE の保護モードを無効にします。これは、IE のインターネットオプションで設定することができます。

##### 問題の切り分け

保護モードを無効にして問題が発生しなくなる場合、保護モードの可能性が高いといえます。また、保護モードを無効にしても問題が発生する場合、UAC の問題である可能性があります。

#### IEの保護モードに対応するには

* 適切な API を使用
* IEIsProtectedModeProcess()

保護モードで動作しているかどうかの判定することができます。

* IEGetWriteableHKCU()

HKCU 下の書き込み可能なレジストリを取得することができます。

* ファイル・レジストリの使用

以下のフォルダーやレジストリは、低IL のまま利用することができます。

* %UserProfile%\AppData\LocalLow
* %Temp%\Low
* HKCU\Software\AppDataLow

上記以外のフォルダーに保存したい場合には、以下の手順が必要です。

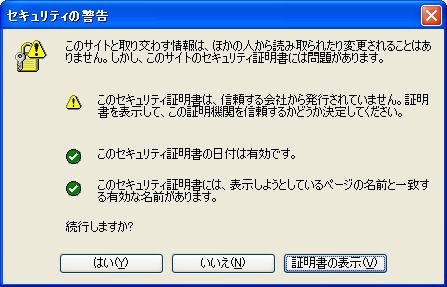
1. 「%UserProfile%\AppData\LocalLow」フォルダーに、暫定版のファイルを作成します。
2. IEShowSaveFileDialog() を実行し、「名前を付けて保存」ダイアログを表示します。これにより、ユーザーはファイルの保存先を決定することができます。
3. IESaveFile() を使用して、ユーザーが指定した場所にファイルをコピーします。
4. この手順を使用した場合には、ユーザーブローカーにより、中IL の権限が取得されます。

### デジタル証明書とHTTPS

IE7 と IE 8では、Web サイトから以下のようなデジタル署名が提示された場合、HTTPS へのナビゲーションをブロックします。

* 現在のURL hostname とは異なる hostname が設定されているデジタル証明書
* 信頼されていないルート証明機関から発行されたデジタル証明書
* 期限切れのデジタル証明書
* 無効なデジタル証明書

たとえば、信頼されていない証明機関から発行されたデジタル証明書がサイトから提示されたとき、IE 6では、図4-22のようなダイアログボックスが表示されました。ここで、「はい」ボタンをクリックすると、HTTPS を使用してページが表示されました。



* 1. IE 6でのセキュリティ警告

しかし、IE7 とIE 8では同じ問題が発生すると、ページがブロックされるため、表示することはできません。このような問題が発生する場合には、デジタル証明書や Web サーバーの環境を変更しなければなりません。以下に対応例を示します。

* デジタル証明書の FQDN（Fully Qualified Domain Name） と、保護されたページの FQDN を一致させます。たとえば、「secure.example.com 」Webサーバー上のデジタル証明書のhostnameが「www.exemple.com 」と設定されている場合は、両者が一致しないのでエラーになります。
* 信頼された証明機関から発行されたデジタル証明書を使用します。IE のインターネットオプションで、信頼された証明機関の一覧を確認したり、追加したりすることができます。

## Windows リソース保護 ( WRP )

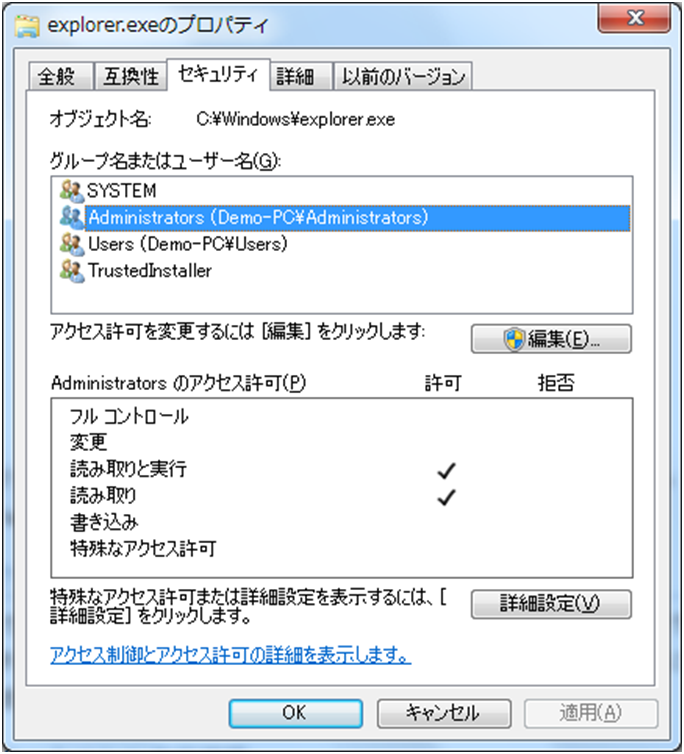
### WRP とは

WRP とは、Windows のリソースやコンポーネントを、読み取り専用で保護する機能です。OSが使用しているシステムファイルやフォルダー、レジストリなどが、故意もしくは偶発的に不正に変更されてしまうと、システムが不安定になるなど、信頼性が損なわれる可能性があります。

そこで、これらのリソースに対しては、通常のユーザーからは変更されないよう読み取り専用として設定されます。これは、ビルトインのAdministrator、Administrators、System アカウントでさえも例外ではありません。図4-23は、「explorer.exe」ファイルのアクセス許可を示しています。Windows VistaやWindows 7では、管理者やシステムアカウントに対してさえも、「読み取りと実行」権しか与えられていないのが分かります。そのため、たとえ管理者でも上書き保存や、削除はできません。代わりに、「TrustedInstaller」ユーザーに対しては、「フルコントロール」権が許可されています。このユーザーは、「Windows モジュールインストーラサービス」で使用されるユーザーアカウントです。

**Windows XP**

**Windows 7**



* 1. Windows XPとWindows Vista / Windows 7のシステムファイルへのアクセス許可

保護されるリソースの一覧は、Microsoft の Web サイト(http://msdn2.microsoft.com  
/en-us/library/aa382530.aspx ) を参照してください。

### 既存アプリケーションへの影響と回避策

WRP によって保護されたシステムファイルやレジストリに変更処理を試みると、アクセス拒否エラーが表示されます。アプリケーションのインストール時などに、エラーが起こることが多いと予想されます。

#### 既存アプリケーションへの回避策

以下の3つの条件を満たす場合、アクセス拒否エラーの表示を OS 側で自動的に制御します。そのため、エラーメッセージは表示しません。

* インストーラの実行権限がアプリケーションマニフェストで定義されていない
* 管理者権限でアプリケーションが実行されている
* WRP で保護されたリソースの変更や作成、削除によるエラーである

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| エラーが表示されないため、コードは成功したかのように見えますが、WRP リソースへの変更はおこなわれません。 |

### WRP対応アプリケーションの作成

#### 独自のプログラムでシステムファイルの更新をおこなわない

Microsoft が提供している再配布パッケージや、サービスパック等以外でシステムファイルやコンポーネントのインストールや更新をおこなわないようにします。それ以外のプログラムでの変更は失敗します。これは管理者で実行しても同様です。また、再配布パッケージを分解したり、再パッケージ化したりして配布しようとしても失敗します。

#### WRPにより保護されているかどうかを判定するには

ファイルのプロパティや、レジストリエディタを使ってアクセス許可を確認することができます。WRPにより保護されている場合には、「TrustedInstaller」にフルコントロール、Administrators や SYSTEM 、およびUsersには読み取りと実行権しか与えられていません。

また、アプリケーションの中から、WRP により保護されているかどうかを確認するには、以下の API を使用します。なお、Windows インストーラは、この操作を自動的におこなっています。

|  |  |
| --- | --- |
| ファイル | SfcIsFileProtected() |
| レジストリ | SfcIsKeyProtected() |

#### レジストリキーへのアクセス拒否エラー

WRP で保護されるキーの作成や修正、削除をおこなうと、アクセス拒否エラーが通知されます。WRP のために発生したアクセス拒否エラーへの対処方法は、このエラーがアプリケーションに与える影響によって、以下の2種類が考えられます。

* アプリケーションが既存のキーや値への書き込みをおこなっている場合、エラーメッセージを無視することができます。
* キーや値が存在しない場合は、開発者としてアプリケーションへの影響を判断する必要があります。このキーがアプリケーションを正常に実行するために必要かどうかを考え、必要でない場合は、エラーを無視することができます。必要である場合は、アプリケーションを再設計して代替キーに書き込むなどの対応策が必要となります。

## セッション 0 の分離

### セッション0の分離とは

Windows XP、Windows Vista やWindows 7 では、ユーザーの切り替え機能をサポートしているため、複数のユーザーが同時に１台のコンピューターにログオンできます。ユーザーは Windows やプログラムを終了せずに、セッションを切り替えることができます。たとえば、ユーザーAがログオンしてWordでドキュメントの編集作業をおこなっている時、ユーザーBが電子メールをチェックする必要があるなら、ユーザーAのWordは実行したまま、ユーザーBがログオンして電子メールアプリケーションを起動することができます。そして、電子メールのチェックが済んだら、再びユーザーAは、Wordが実行されたセッションに戻ることができ、ドキュメントの編集作業を続けることができます。

図4-24 にもあるように、Windows XPやWindows Server 2003、およびそれ以前のWindows では、すべてのサービスはログオンした最初のユーザーと同じセッションで実行されます。このセッションは、セッション 0 と呼ばれます。ただ、サービスをユーザーアプリケーションと同じセッションで実行することは、セキュリティ上危険です。通常、サービスはユーザーアプリケーションより高い権限で実行されているため、悪意のあるプログラムが、自身の権限レベルを昇格させる手段に利用する可能性があるからです。

そこで、Windows Vista と Windows 7 では、セッション 0 ではサービスのみを実行するよう分離しました。これにより、セッション 0 とユーザーアプリケーションは対話することができなくなるため、セキュリティ上の危険が軽減されます。つまり、セッション 0では、システムプロセスとサービスのみを実行します。そして、最初のユーザーはセッション 1 にログオンし、その後のユーザーはそれに続くセッションにログオンします。

**Windows XP**

セッション0

サービス1

アプリケーション1

サービス2

アプリケーション2

サービス3

アプリケーション3

セッション1

アプリケーション4

アプリケーション5

アプリケーション6

セッション2

アプリケーション7

アプリケーション8

アプリケーション9

**Windows Vista、7**

セッション0

サービス1

サービス2

サービス3

セッション1

アプリケーション4

アプリケーション5

アプリケーション6

セッション2

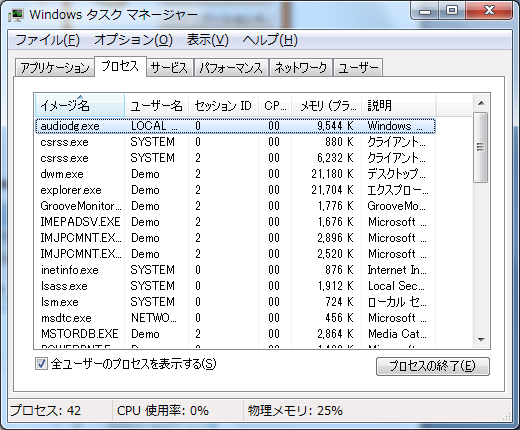
アプリケーション7

アプリケーション8

アプリケーション9



* 1. Windows XPとWindows Vista / Windows 7のセッションの違い



* 1. タスクマネージャーによるセッションIDの表示

### 互換性への影響と検証のポイント

#### 影響を受けるプログラムの例

ユーザーがセッション 0 にいると決めつけているドライバーやサービスなどは、セッション 0の分離により、問題が起こる可能性があります。たとえば、以下のようなサービスが考えられます。

* ユーザーインターフェイス ( UI ) を作成しているサービス

セッション 0 に対してUIが作成されてしまうため、別のセッションを利用するユーザーからレスポンスを受信することはできません。

* ウィンドウメッセージを送信しているアプリケーションやサービス

セッションが異なるため、ユーザーがメッセージを受け取ることはできません。

* 画面のプロパティ情報を取得しているサービス

セッション 0 にグラフィックス能力がないため、正しい情報を得ることができません。

* ローカルオブジェクトでサービスと同期をとるアプリケーション

セッションが異なるため、ローカルオブジェクトで同期をとることはできません。

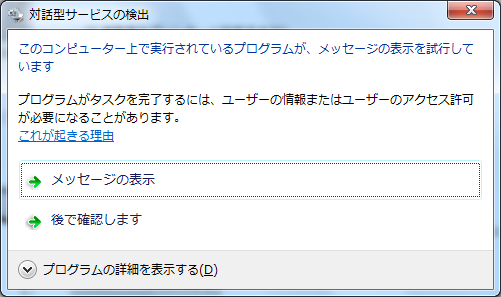
影響を受けるプログラムの具体例は、以下のようなものがあります。

* Print Spooler サービスにロードされるプリンタドライバー
* User-Mode Driver Framework ( UMDF ) に準拠したドライバー

これらのドライバーは、セッション 0 のプロセスでホストされています。

#### 対話型サービス検出サービス

互換性を維持するため、Windows VistaやWindows 7 では「対話型サービス検出」サービスを提供しています。このサービスは、セッション 0 へのダイアログやウィンドウの作成が検出されると、別セッションのユーザーに通知し、ユーザーのデスクトップに図4-26のようなダイアログボックスを表示します。ここで、[ 要求の確認 ] ボタンをクリックすると、セッション 0 に切り替わるため、サービスからのメッセージを読んだり、ダイアログなどを処理したりことができます。「対話型サービスダイアログの検出」ダイアログが表示された後、処理がおこなわれないと、5分以内にユーザーに再度通知されます。



* 1. 「対話型サービスの検出」ダイアログ

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| この機能はあくまで、既存サービスとの互換性のために用意された一時的な回避策です。新たにサービスを提供する際には、この機能に依存せず、正しい対策をとることが重要です。正しい対策については、この節の「セッション0の分離への対応」を参照してください。 |

#### セッション 0 の分離問題を検証するには

問題がセッション 0 の分離に起因するものなのかどうかを検証するためには、以下の動作を確認します。

* 「対話型サービスダイアログボックスの検出」ダイアログが表示された場合

セッション 0 へのウィンドウ作成が試みられたことを意味します。セッション 0 の分離の影響を受けている可能性が高いです。

* Windows XP で、2人目以降のログオンユーザーで問題が発生

Windows XP のユーザー切り替え機能を使って実行し、1人目のログオンユーザーでは問題が起こらずに、2人目以降のログオンユーザーで発生するなら、セッション 0 の分離の影響を受けている可能性が高いといえます。

### セッション0の分離への対応

* アプリケーションとの通信

アプリケーション間の通信は、リモートプロシージャコール ( RPC ) や名前付きパイプなど、クライアントサーバ型の通信メカニズムを使用するようにします。また、サービスからのメッセージの提示に、ウィンドウメッセージは使わないようにします。たとえば、ユーザーがログオンした直後からサービスと通信をおこなう場合、アプリケーションをスタートアップメニューに登録するか、Run レジストリに登録し、ログオン直後に自動的に起動されるようにします。このアプリケーションを利用して複雑な UI を表示することで、サービスとユーザーインターフェイスを分離することができます。

* WTSSendMessage() によるメッセージボックスの作成

簡単なメッセージボックスをユーザーのデスクトップに表示する際には、WTSSendMessage関数を使用します。この関数を使用すると、サービスはユーザーに通知を送信し、ユーザーからのレスポンスを取得することができます。

* CreateProcessAsUser() による複雑な UI の作成

複雑な UI を作成したいときには、CreateProcessAsUser 関数を使用します。この関数の使い方は、以下の通りです。

1. CreateProcessAsUser 関数を呼び出し、ユーザーのセッションにプロセスを作成します。
2. 作成したプロセスから、UI を表示します。
3. クライアントサーバ型の通信メカニズムで、レスポンスを取得します。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote** |
| 「CreateProcessAsUser」を使用する方法は、「アプリケーションとの通信」で紹介したような、ユーザーセッション内にアプリケーションを起動しておく手間を必要としません。しかし、ユーザーがログオンしているかどうか、また、複数のユーザーがログオンしている場合にはどのユーザーに対して UI を表示するかなど、複雑な判断が必要となるため、推奨されません。 |

* サービスとオブジェクトを共有

サービスで利用可能なイベントやマッピングされたメモリなどのオブジェクトには、明示的にローカルまたはグローバルのいずれかの名前空間を指定します。

## ファイルマッピングオブジェクト

### SeCreateGlobalPrivilege 権限が必要

セッション1以降のセッションから、グローバル名前空間にファイルマッピングオブジェクトを作成するには、SeCreateGlovalPrivilege 権限が必要です。この条件は、Windows Server 2003からセキュリティ強化のために導入されました。SeCreateGlovalPrivilege権限を持つユーザーは、OS によって異なります。以下に一覧を示します。

|  |  |
| --- | --- |
| OS の種類 | SeCreateGlobalPrivilage 権限を持つユーザー |
| Windows Server 2003 | Administrators、Service |
| Windows XP SP2 | Administrators、Service、Interactive |
| Windows Vista  Windows Server 2008  Windows 7 | Administrators、Service |

Windows XP SP2では、互換性問題を軽減するため、一時的にInteractive ユーザーに対して SeCreateGlobalPrivilage 権限を与えていました。これにより、Windows XP にローカルにログオンしているユーザーが使用しているアプリケーションは、グローバル名前空間にファイルマッピングオブジェクトを作成することができました。しかし、Windows VistaやWindows 7 ではInteractive ユーザーに対してこの権限は付与されていません。そのため、このようなタイプのアプリケーションではエラーが発生します。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote** |
| Windows Vista や Windows 7 では、標準ユーザー(フィルタされているAdministrators グループのユーザーを含みます) では、特権が利用できないためエラーになります。 |

### 回避方法および対応方法

グローバルオブジェクトを作成するアプリケーションを、権限の昇格を使用して管理者として実行します。ただしこの方法は、アプリケーションのすべてのコードが管理者権限で実行されてしまうため、以下の方法が推奨されます。

* グローバルオブジェクトを作成する必要があるか検討します。代替の方法があれば変更します。
* サービスとユーザーアプリケーションがオブジェクトを共有する必要があるなら、サービス側からグローバルオブジェクトを作成します。
* グローバルオブジェクトをアプリケーション側で作成する必要がある場合にのみ、権限を昇格し管理者権限で実行します。

## 新フォント環境

### JIS2004対応

Windows Vista と Windows 7は、「JIS X 0213:2004」( 以後JIS2004 ) に対応しています。JIS2004は、経済産業省が2004年に制定した漢字コードに関するJIS規格です。これにより、以前のWindows と比較して、以下のような変更がおこなわれています。

#### 字体の変更

一部文字の形が変わります。たとえば「祇園」を「祇園」、「葛飾区」を「葛飾区」といったように日本語本来の形で表示することができます。Windows VistaやWindows 7 に搭載されるすべての日本語OpenTypeフォントは、JIS2004に対応しています。以下に具体的な対応フォントを挙げます。

* MS ゴシック
* MS P ゴシック
* MS UI Gothic
* MS 明朝
* MS P 明朝
* メイリオ
* Meiryo UI

|  |  |  |
| --- | --- | --- |
| OS | MS ゴシック・MS 明朝のバージョン | JIS |
| Windows 98  Windows Me  Windows NT4.0  Windows 2000  Windows XP  Windows Server 2003 | 2.3 | JIS90 |
| Windows Vista  Windows Server 2008 | 5.0 | JIS2004 |
| Windows 7 | 5.01 | JIS2004 |

##### 字体の変更による問題点と回避方法

* 自体が変更されるだけで、文字コードは変わりません。そのため、以前の Windows で作成したドキュメントを Windows Vista やWindows 7 で開くと、JIS2004の字体に置き換えられます。反対に、Windows Vista やWindows 7 で作成したドキュメントを以前の Windows で開くと、 JIS90の字体に置き換えられます。2つのフォントのメトリック ( 文字の高さや幅 ) は変わらないので、ドキュメントのレイアウトが壊れてしまうことはありません。



**Windows 7**

**(MS フォント5.0)**



**Windows XP**

**(MS フォント2.3)**

**JIS2004環境**

**JIS90環境**



* 1. 字体の変更
* JIS2004対応のフォント (MS ゴシック・MS 明朝 Version 5) は、Windows XPやWindows Server 2003向けにも提供されています。これらを使用すれば、Windows XP やWindows Server 2003でもJIS2004の字体で表示することができます。このフォントは、Microsoft 以下のサイトからダウンロードすることができます。

http://www.microsoft.com/japan/windows/products/windowsvista

/jp\_font/jis04/default.mspx

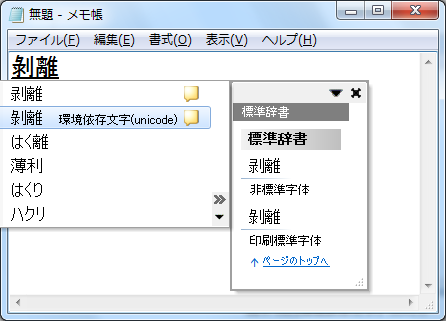
* 日本語OpenTypeフォントのFeature Tagを使用すると、従来の文字セット(JIS90)とJIS2004のどちらを使うか切り替えることができます。たとえば、.NET Framework のWPF ( Windows Presentation Foundation ) は、Feature Tagをサポートしているため、XAML内でタグを指定するだけで、JIS90 とJIS2004を切り替えることができます。

#### 文字の追加

JIS 第3水準、第4水準漢字が追加され、利用できる文字が増えました。たとえば以下のような文字が追加されました。

俱 剝 𠮟 噓 姸 屛 幷 瘦 繫

Microsoft IMEでは、JIS2004で追加された文字に変換しようとすると、図4-28のように、「環境依存文字 ( unicode ) 」というコメントが表示されるため、一目で確認することができます。



* 1. [環境依存文字] コメント

##### 文字の追加による問題点と回避方法

Windows Vistaおよび Windows 7で作ったドキュメントを以前のWindowsで開くと、JIS2004で追加された文字が中黒「・」や四角「■」で表示されます。



**Vista**

**(MS フォント5.0)**



・

・

・

・

**XP**

**(MS フォント2.3)**

**JIS2004環境**

**JIS90環境**



* 1. 文字の追加

この場合、Unicode で保存する必要あります。Shift\_JISやEUC-JPなどで保存すると、文字がなくなってしまうことがあります。たとえば、メモ帳で開くとこれらの文字は「？」で置き換えられてしまいます。

### メイリオフォント

Windows Vista とWindows 7 の標準日本語フォントとして、「メイリオ」や「Meiryo UI」が搭載されています。「メイリオ」の名は、日本語の「明瞭」に由来しており、その名の通り画面上で見ても印刷しても極めて明瞭で読みやすのが特徴です。特に、ClearType に対応しているため、スクリーン上での日本語の可読性が向上しています。そのため、小さな文字もつぶれずに表示することができます。

また、JIS2004 に対応しているため、漢字などは日本語本来の字体で表示することができます。

レギュラーとボールドの 2 書体が提供されていますが、イタリック体は欧文文字のみの提供となっています。以下に、各フォントのサンプルを示します。

|  |  |  |  |
| --- | --- | --- | --- |
| フォント | レギュラー | ボールド | イタリック |
| MS ゴシック | コンピューター  ネットワーク  プリンタとFAX | **コンピューター**  **ネットワーク**  **プリンタとFAX** | *コンピューター*  *ネットワーク*  *プリンタとFAX* |
| MS Pゴシック | コンピューター  ネットワーク  プリンタとFAX | **コンピューター**  **ネットワーク**  **プリンタとFAX** | *コンピューター*  *ネットワーク*  *プリンタとFAX* |
| MS UI Gothic | コンピューター  ネットワーク  プリンタとFAX | **コンピューター**  **ネットワーク**  **プリンタとFAX** | *コンピューター*  *ネットワーク*  *プリンタとFAX* |
| メイリオ | コンピューター  ネットワーク  プリンタとFAX | **コンピューター**  **ネットワーク**  **プリンタとFAX** | *コンピューター*  *ネットワーク*  *プリンタとFAX* |
| Meiryo UI | コンピューター  ネットワーク  プリンタとFAX | **コンピューター**  **ネットワーク**  **プリンタとFAX** | *コンピューター*  *ネットワーク*  *プリンタとFAX* |

JIS2004やフォント、移行のシナリオについての詳細は、MicrosoftのWebサイト ( http://www.microsoft.com/japan/windows/products/windowsvista/jp\_font/default.mspx ) を参照してください。

## プログラムの互換性アシスタント ( PCA )

### PCA とは

既存のアプリケーションを動作させるため、Windows Vista やWindows 7 では、「互換性」タブを使っての実行環境を調整することができます。これは大変便利なツールですが、ユーザーが設定をしないと使えないという欠点があります。そこで、PCA 機能が提供されています。これは、互換性に問題がある既存のアプリケーションのパターンを検出し、回避策を提供するという機能です。検出する互換性問題には、以下のようなものがあります。

* インストールの失敗
* プロセス作成の失敗

さらに、実行時に問題が起こることがすでに分かっているプログラムを起動しようとした時には、警告などのメッセージを表示します。

### インストール失敗の検出

インストールが失敗した可能性があると、「プログラム互換性アシスタント」ダイアログボックスが表示されます。ここでユーザーは、3つのオプションのうちのいずれかを選択することができます。

* 「推奨の設定を使用して再インストールする」

セットアップが正常に完了しなかった場合に選びます。このオプションを選択すると、「Windows XP SP2互換モード」を使用して、セットアップを再度実行します。

* 「このプログラムは正しくインストールされました」

セットアップが正常に完了した場合に選びます。たとえ、セットアップが正しく完了しても、PCA によってダイアログが表示されることがあります。ここで、ユーザーがこのオプションを選択しておくと、次回から PCA ダイアログが表示されないように設定することができます。

* キャンセル

PCA は何も処理を行いません。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote: PCA による検出条件** |
| PCA は、UAC によりインストーラとして検出されたアプリケーションに対して、インストールが成功したかどうかがを確認します。UAC により、インストーラとして認識された場合には、PCA により実行結果が確認されます。具体的には PCA は [プログラムの追加と削除] のエントリを探します。エントリが追加されていないと、インストールに失敗した可能性があるとして、「プログラム互換性アシスタント」ダイアログボックスを表示します。  だたし、以下のようなインストーラには、PCA による検出はおこなわれません。   * ネットワークインストールである * すでに互換フィックスが適用されている * マニフェストで実行権限が設定されている |

### プロセス作成の失敗の検出

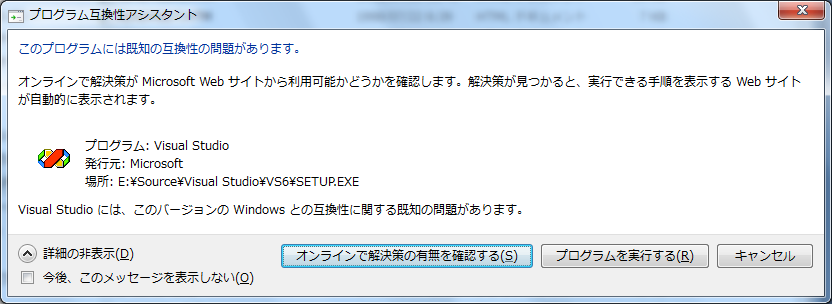
管理者として実行されていないプログラムが、管理者権限が必要な子プログラムを昇格なしに呼び出すとエラーが発生します。これは既存のアプリケーションにおいては、発生する可能性が高いエラーと考えられます。そこで、子プログラムの起動中にエラーが発生した場合には、PCA により失敗が検出され、ダイアログボックスが表示されます。そして、呼び出し側のプログラムに対して、「ElevateCreateProcess」互換フィックスが自動的に適用されます。この互換フィックスは、自身から呼び出すプログラムに対して、昇格をおこない管理者権限で実行できるようにします。互換フィックスの適用により、次回同じプログラムを起動したときには、UAC による昇格確認用のダイアログボックスが表示され、管理者権限で実行することが可能になります。2回目以降の起動時には、「ElevateCreateProcess」互換フィックスが自動的に適用されるため、PCA のダイアログボックスは表示されません。

### 既知の問題に対するメッセージプロセス作成

実行時に問題が起こることがすでに分かっているプログラムを起動しようとした時には、以下のいずれかの処理が行われます。

* 警告

システム動作への影響はありませんが、使用上問題があるアプリケーションを実行した場合に表示されるメッセージです ( 図4-30参照 )。ここで、「オンラインで解決策の有無を確認する」ボタンをクリックすると、Windows エラー報告が Microsoft に送信され、オンラインで回答が表示されます。また、「プログラムを実行する」ボタンをクリックすると、操作を継続することができます。



* 1. 「プログラム互換性アシスタント」ダイアログ
* ブロック

ブルースクリーンになるなど、システムの動作に影響があるアプリケーションを実行した場合に表示されるメッセージです。ブロックメッセージのダイアログボックスでは、ボタンは「オンラインで解決策の有無を確認する」しか表示されません。そのため、アプリケーションの操作を継続することはできません。

## Internet Explorer 8 の標準準拠

**Windows 7 Only**

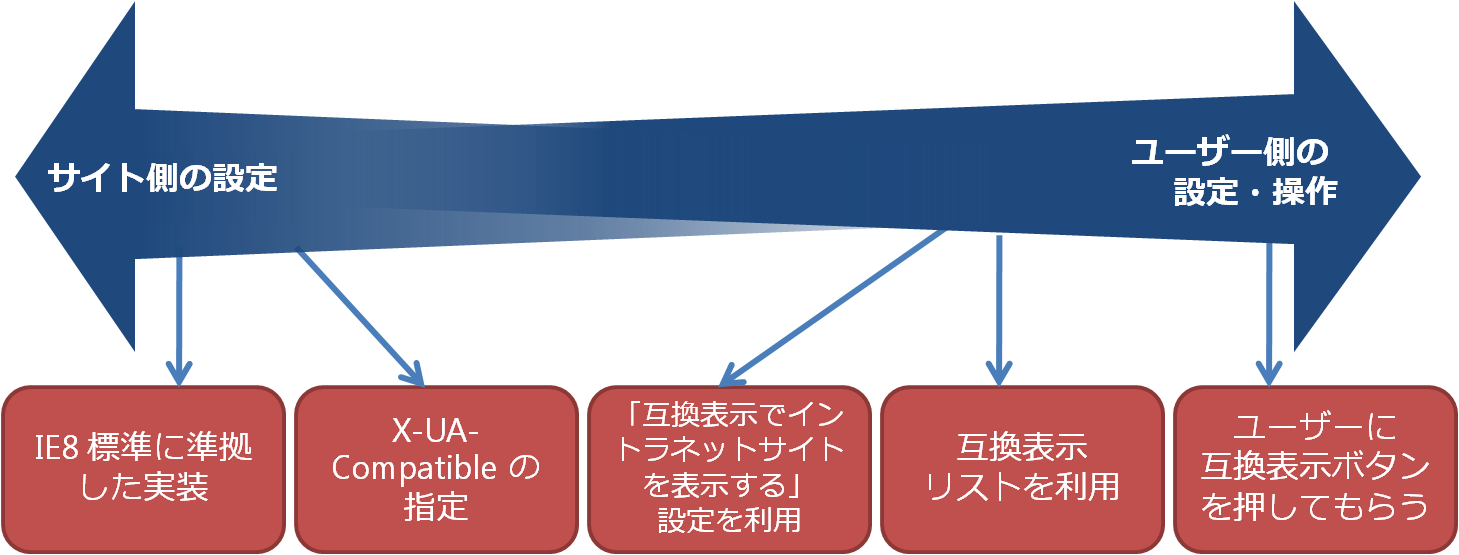
IE8は HTML や CSS の標準に準拠することを重視して開発されました。これまでも、基本的には W3C が定める標準規格を基に作成されていましたが、解釈にずれがあったり、独自の仕様が追加されていたりしたため、ブラウザー間で表示が異なるケースがありました。そのため、Web 開発者はブラウザーごとにCSSを記述するといった必要がありました。IE8 は標準規格に厳格に準拠しているため、相互運用性や保守性を高めることができます。

その一方で、これまでの IE7 で表示できた Web ページが、IE8 ではレイアウトやデザインが崩れてしまう可能性があります。そこで IE8 で IE7 との互換性を保つために、以下の回避方法と解決策が考えられます。

1. IE8 標準に準拠した実装にページを書き換える
2. X-UA-Compatible を指定する
3. IE8 のメニューから「互換表示でイントラネットを表示する」設定を利用する
4. IE8 の互換表示リストを利用する
5. ユーザーにIE8 の互換表示ボタンを押してもらう

これらの方法のうち、①と②はサイト側、つまり開発者や管理者がおこなう設定です。一方、③から⑤は、ユーザーがおこなわなければならない設定や操作です。つまり、③から⑤は、ユーザー任せになってしまうため、ユーザーがその機能を知らなければ、ページは正しく表示されないままになってしまいます。そこで、サイト管理者・開発者は①と②をおこなう必要があります。

①は 開発者がWeb ページをIE8 対応に完全に書き換えるものです。最終的にはこれが解決策となりますが、容易ではありません。一方、②はIE7 との互換性を保つための設定です。設定ベースなので容易にできるため、IE7 対応のページを早急に IE8 上でも表示できるようにしたい、といった場合には適切な回避策といえます。このドキュメントは、②について詳細に取り上げます。



* 1. サイト側の設定とユーザー側の設定・操作

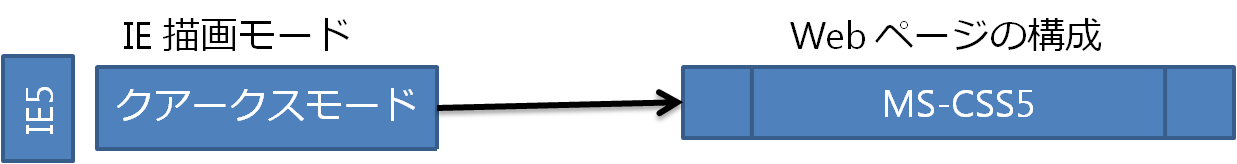
|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote: X-UA-Compatible以外の互換性機能の概要** |
| * IE8 標準に準拠した実装にページを書き換える   IE8では以下の標準テクノロジに準拠しています。IE8では、「IE に対応した Web ページ」というよりも、「標準に準拠したWebページ」を意識して開発することになります。   * CSS 2.1 * DOM の標準 * HTML の標準 * Acid 2 テスト対応 * IE8 のメニューから「互換表示でイントラネットを表示する」設定を利用する   企業内の Web アプリケーションの対応を容易にする機能です。  IE8には「互換表示設定」というメニューがあります。このメニューによって表示されるダイアログで、「互換表示でイントラネットサイトを表示する」を選択します (既定で有効) 。 これにより、イントラネットのサイトは IE7 互換モードで描画されるようになるため、IE7 で表示できた 企業内のサイトが正しく表示されるようになります。     * 1. 互換表示でイントラネットサイトを表示する * IE8 の互換表示リストを利用する   IE7 互換モードで表示したいサイトをあらかじめ登録しておくことができます。ここに登録しておくことで、常にそのサイトに対しては IE7 互換モードでアクセスすることができます。  また、「マイクロソフトからの更新された Web サイト一覧を含める」を有効にしておくと、よく利用されている有名なサイトに対しては、Microsoftが管理している IE7 で表示できるサイトのリストを自動的に適用することができます。     * 1. 互換表示リストに追加 * ユーザーにIE8 の互換表示ボタンを押してもらう   表示したページが崩れていた場合、ユーザーはアドレスバーの右側に表示される「互換表示」ボタンをクリックすることで、手動で IE7 互換モードに切り替えることができます。     * 1. 互換表示ボタン |

### これまでの IE の各バージョンの描画モード

IE5からIE7 までの IE には、以下の描画モードがありました。

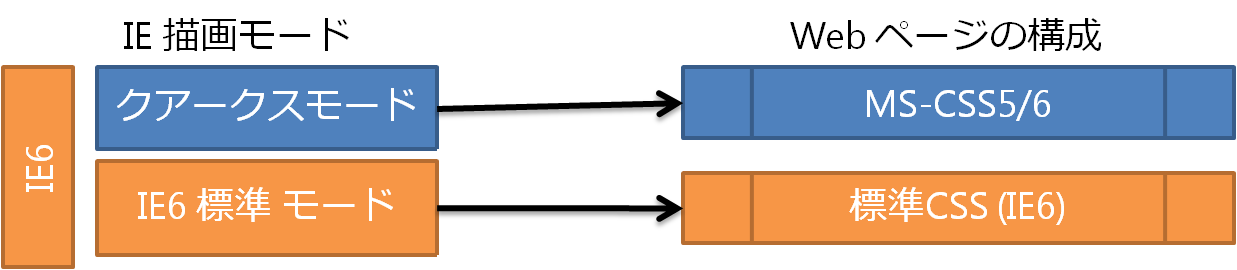
* IE5

IE5 では、IE 独自のクアークスという描画モードが使われていました。



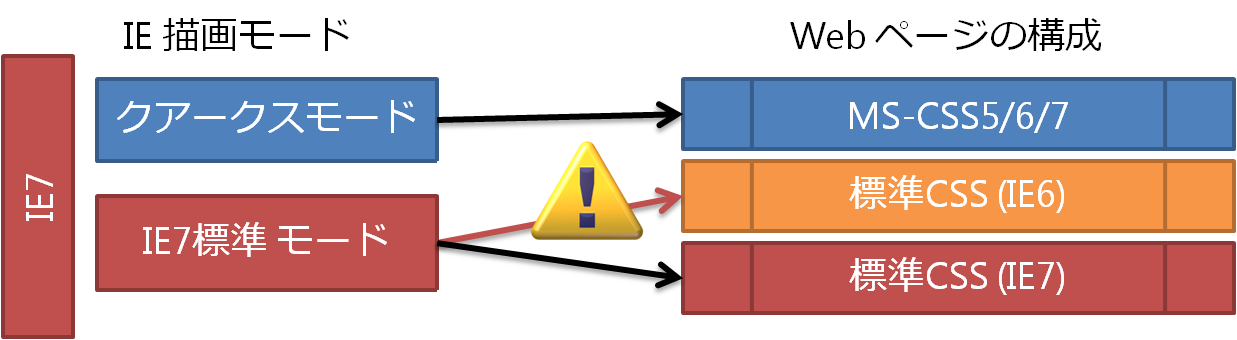
* 1. IE5 の描画モード
* IE6

IE6 以降、標準化が重要視されたため、「IE6 標準モード」が追加されました。また、従来のページも表示できなくてはいけないため、「クアークスモード」もサポートされていました。そして、表示するページによってモードを切り替えていました。



* 1. IE6 の描画モード
* IE7

IE7 では、クアークスモードはそのまま搭載され、標準モードがIE7 の CSS用にバージョンアップされました。



* 1. IE7 の描画モード

このように、IE6 と IE7 では、クアークスまたは標準の2つのモードがサポートされていました。そのため、クアークスか標準かが分かれば、適切に描画モードを分けることができました。

### これまでの IE のDTD 判別ロジック

IE5 から IE7 では、クアークスモードで描画するか、標準モードで描画するかは、DTD (Document Type Definition) を使用して設定します。例えば、以下はクアークスモードで描画される DOCTYPE 設定の一例です。

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN">

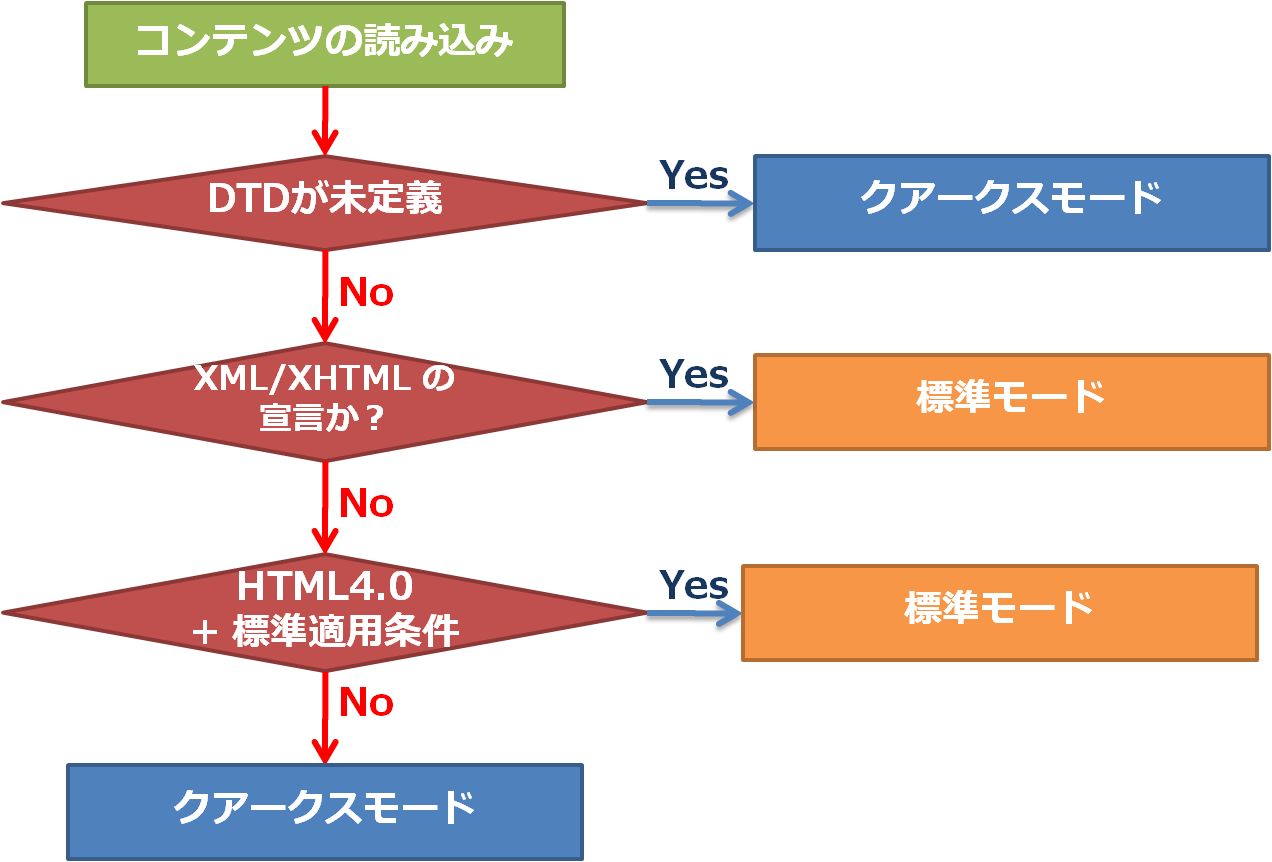
また、以下の DOCTYPE は標準モードで描画される一例です。

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Strict//EN">

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| DOCTYPE の設定の詳細は、付録「描画モード切り替え表」を参考にしてください。 |

このDTD の設定には、図4-38のような明確な判別ロジックがあり、これによって適切な描画モードが選択されることになります。

1. コンテンツを読み込んだときに、DTD が未定義だった場合は、IE6よりも前のコンテンツであると判断され、クアークスモードで描画されます。
2. DTD が定義されていて、XML や XHTML の宣言がある場合は、標準モードが適用されます。
3. DTD が定義されていて、XHTML 4.0 および標準適用条件を満たしている場合も標準モードが適用されます。
4. 上記のいずれの条件にも当てはまらない場合は、クアークスモードが適用されます。



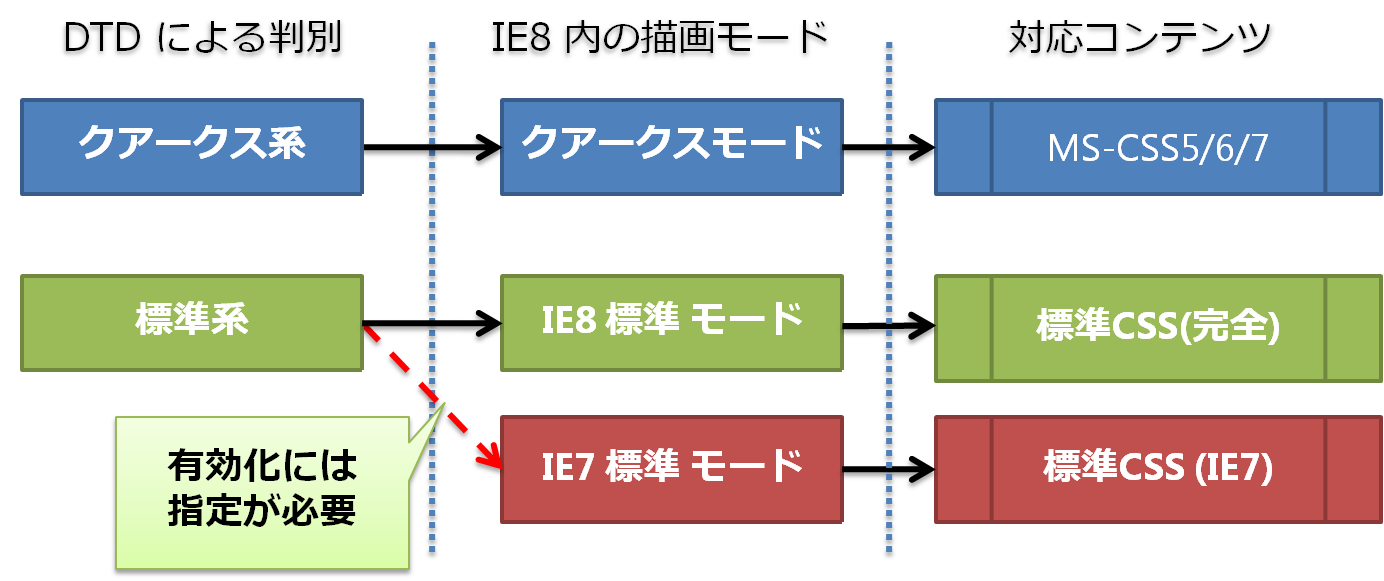
* 1. IE の DTD 判別ロジック

### IE8 の描画モード

IE5 から IE7 とは異なり、IE8 では3つの描画モードが搭載されています。クアークスモードと IE7 標準モード、そして IE8 標準モードです。2つの標準モードをサポートしているのは、IE7 と IE8 の描画には差異があるからです。IE7 用に作成されたページが、IE8 で正しく表示されない可能性があるため IE7 の互換モードが用意されています。

|  |  |  |  |
| --- | --- | --- | --- |
| DTD で判断される部分 | クアークス系 | 標準系 | |
| 適用される描画モード | クアークスモード | IE7 標準 モード | IE8 標準 モード |
| 対応CSS | MS-CSS | 標準CSS(IE7) | 完全標準CSS |

どの描画モードが適用されるかは、IE7 と同様、まずは DTD によって決まります。つまり、ここでクアークスか標準かが決まります(図4-39)。標準系だった場合、既定では IE8 の標準モードで描画されます。そして、IE7 の互換指定があった場合にのみ、IE7 標準モードが使用されます。そのため、IE8 標準モードで表示したときに問題があるページでは、IE7 の互換指定をしなければなりません。



* 1. IE8 の描画モード

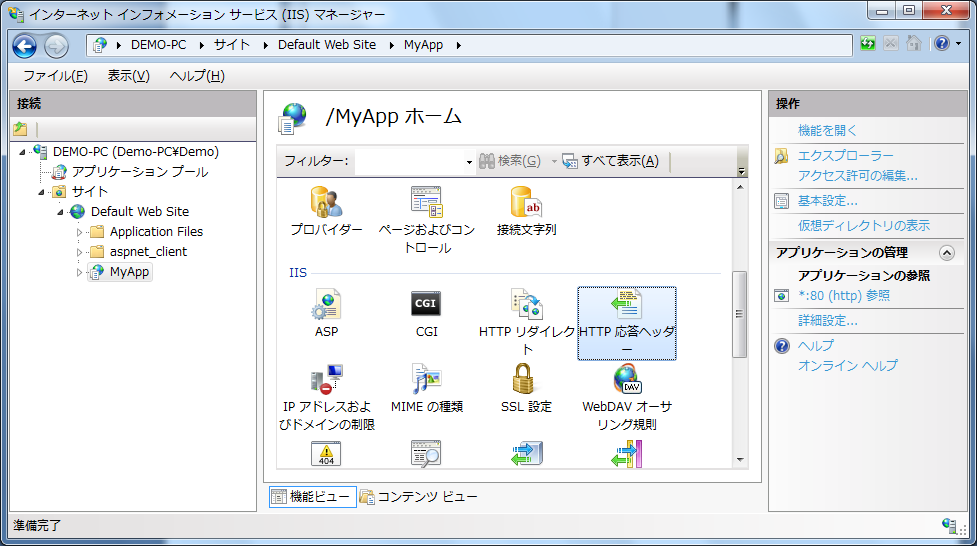
### X-UA-CompatibleによるIE7 互換モードへの切り替え

IE8で IE7 の互換モードで描画する必要があるときには、X-UA-Compatible 互換モードスイッチを指定します。この設定は、以下のいずれかの方法で設定します。

* HTTP ヘッダーによる指定

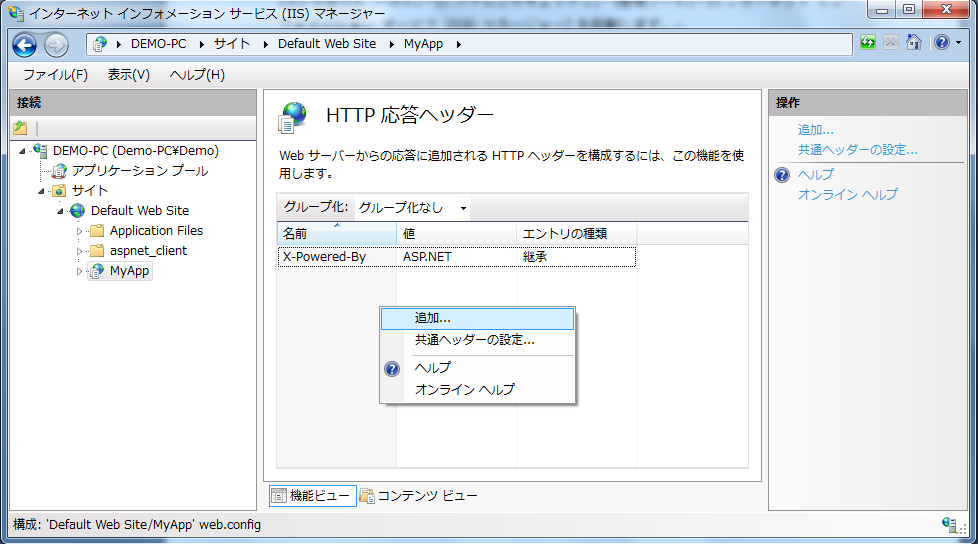
HTTP ヘッダーに「X-UA-Compatible: IE=EmulateIE7」という値を含めます。この方法を使用すると、Web サイト単位やフォルダー単位など、まとめて互換モードスイッチの指定をすることができます。IIS の管理ツールを使用して以下の手順で設定します。

1. [コントロール パネル]－[システムとセキュリティ]－[管理ツール]－[インターネット インフォメーション サービス (IIS) マネージャー] を起動します。
2. 設定したい Web サイトや フォルダーを選択し、[HTTP 応答ヘッダー] をダブルクリックします。



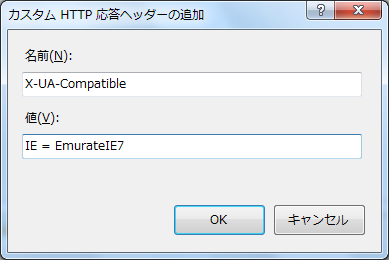
* 1. IIS管理ツール

1. 真ん中のペインを右クリックし、[追加] を選択します。



* 1. HTTP 応答ヘッダー

1. 「カスタムHTTP 応答ヘッダーの追加」で、名前として「X-UA-Compatible」、値として「IE = EmurateIE7」を設定し、「OK」ボタンをクリックします。



* 1. カスタムHTTP 応答ヘッダーの追加ダイアログ

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| IIS 管理ツールでの設定により、Web.config に以下のタグが追加されます。直接 Web.configにこの設定を追加してもかまいません。  <?xml version="1.0" encoding="UTF-8"?>  <configuration>  <system.webServer>  <httpProtocol>  <customHeaders>  <add name="X-UA-Compatible" value="IE = EmurateIE7" />  </customHeaders>  </httpProtocol>  </system.webServer>  </configuration> |

* METAタグによる指定

ファイル単位で設定をしたいなら、コンテンツ(HTMLファイル) に直接 以下のような META タグを追加する方法もあります。

<meta http-equiv=“X-UA-Compatible” content=“IE＝EmulateIE7”>

なお、HTTP ヘッダーによる指定と、META タグによる指定が同じコンテンツに対しておこなわれた場合は、META タグの設定値が優先されます。

### X-UA-Compatibleの設定値

X-UA-Compatible では、以下の表のような値を設定することができます。

|  |  |
| --- | --- |
| 値 | 適用されるモード |
| IE=5 | クアークスモード |
| IE=7 | IE7標準モード |
| IE=EmulateIE7 | IE7標準/クアークスモード(DTDによって切り替え) |
| IE=8 | IE8標準モード |
| IE=EmulateIE8 | IE8標準/クアークスモード(DTDによって切り替え) |
| IE=edge | 最新モード |

「IE = EmulateIE7」は、「IE7 と同様の動作をする」という意味です。つまり、DTD の設定によって標準モードかクアークスモードかが切り替わります。そのため、今まで IE7 で表示できていたページは IE8 でも表示できることになります。「IE = EmulateIE8」も同じです。DTD に依存することになります。

「IE = edge」は、最新モードで表示するための設定です。つまり、IE8 であれば IE8 の標準モードで、そして今後IEの新しいバージョンがリリースされた場合には、その最新バージョンでで表示されることになります。

現在のモードを知るためには、document.documentModeプロパティを使用します。IE のアドレスバーに以下のJavaScriptを入力すると、現在のプロパティ値を表示することができます。

javascript:alert(document.documentMode)



* 1. documentModeプロパティの確認



* 1. documentMode プロパティの値表示

各モードの戻り値は以下の通りです。

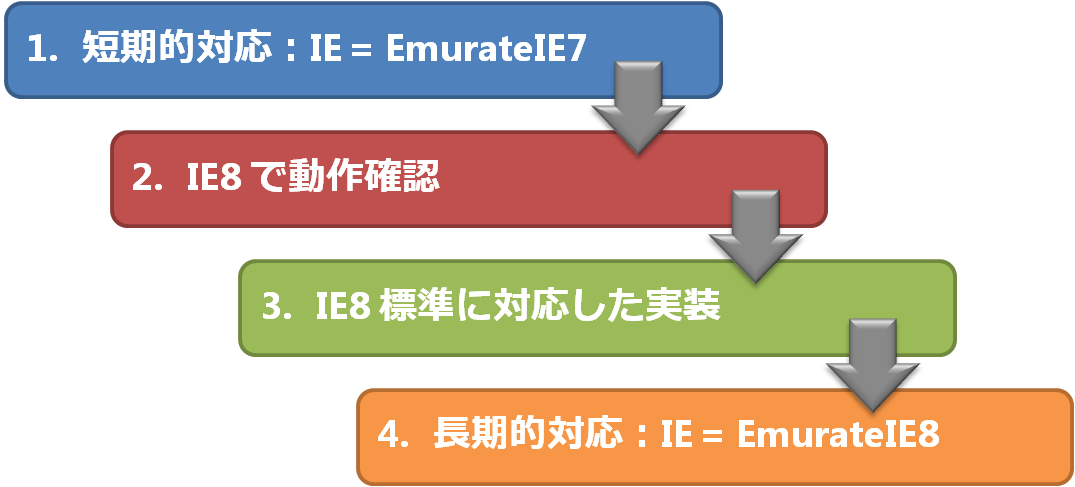
|  |  |
| --- | --- |
| モード | 値 |
| クアークスモード | 5 |
| IE7標準モード | 7 |
| IE8標準モード | 8 |

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote: ユーザーエージェント文字列** |
| ユーザー エージェント文字列は、Web サーバーにレポートされるブラウザーの IDです。Web アプリケーション開発者は、ユーザーエージェント文字列を使用して、クライアントが使用しているブラウザーの種類を検出することができるため、ブラウザーの種類ごとにコンテンツを変えたい場合に役立ちます。ユーザーエージェント文字列を表示するためには、navigator.userAgentプロパティを使用します。IE のアドレスバーに以下のJavaScript を入力することで確認できます。  javascript:alert(navigator.userAgent)     * 1. userAgentプロパティの確認   たとえば、IE8 では、図4-46のようなユーザー エージェント文字列を渡します。     * 1. IE8 のユーザーエージェント文字列   図4-46の「MSIE 8.0」はブラウザーのバージョンです。この値は IE7 を使用している場合は「MSIE7.0」です。また、IE8 のIE7 互換モードの場合にも同じく「MSIE7.0」が返されます。これだと、本当の IE7 なのか、互換モードのIE7 なのかを区別することができません。そこで、役に立つのが「Trident/4.0」です。IE8 の場合には、このユーザーエージェント文字列が追加されています。  ユーザーエージェント文字列の詳細は、以下のサイトをご覧ください。  http://msdn.microsoft.com/ja-jp/library/ms537503.aspx |

### 段階的なサイト互換性の実現

Web サイトで IE7 から IE8 に移行する際には、図4-47のようなプロセスで実行します。

1. IE7 で表示できたページが IE8で表示できなくなってしまった場合、短期的な対応としては IE = EmurateIE7 を設定し互換性を保ちます。
2. IE8 のIE7 互換モードで正しく表示されるかどうかをテストします。
3. 将来的には、IE8 標準に対応するよう ページを書き換えます。
4. IE = EmurateIE8 を設定し、IE8 標準モードで表示されるようにします。これが 最終的な互換性問題の解決策です。



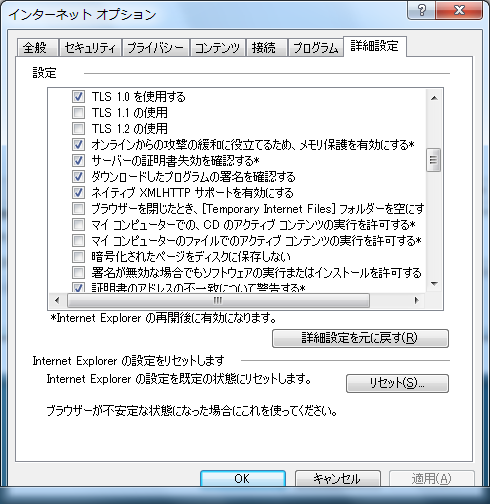
* 1. 段階的なサイトの互換性実現

## DEP/NXの既定値

**Windows 7 Only**

### DEP/NX とは

DEP/NXは、データ実行防止 (DEP) または実行不可 (NX) とも呼ばれ、実行不可とマークされたコードがメモリ上で実行されないようにします。そのため、バッファ オーバーランのようなメモリ関連の脆弱性を利用する攻撃を阻止することができます。この機能は、Internet Explorer の [ツール]－[インターネットオプション] を使用して、有効/無効を切り替えることができます。



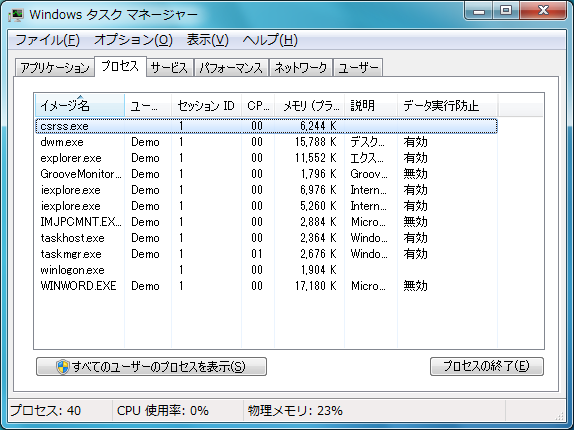
* 1. メモリ保護の有効

このオプション自体は Internet Explorer 7 にも存在しましたが、互換性を配慮し、既定では無効になっていました。それがInternet Explorer 8 では、既定で有効に変更されました。

### 発生する互換性問題

DEP/NX は、Internet Explorer だけでなく、読み込まれるアドオンにも適用されます。そのため、この機能を有効にすると、DEP/NX と互換性のないアドオンはIE8 で読み込むだけでクラッシュします。最も多い問題としては、古いバージョンの ATL ライブラリ (Version 7.1 SP1 より前のバージョン) で構築されたアドオンです。

なお、タスク マネージャーの [プロセス] タブでは、DEP/NX で保護されるプロセスを確認することができます。



* 1. タスクマネージャーによるデータ実行防止確認

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| 「データ実行防止」列は、既定では表示されません。表示するためには、タスクマネージャーの [表示] メニューから、[列の選択] で [データ実行防止] チェックボックスをオンにします。 |

### 問題の回避

クラッシュしたアドオンの更新バージョンを展開できるまで、DEP/NX を無効にします。この設定は、管理者として IE8 を実行する必要があります。

### 問題の解決

* エンドユーザー向け

DEP/NX と互換性のある最新バージョンのアドオンをインストールします。

* 開発者向け

DEP/NX と互換性のあるアプリケーションとしてコンパイルします。そのためには、以下の機能を活用することができます。

* /NXCOMPAT リンカ オプションを使用して、DEP/NX 互換性を指定します。
* スタック保護 (/GS)、安全な例外処理 (/SafeSEH)、および ASLR (/DynamicBase) など、利用できる他の保護策をコードで有効にします。

## Windows メールの削除

**Windows 7 Only**

### Windows メールの削除

Windows Vistaに付属していた「Windows メール」は、Windows XPでの「Outlook Express」の後継で、フィッシング詐欺、スパムメール対策が施されたメーラーです。Windows 7 では、このWindows メールが削除されます。これに伴い、CoStartOutlookExpress API は無効化されます。その他のメール APIも、非推奨としてマークされ、今後のバージョンの Windows では廃止される予定です。ただし、公開された API については、廃止または非推奨としてマークされていない限り、Windows 7 で引き続き動作します。さらに、ユーザーの電子メール ファイル (.eml) およびニュース ファイル (.nws) もシステム上に維持されます。

Windows メールのほかに、以下の機能もWindows 7 から削除されます。これらは Windows Live で同様のアプリケーションが提供されます。

* Windows メール
* Windows フォトギャラリー
* Windows ムービーメーカー
* Windows カレンダー
* Windows アドレス帳

### 発生する互換性問題

Windows メールの削除による主な互換性問題は、次のようなものが考えられます。

Windows メールのUI の起動を試みるAPI は、警告なしで失敗します。関数からは成功の値が返されますが、ユーザーに対して UI は表示されません。他のダイアログ ボックスを呼び出す API (スプーラーやアカウントのダイアログなど) については、引き続き UI が表示されます。

プロトコル (mailto、ldap、news、snews、nntp) ハンドラーは、今後、Windows メールにも連絡先にも関連付けられません。起動しようとすると、エラー ダイアログが表示され、別のプログラムとの関連付けを設定するように促されます。

ファイルの関連付け (.eml、.nws、.contact、.group、.wab、.p7c、.vfc) は解除されるか、無効化されます。これらの拡張子を持つファイルを開こうとすると、既にインストールされていて使用可能な状態にある他のアプリケーションを一覧にしたダイアログ ボックスが表示され、解決策を提供する Web ページに誘導されます。

また、デフォルトのメールクライアントを見つけるために、以下のレジストリを参照している場合、Windows Live メールをインストールしても起動されません。

HKLM\Software\Clients\Mail

これは、Windows メールではここに自身を登録していましたが、Windows Live メールはここに書き込まないからです。

### 問題の回避と対応

* エンドユーザー向け

.eml および .nws ファイルの読み取りには、Windows Live メールなど、対応した他のメール製品をインストールします。

* 開発者向け
* アプリケーションを Windows 7 環境で実行し、Windows メールを呼び出していないかどうかを確認します。Windows メールに関する API を呼び出すコードは正常に動作しないので、そのような設計はおこなわないようにします。特に、「deprecated」や「obsolete」とマークされているものは、今後削除される可能性があるので使用しないようにします。
* デフォルトのメールクライアントを見つけるために、「HKLM\Software\Clients\Mail」は使用しないようにします。代わりに「HKCU\Software\Clients\Mail」を参照するようにします。

## 容易な High DPI 設定

**Windows 7 Only**

### High DPI とは

Hight DPI を使用すると、テキストやイメージのサイズを画面の解像度ではなく DPI で変更できるようになります。これにより、解像度を低くすることなく、テキストを大きく表示することができるようになります。また、適切な解像度でないために、テキストがぼやけてしまうといったことがなくなります。

High DPI 自体は、Windows XP 以来サポートされていましたが、Windows 7 では、次の 2 つの機能が新たに追加されました。

* ユーザー単位で DPI 設定が可能になりました。 (以前はコンピューター単位)
* DPI を変更しても再起動が必要なくなりました。 (ただし、ログオフ/ログオンは必要)

このように、より容易に High DPI の設定ができるようになったため、使用するユーザーが増える可能性があります。それに伴い、これまで発覚しなかった問題が顕在化する可能性があります。

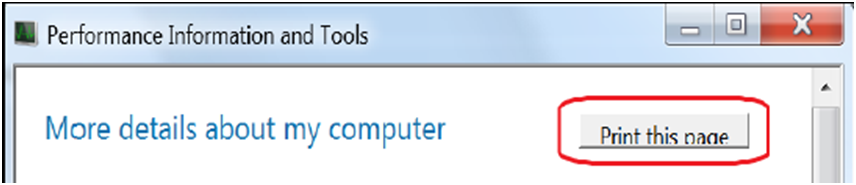


* 1. DPI 設定画面(コントロールパネル)

### 発生する互換性問題

High DPI を想定していないアプリケーションでは、次に示すような視覚的な問題が生じることがあります。

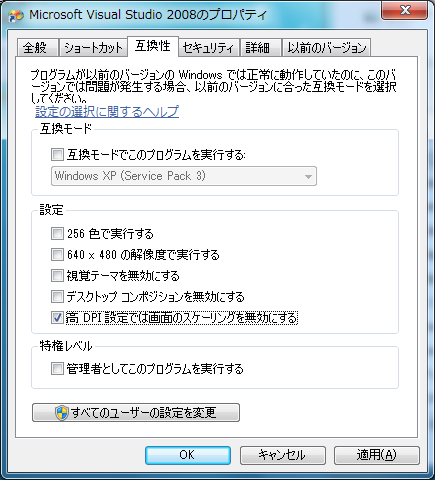
* ボタンがスクリーンからはみ出してしまう
* ドラッグアンドドロップができなくなる
* 字が重なってしまう
* 適切でないフォント・サイズで表示されてしまう



* 1. High DPI により発生する問題

### 問題の回避と対応

DPI の設定が 150% 以上に設定した場合に問題が発生するなら、互換性タブを利用することができます。互換性タブには、図4-52のように、「高DPI 設定では画面のスケーリングを無効にする」というチェックボックスがあります。これを有効にすることで、問題を回避できるアプリケーションもあります。



* 1. 高DPI 設定では画面のスケーリングを無効にする オプション

これは、DPI 150% 以上で使用されるスケーリング技術がWindows Vista から変更されたからです。このチェックを有効にすると、新しいスケーリングを無効にすることができるため、以前と同じように表示できるアプリケーションもあります。

また、アプリケーションを DPI 対応にするには、次の手順に従います。

1. DPI 対応評価を実行し、検出されたすべての問題を記録します。
2. 個々の問題を DPI のコーディングに関する一般的な問題と照らしてチェックします。
3. アプリケーションを完全に DPI 対応化するためのコストを評価します。
4. 必要な High DPI アセット (ボタン、アイコンなど) の一覧を作成します。
5. ①で見つかった DPI の問題を解消します。
6. ④で把握した新しいアセットを統合します。
7. アプリケーションを DPI 対応として宣言します。
8. DPI 対応評価を再度実行し、問題が解決されていることを確認します。

なお、High DPI 対応アプリケーションの開発に関しては、以下の Web サイトを参考にしてください。

http://msdn.microsoft.com/en-us/library/dd464660.aspx

## 長音表記の変更

**Windows 7 Only**

### 長音表記の変更

Microsoft では、外来語に由来するカタカナ用語の末尾の長音表記を、国語審議会の報告をもとに告示された 1991 年の内閣告示第二号に基づく表記に準拠することになりました。これにより、Windows 7 では、ダイアログやメニューなどで、「コンピュータ」は「コンピューター」に、「フォルダ」は「フォルダー」といったように変更されています。

|  |  |
| --- | --- |
| ～ Windows Vista | Windows 7 |
| コンピュータ | コンピューター |
| エクスプローラ | エクスプローラー |
| ドライバ | ドライバー |
| フォルダ | フォルダー |

### 発生する互換性問題と問題の解決

マニュアルやドキュメントなどで表記の厳密性が求められる場合、変更の必要性を確認し対応する必要があります。

# Application Compatibility Toolkit ( ACT )

## ACTとは

ACTは、アプリケーションの互換性に関する検証やテストをおこなうための移行支援ツールで、以下の機能があります。

* Inventory Collector

現在の環境に存在する全てのアプリケーションを調査し、正確な一覧 ( インベントリ ) を作成することができます。

* Evaluator

互換性の問題点を評価するツールで、問題点が業務に対してどのぐらいインパクトがあるのかを調べることができます。以下の評価機能を用意していますが、これらのツールはユーザーコンピュータ上で実行させ続けます。そして、ユーザーには通常どおりアプリケーションを使用してもらい、アプリケーションで問題が起こったときには Evaluatorがレポートします。

* Internet Explorer Compatibility Evaluator

IE 8の互換性や問題点をチェックすることができるツールです。このツールはIE 8をインストールしたコンピューターでテストを実行する必要があります。

* User Account Control Compatibility Evaluator

UAC の互換性や問題点をチェックすることができるツールです。このツールは、Windows XPやWindows Server 2003上で実行することができるため、Windows 7を展開する前にテストすることができます。

* Application Compatibility Manager

Evaluator によりレポートされた問題点をACT が自己評価します。

Compatibility Administrator

問題解決ツールです。互換性問題への対応を作成し、各コンピューターに展開することができます。

ACT 5.5は、Microsoft の以下のWebサイト からダウンロードすることができます。

http://www.microsoft.com/downloads/details.aspx?

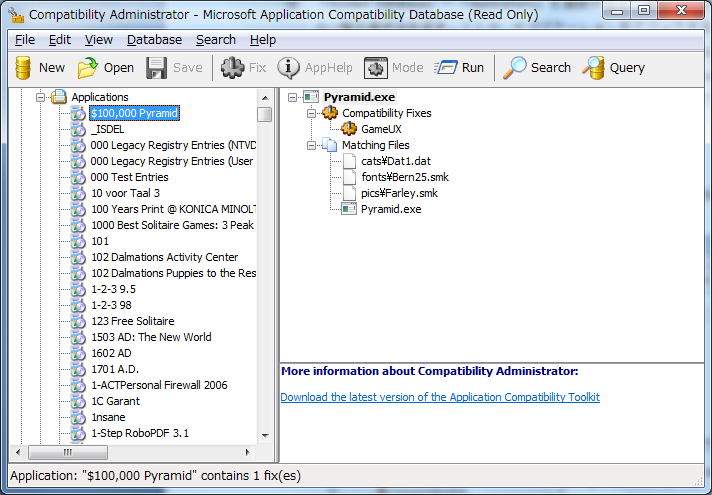
FamilyID=24da89e9-b581-47b0-b45e-492dd6da2971&displaylang=en

## Compatibility Administratorの利用

Compatibility Administratorを使用すると、アプリケーションの互換フィックスや互換モード、アプリケーションヘルプなどを作成することができます。また、これらをデータベースに保存し、各コンピューターに展開することもできます。以下に操作方法を示します。

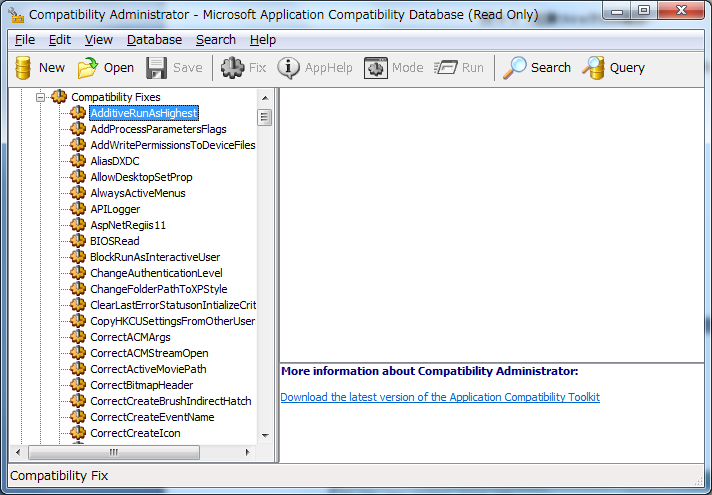
### **Compatibility Administrator** の起動

1. スタートメニューから、[ Microsoft Application Compatibility Toolkit 5.5 ]－[ Compatibility Administrator ] を右クリックし、[ 管理者として実行 ] を選択します。Compatibility Administratorは、管理者権限で実行します。
2. 「System Database」－「Applications」を展開すると、登録されているアプリケーションの一覧が表示されます。ここで、アプリケーションをクリックすると、識別情報や実行に必要な互換性フィックスが表示されます。これらのアプリケーションに関する情報は、すでにコンピューターに登録されています。（%Windir%\AppPatch\sysmain.sdbファイル）そのため、登録されているアプリケーションを実行する際には、互換フィックスが自動的に適用されます。



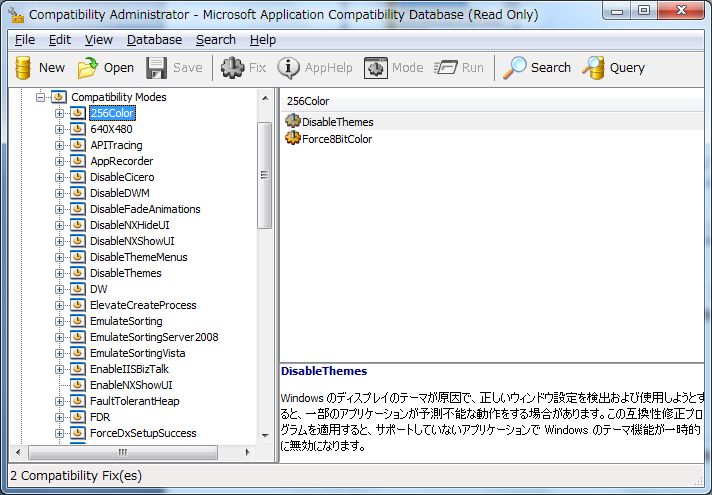
* 1. アプリケーション情報の確認

1. 「System Database」－「Compatibility Fixes」を展開すると、登録されている互換フィックスの一覧が表示されます。



* 1. 互換フィックスの確認

1. 「System Database」－「Compatibility Modes」をクリックすると、登録されている互換モードの一覧が表示されます。各互換モードをクリックすると、互換モードに含まれる互換フィックスを確認することができます。

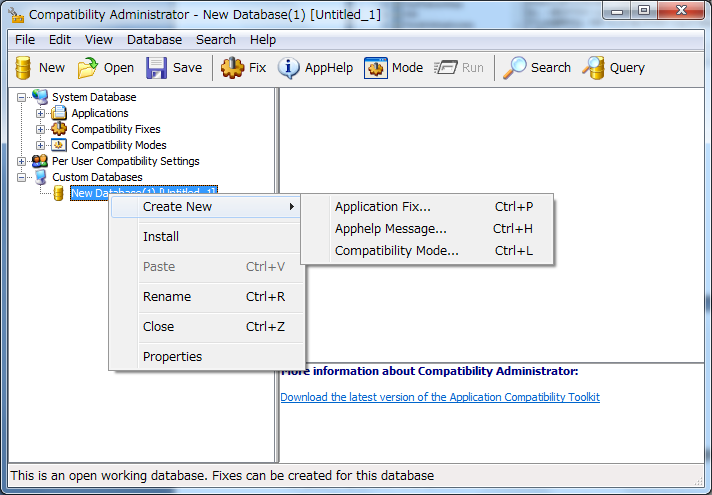


* 1. 互換モードの確認

### アプリケーションへ適用する互換フィックスの定義

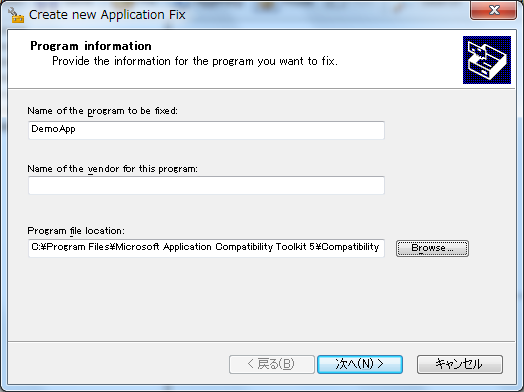
Compatibility Administrator 使用すると、特定のアプリケーションに適用したい互換フィックスを追加することができます。

1. 「Custom Databases」－「New Database(1) [Untitled\_1]」を右クリックし、[ Create New ]－[ Application Fix ] を選択します。



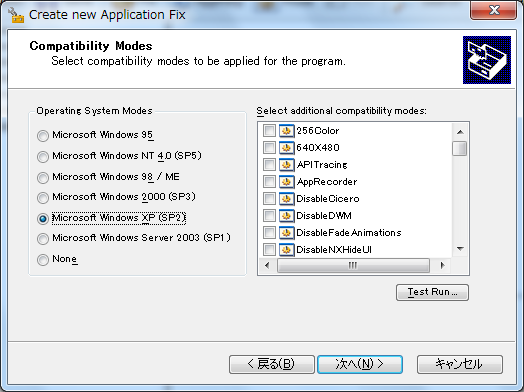
* 1. 互換フィックスの新規作成

1. 「Program information」画面で、互換フィックスを適用したいアプリケーションのパスなどを指定し、「次へ」をクリックします。



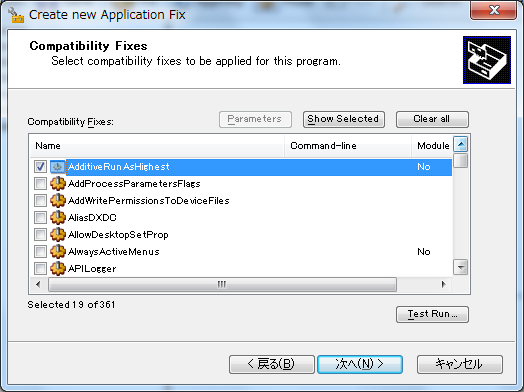
* 1. アプリケーションの選択

1. 「Compatibility Modes 」画面で、アプリケーションに必要なOS の種類を選択し、[ 次へ ] ボタンをクリックします。



* 1. OS と互換モードの選択

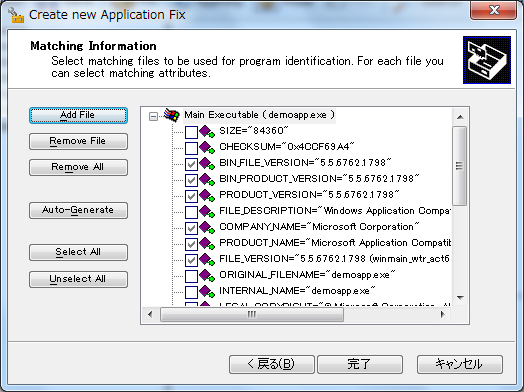
1. 「Compatibility Fixes」画面で、アプリケーションに必要な互換フィックスを選択し、[ 次へ ] ボタンをクリックします。



* 1. 互換フィックスの選択

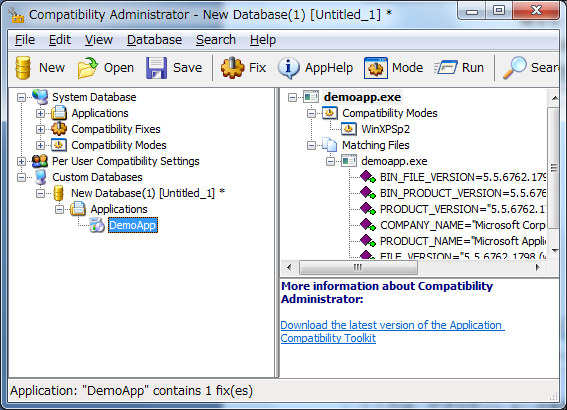
|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| ここで「Test Run」ボタンをクリックすると、指定した互換フィックス環境でアプリケーションをテスト実行することができます。 |

1. アプリケーションを識別するための情報が表示されます。必要があれば変更し、[完了]ボタンをクリックします。



* 1. アプリケーションの識別情報

1. 指定したアプリケーション用の識別情報や、互換フィックスなどが追加されます。



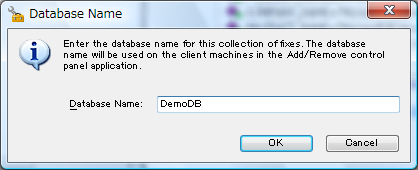
* 1. 登録されたアプリケーション

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| ここでは、アプリケーションの互換フィックスを作成する方法を紹介しましたが、同様の手順で、「アプリケーションヘルプ」や「互換モード」も作成することができます。     * 1. アプリケーションヘルプ・互換モードの作成 |

### カスタムデータベースファイルへの保存

作成した互換フィックスをファイルに保存します。保存することでフィックスのグループをパッケージ化して他の コンピューターに配布することができます。

1. Compatibility Administrator で、「Save」ボタンをクリックします。
2. 「Database Name」ダイアログボックスが表示されたら、名前を指定し、「OK」ボタンをクリックします。



* 1. カスタムデータベース名の指定

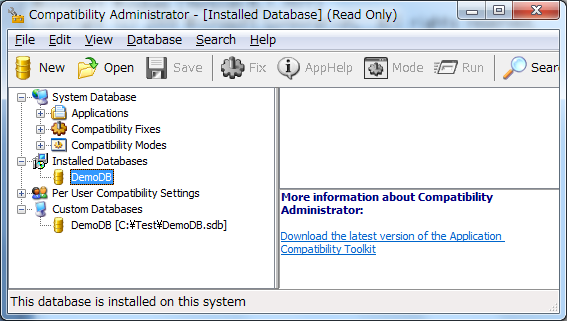
1. 「Save Database」ダイアログボックスで、ファイルパスや名前を指定し「保存」ボタンをクリックします。拡張子「.sdb」のファイル名で保存されます。

### コンピューターへの展開

1. 作成したカスタムデータベースファイル(.sdb) を各コンピューターに適用するため、以下のコマンドを実行します。引数には、sdbファイルのパスを指定します。

sdbinst.exe DemoDB.sdb

1. Compatibility Administrator ツールに戻ると、「Installed Databases」内に追加したカスタムデータベースが表示されます。



* 1. インストールされたカスタムデータベース

1. この後アプリケーションを実行すると、作成した互換フィックスが適用された環境で実行されます。

|  |
| --- |
| **E:\講習会コンテンツ\.NET3.5SP1\TrainingKit\Labs\AspNetAjax35Sp1\local\note.gifNote:** |
| 以下の方法を使用すると、組織内の多数のコンピューターに容易に展開することができます。   * Windows Deployment System * ログオンスクリプト * グループポリシー * System Management Server |

# 参考情報

本資料に掲載されている内容以外のWindows 7 対応アプリケーションに関する情報をお求めの方は下記のリンクをご覧ください。

* Windows 7 サイト

[http://www.microsoft.com/japan/Windows/windows-7/](http://www.microsoft.com/japan/Windows/windows-7/default.mspx)

* Windows 7 開発者ガイド

http://download.microsoft.com/download/E/5/A  
/E5AC514C-D324-4FD3-AA5B-2AF86CC48BC5  
/Windows\_7\_Developer\_Guide.docx

* [Windows 7 Application Quality Cookbook](http://code.msdn.microsoft.com/Windows7AppQuality) (英語)

http://code.msdn.microsoft.com/Windows7AppQuality/

* Windows アプリケーション品質CookBook:   
  アプリケーションの互換性、信頼性、パフォーマンスを高めるための開発者ガイド

http://download.microsoft.com/download/5/B/9  
/5B9EFD22-781D-4A4E-91DC-9C1FCF645020  
/Windows\_Application\_Quality\_Cookbook\_v\_09.docx

* Internet Explorer デベロッパー センター

http://msdn.microsoft.com/ja-jp/ie/

* Windows デベロッパー センター

[http://msdn.microsoft.com/ja-jp/windows/default.aspx](http://www.microsoft.com/japan/msdn/windowsvista/)

* Windows 互換性情報サイト

http://www.microsoft.com/japan/windowsvista/compatible/

* マイクロソフト イノベートオン プログラム

http://www.microsoft.com/japan/isv/innovateon/default.mspx

# 付録

描画モード切り替え表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| METAタグ  or応答ヘッダー  の指定 | DOCTYPE  フォームの終わりDOCTYPE内のURL | 表記の  有無 | 適用されるレンダリングモード | DOCTYPEの例 |
| 指定なし  または  IE=EmulateIE8 | DOCTYPE  なし | いずれの  場合も | クアークス | - |
| HTML  (no version) | いずれの  場合も | クアークス | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML //EN"> |
| HTML 2.0 | いずれの  場合も | クアークス | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 2.0 Final//EN"> |
| HTML 3.0 | いずれの  場合も | クアークス | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"> |
| HTML 4.0 | いずれの  場合も | IE8 標準  (CSS 2.1) | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"> |
| HTML 4.0  Frameset | URLあり | IE8 標準  (CSS 2.1) | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN“  "http://www.w3.org/TR/REC-html40/frameset.dtd"> |
| URLなし | クアークス | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN"> |
| HTML 4.0  Transitional | URLあり | IE8 標準  (CSS 2.1) | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN“  http://www.w3.org/TR/html4/loose.dtd"> |
| URLなし | クアークス | <!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.0 Transition//EN”> |
| HTML 4.0  Strict | いずれの  場合も | IE8 標準  (CSS 2.1) | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN”> |
| XHTML | いずれの  場合も | IE8 標準  (CSS 2.1) | <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN”> |
| XML | いずれの  場合も | IE8 標準  (CSS 2.1) | -<?xml version="1.0" encoding="UTF-8"?>  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"> |
| Unrecognized DOCTYPE | いずれの  場合も | IE8 標準  (CSS 2.1) | - |
| IE8 指定あり(IE=8) | DOCTYPEの  有無、内容に  かかわらず | いずれの  場合も | IE8 標準  (CSS 2.1) | - |
| IE7 指定あり(IE=7) | DOCTYPEの  有無、内容に  かかわらず | いずれの  場合も | IE7 標準 | - |
| 互換設定(IE=5) | DOCTYPEの  有無、内容に  かかわらず | いずれの  場合も | クアークス | - |
| 指定有り  (IE=Edge) | DOCTYPEの  有無、内容に  かかわらず | いずれの  場合も | IE8 標準  (CSS 2.1) | - |
| METAタグ  or応答ヘッダー  の指定 | DOCTYPE  フォームの終わりDOCTYPE内のURL | 表記の  有無 | 適用されるレンダリングモード | DOCTYPEの例 |
| IE7標準/  クアークス  (IE=EmulateIE7) | DOCTYPE  なし | いずれの  場合も | クアークス | - |
| HTML  (no version) | いずれの  場合も | クアークス | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML //EN"> |
| HTML 2.0 | いずれの  場合も | クアークス | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 2.0 Final//EN"> |
| HTML 3.0 | いずれの  場合も | クアークス | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"> |
| HTML 4.0 | いずれの  場合も | IE7 標準 | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"> |
| HTML 4.0 Frameset | URLあり | IE7 標準 | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN"  "http://www.w3.org/TR/REC-html40/frameset.dtd"> |
| URLなし | クアークス | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN"> |
| HTML 4.0 Transitional | URLあり | IE7 標準 | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"  "http://www.w3.org/TR/html4/loose.dtd”> |
| URLなし | クアークス | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"> |
| HTML 4.0 Strict | いずれの  場合も | IE7 標準 | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"  "http://www.w3.org/TR/html4/strict.dtd"> |
| XHTML | いずれの  場合も | IE7 標準 | <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"> |
| XML | いずれの  場合も | IE7 標準 | <?xml version="1.0" encoding="UTF-8"?>  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"> |
| Unrecognized DOCTYPE | いずれの  場合も | IE7 標準 | - |