



## Windows ユーザー エクスペリエンス ガイドライン

「何事も、あるものにとっては最高であり、他のものにとっては最悪である。

肝心なのは、何を、いつ、誰のために、なぜ行うのかを知ることだ。」 - ビル バクストン

この公式の Windows ユーザー エクスペリエンス ガイドライン(略して "UX ガイド")は、以下のことを目的としています。

- Windows ベースのすべてのアプリケーションにとって、質の高い、一貫性のある基準を設定する。
- ユーザー エクスペリエンスに関する特定の質問に答える。
- 仕事をより簡単にする。

### UX ガイドのダウンロードと印刷

多数のご要望にお応えし、UX ガイドを PDF 形式で配布します。

### ご意見、ご感想

ご意見、ご感想をお待ちしています。個別のご質問、コメント、ご要望については、[winui@microsoft.com](mailto:winui@microsoft.com)までお問い合わせください。

技術サポートについては、以下のとおりです。

- Windows の技術サポートについては、[Windows Vista サポート ページ](#)を参照してください。
- 特定のタスクに対するサポートについては、[Windows ヘルプと使い方](#)を参照してください。
- Windows に関するご意見、ご感想は、[Windows Vista のフィードバックに関するページ](#)を参照してください。
- 一般的なヘルプとサポートについては、[Microsoft サポート オンライン](#)にアクセスしてください。

日本語版 最終更新日: 2009 年 7 月 15 日

# ガイドライン

Windows® のユーザー エクスペリエンスに関する詳細なガイドラインは、以下のセクションにより構成されます。

- デザイン原則
- コントロール
- コマンド
- テキスト
- メッセージ
- 対話操作
- ウィンドウ
- 外観
- エクスペリエンス
- Windows 環境

## デザインの原則

次の原則を参照して、Windows® プログラムのユーザー エクスペリエンスのデザインに役立ててください。

- **Windows ユーザー エクスペリエンスのデザイン原則。** Windows 7 のデザインに使用された原則です。
- **よくあるガイドラインからの逸脱。** ユーザー インターフェイスのデザインで注意すべきいくつかの一般的なミスや不統一です。
- **優れたユーザー エクスペリエンスをデザインする方法。** デザインに関するさまざまなアドバイスを挙げています。
- **パワフルかつシンプルに。** 機能の選択と表示のバランスを入念にとることによって、機能的でシンプルなデザインを実現できます。
- **Windows Presentation Foundation を使ってデザイン。** Windows Presentation Foundation (WPF) を活用するためのガイドラインです。

## Windows ユーザー エクスペリエンスのデザイン原則

### コンセプトを減らして、信頼を高める

- 新しいコンセプトを導入しましたか？その理由は何ですか？それは必要ですか？
- 不要なコンセプトを削除できますか？
- 意味のある区別を行っていますか？
- UX は同じコンセプトを引き継いでいますか？

### どんなに小さなことも重要である

- 頻繁に見られる、または多くのユーザーが目にする、小さくても重要なことは何ですか？
- どんな小さな問題を解決していますか？
- 操作は少ないほどよい。
- エクスペリエンス内の些細なことも無視しません。
- 細かいところまでよく考えて計画します。
- 小さなバグも修正します。

### "外観" と "内容" を重視する

- UX は何を重視していますか？その外観には、重視していることが反映されていますか？
- ユーザーが目にする最初のものに、UX で重視していることが反映されていますか？
- UX は期待どおりですか？
- ユーザーができるることは明確ですか？
- 必要な手順のみを示していますか？

### 探し回ることのないように、見つけやすくする

- 気を散らすものを減らします。
- 機能が競合しないようにします。
- 新しい機能を使用するようにします。
- 以下は、見つけやすくするための解決策ではありません。
  - スタートメニューにアイコンを固定する。
  - デスクトップにアイコンを置く。
  - 通知領域にアイコンを置く。
  - 通知を使用する。
  - 最初の実行エクスペリエンスを設定する。
  - ツアーを設定する。

### 開始前の UX と質問

- 質問量を少なくします。
- 質問は 1 回にします。
- 構成による値の設定を要求しない。
- この質問は既にしていますか。
- 機会を見つけて質問を整理するようにします。

### カスタマイズではなく、個人設定にする

- その機能によって、ユーザーは自分自身用の要素を表現できますか？
- 個人設定とカスタマイズを区別していますか？
- 個人設定は、新しい機能として扱う必要がありますか、それとも既存の機能や情報(ユーザーの場所、背景写真、スタイルなど)を利用できますか？

### エクスペリエンスのライフサイクル

- 次のすべての段階で、ユーザー エクスペリエンスについて検討します。
  - インストールと作成
  - 最初の使用とカスタマイズ
  - 通常の使用
  - 管理とメンテナンス
  - アンインストールとアップグレード
- 12か月使用したものとしてエクスペリエンスを検証し、以下を満たしているかどうかを確認します。
  - 現実的な内容
  - 現実的な量

## モバイル ユーザー向けに作成する

- すべての UX の原則を、12 インチと 20 インチの画面サイズで同じように適用します。
- 中断可能にします。
- 開始と停止の理由を説明します(すばやく戻り、他の UX を妨げないようにします)。
- 接続と切断の理由を説明します。
- パフォーマンスは、UX 全体を魅力的なものにします。

## その他の資料

これらの原則が Windows 7 のデザイン プロセスでどのように使用されているかについては、以下を参照してください。

- [Windows 7 のデザイン原則に関するビデオ \(英語\)](#)
- [Windows 7 のデスクトップ エクスペリエンスのデザインに関するビデオ \(英語\)](#)

## 優れたユーザー エクスペリエンスをデザインする方法

以下の考えは簡単に説明されていますが、それぞれ深い意味を持っています。それを個別の記事にすることもできますが、ここではその代わりに短く説明します。足りない情報は、実際の経験から補つてください。

### 1. 基礎を強固にする

主要なシナリオ (ユーザーが Windows® プログラムを使用する主な理由) は、付随的なシナリオ (ユーザーが 実行する場合も、しない場合もあること) よりもはるかに重要です。基礎を固めることが大切です。また、基礎が固まっている場合は付随的な事柄はそれほど問題とされません。

### 2. 機能をデザインするのではなく、エクスペリエンスをデザインする

個々の機能をデザインするのではなく、エクスペリエンスを最初から最後までデザインします。そして、製品全体のエクスペリエンスを通して、デザインするに当たって設定した基準を維持します。たとえば、プログラムのセットアップが使いにくくてバグが多いものであると、ユーザーはプログラムそのものも使いにくくてバグが多いと考えます。そのように考えるのは当然のことです。

### 3. 特に得意な分野を持つ

"実際の" ユーザー (マーケティング部門や PR 部門といった "組織" ではなく) がプログラムをどう評価するか考えてみます。対象ユーザーを特定し、そのユーザーに「このプログラムは、A、B、C をとてもうまく処理できるので愛用しています」と言ってもらえるようにします。ユーザーにこのように言ってもらえない場合、何が問題なのでしょうか。"十分" という評価は、既に十分ではありません。ユーザーに愛されるプログラムを目指す必要があります。

### 4. 八方美人にならないようにする

すべてのユーザーを満足させようとするよりも、対象ユーザーを喜ばせようとする方が、プログラムは成功します。すべてに集中するということは、事実上不可能であることを忘れないでください。

### 5. 厳しく決断する

本当に必要な機能、コマンド、オプションであれば、しっかり作ります。そうでなければ、省きます。何でもオプションにしたり、構成可能なようにして、厳しい決断を避けてはなりません。

### 6. 打ち解けた会話のようなエクスペリエンスにする

UI は、設計者と対象ユーザーとの会話だと考えてデザインします。プログラムを使っているところを後ろから覗き込んでいることを思い浮かべてください。彼または彼女が「ここはどうすればいいの?」と聞いてきたとして、それに対する説明 (手順とその順番、使用する言語、説明する方法など) を考えます。この場合、あえて "説明しない" ことも考えられます。これが UI のあるべき姿です。つまり、よく読まないとわからぬような難解なものは避け、友人間の会話のようなものを目指します。

### 7. 既定で、適切な設定を行う

もちろん、ユーザーが設定を変更できるオプションをたくさん作成することができますが、必要性を考えます。安全で、セキュリティ保護された、便利な値を既定値に選択します。また、既定の設定でエクスペリエンスが対象ユーザーに適したものになるようにします。ユーザーは不適切な初期設定を修正するものだという前提に立たないでください。ユーザーは既定の設定を変えないものです。

### 8. 何もしなくても適切に動作するようにする

ユーザーは、さまざまなことを設定したり習得したいのではなく、プログラムを使用したいと思っています。初期設定を選択し、最も一般的で重要なタスクの実行方法を明確にしたら、すぐにプログラムが機能するようにします。

### 9. 質問は慎重に行う

重要でない質問はモーダル ダイアログ ボックスで行うこと避け、代わりにモードレス ダイアログ ボックスで行います。できれば、現在のコンテキストからユーザーの意図をくみ取ることによって、質問が必要になる機会を減らします。UI で質問をする必要がある場合は、テクノロジーの観点からではなく、ユーザーの目的とタスクの観点から示します。ユーザーが理解でき、明確に区別できるオプションを (ここでも、テクノロジーではなく目的とタスクの観点から) 提供します。ユーザーが情報に基づいた判断を行うことができるように、十分な情報を提供するようにします。

### 10. 楽しく使用できるようにする

プログラムが十分にその役割を果たせるようにします。適切な機能セットを用意し、その機能を適切な場所に配置します。詳細に注意し、すべてが洗練された状態になるようにします。ユーザーは細かいことには気付かないものなどと思ってはなりません。必ず気付かれます。

### 11. 見た目を楽しくする

標準の ウィンドウ 枠、フォント、システム カラー、コモン コントロール、ダイアログ ボックス といった

Windows の標準的な外観、および標準的なレイアウトを使用します。カスタム UI の使用は避け、ブランド化は控えめにします。法的に有効な場合に限り、できるだけ、標準の Windows アイコン、グラフィック、アニメーションを使用します。独自のグラフィックやアイコンについては、プロのデザイナーに作成を依頼します。デザイナーに依頼する余裕がない場合は、シンプルなグラフィックを少しだけ使用するか、グラフィックを一切使用しないようにします。

また、スキンを提供することで平凡な外観をフォローできると考えないでください。ユーザーの多くはスキンにこだわりがありません。ほどほどの外観が大量に提供されているよりは、優れた外観が 1 つある方が良い印象を与えることができます。

## 12. 応答性を向上させる

プログラムの応答性は、エクスペリエンス全体にとってきわめて重要です。必要以上に時間がかかる応答性の悪いプログラムだと、ユーザーは使用に適さないプログラムだという印象を持ちます。すべてのパフォーマンスが重要な機能について、ユーザーの目的と期待をまず理解し、その目的を達成できる最も軽量なデザインを選択します。一般的に 10 秒以上かかるタスクについては、処理状況のフィードバックとキャンセル機能が必要になります。ユーザーが処理速度をどのように感じるかは、実際の処理速度と同じくらい重要です。ユーザーが感じる処理速度は、主にプログラムが応答可能になる早さによって決まります。

## 13. シンプルさを維持する

目的を果たすことができる範囲で、最もシンプルなデザインを目指します。ただ "必要" であるというだけのものを超えるデザインにします。何かを実行する方法が 1 つで済む場合に、3 つの方法を用意しないようにします。不要な機能はすべて排除または縮小します。

## 14. 不適切なエクスペリエンスを避ける

口で言うほど簡単ではありませんが、プログラムに対するユーザーの全体的な印象は、良いエクスペリエンスによってではなく、悪いエクスペリエンスによって決まることが多いものです。

## 15. 一般的な問題に対処したデザインにする

そのデザインは、ユーザーが操作を誤ったり、ネットワーク接続が途切れた状況でも、問題のないデザインでしょうか。一般的な問題、ユーザーの誤操作、その他のエラーを想定してデザインします。ネットワークが遅い/使用できない場合、デバイスがインストールされていない/使用できない場合、ユーザーの入力が誤っている場合、ユーザーが手順をスキップした場合、といった状況を検討します。プログラムの各ステップで、起こりうる最悪の事態は何かと自問してみます。そして、それが本当に起きた場合にプログラムがどのように動作するのがよいか考えます。すべてのエラーメッセージで、問題を明確に説明し、実行可能な解策を提示するようにします。

## 16. わずらわしさを排除する

ほとんどの場合、ユーザーが日常的に無視して何の行動もとらないものはすべて、デザインの変更か削除が必要です。これは、ユーザーが繰り返し目にするもの(エラーメッセージ、警告、確認、通知)について、特に当てはまります。サウンドは、特に控えめに使用します。考えられる例外としては、セキュリティや法的・事項に関する UI(同意やライセンス条項など)があります。

## 17. 労力、知識、思考を減らす

プログラムを使用するために必要な労力、知識、思考を削減するには、以下のガイドラインに従います。

- 暗黙に示すのではなく明示します。ユーザーに伝える必要がある情報は、画面に直接表示します。ウィンドウやページ上のメイン指示テキストを慎重に作成し、UI の目的を明確に伝えます。
- 手動ではなく自動にします。実用的であり、望ましい場合は、できるだけ自動化することにより、ユーザーを支援します。簡単なテストとしては、プログラムを閉じ、再始動してから、最も一般的なタスクを実行してみます。手動の操作をいくつ削減できるかを確認します。
- 冗長さをなくして簡潔にします。画面上に配置するときは、簡潔にします。要点だけを伝えます。テキストは、精読用ではなく流し読みできるようにデザインします。補完的な役立つ情報で、必須でないものについては、ヘルプリンクを使用します。
- 無制限にはせず、制約を設けます。コントロールを選択する際は、通常、有効な入力しか受け付けないコントロールを選択するのが最適です。
- 無効化するより、有効なままにします。無効化されたコントロールは混乱を招くことが多いので、なぜコントロールが使用できないのかをユーザーが簡単に推測できるときにのみ使用します。それ以外の場合は、該当しないコントロールを削除するか、または有効なままにしてわかりやすいフィードバックを提供します。
- 情報をできるだけ保持します。セキュリティやプライバシーに関する事項を除き、ユーザーが以前に行った入力や行動を保持して、同じことを簡単に実行できるようにする方が毎回最初からユーザーに操作させるよりも優れています。

- 手掛けりがない状態にはせず、フィードバックします。タスクが実行されたのか失敗したのかを示す、明確なフィードバックを提供します。ユーザーが推測しなくても済むようにします。

## 18. ガイドラインに従う

当然のことですが、UX ガイドを、Windows ベースのプログラムに関する最小限の品質および一貫性の基準と考えてください。UX ガイドを使用して、ベスト プラクティスに従い、日常的な決定を行って、作業をより簡単にします。どのようなプログラムであっても、創造的なエネルギーは日常的なことではなく重要なことに集中させます。だれも使い方がわからないような、風変わりなプログラムは作成しないでください。ガイドラインに従って、受け入れられやすく、同時に他より目立つエクスペリエンスにします。

### UI のテストを行う

[ユーザビリティ調査](#)で実際の対象ユーザーによるテストが済むまでは、プログラムが正しくできているかどうかはわかりません。ほとんどの場合、結果に対して(悪い意味で)驚くことになります。UI について批判を受けるのは喜ばしいことです。これは、最高のものを作るには必要なことです。プログラムをリリースした後は、必ずフィードバックを集めましょう。

## よくあるガイドラインからの逸脱

ウィンドウ  
レイアウト  
テキスト  
コントロール  
キーボード  
マウス  
ダイアログ ボックス  
プロパティ シート  
ウィザード  
ウィザード ページ  
エラーメッセージ  
警告メッセージ  
確認  
アイコン  
ヘルプ

ここでは、UX ガイドにあるガイドラインに関して最もよくある違反をまとめています。これをチェックリストとして利用し、プログラムのユーザーインターフェイスについて、以下の重要なアイテムが適切に設定されていることを確認してください。

### ウィンドウ

- Windows の最小有効解像度 ( $800 \times 600$  ピクセル) をサポートします。セーフ モードで動作する必要のある重要なユーザーインターフェイス (UI) では、 $640 \times 480$  ピクセルの有効解像度をサポートします。タスクバーと共に表示するウィンドウについては、縦方向に 48 相対ピクセル 短くして、タスクバー用の領域を確保します。
- サイズ変更可能なウィンドウのレイアウトを  $1024 \times 768$  ピクセルの有効解像度に最適化します。より低い解像度の場合にも機能するように、これらのウィンドウのサイズが自動的に変更されるようにします。
- 96 dpi (ドット/インチ) の  $800 \times 600$  ピクセル モード、120 dpi の  $1024 \times 768$  ピクセル モード、144 dpi の  $1280 \times 1024$  ピクセル モードで、必ずウィンドウをテストします。コントロール、テキスト、ウィンドウなどが切り詰められていないか、アイコンやビットマップが引き伸ばされたりしていないかなど、レイアウト上の問題をチェックします。
- タッチ PC やモバイル PC 向けシナリオのプログラムでは、120 dpi に最適化します。現在、タッチ PC やモバイル PC では高 dpi 画面が普及しています。
- 所有されるウィンドウの初期表示は、オーナー ウィンドウ前面の "中央配置" にします。下には配置しません。その後の表示については、最後に表示された位置 (オーナー ウィンドウに対する相対的な位置) に表示する方が使いやすいと思われる場合は、その表示方法も検討します。
- コンテキスト ウィンドウの場合は、必ず起動元オブジェクトの近くに表示します。ただし、表示するウィンドウで起動元のオブジェクトが隠れることがないように、適切な位置に配置します (できれば右下にオフセットします)。

### レイアウト

- ウィンドウ内のコントロールや第 2 ウィンドウのサイズを標準的なコンテンツに合うように指定します。テキストが切り詰められないようにし、切り詰められたことを示す省略記号が表示されないようにします。標準的なコンテンツを表示するためにユーザーがウィンドウを操作しなくて済むようにします。非常に大きなコンテンツを扱う場合は、サイズ変更やスクロールができるようにします。特に、以下の点を確認します。
  - コントロールのサイズ。コントロールのサイズを標準的なコンテンツに合うように設定し、必要に応じて、幅や高さを増やしたり、複数行にしたりします。利用できる領域が広いウィンドウでは、スクロールをなくすか、スクロール幅が縮小されるようにコントロールのサイズを設定します。また、ラベルの表示が欠けたり、利用可能な領域が広いウィンドウ内でテキストの表示が欠けたりしないようにします。ただし、テキストを読みやすくするには、行幅を 65 文字に制限するようにします。
  - 列幅。リスト ビューの列に、最適な既定のサイズ、最小サイズ、および最大サイズが表示されるようにします。特にリスト ビュー内に利用可能な領域がある場合は、テキストの表示が欠けないような列幅を既定にします。
  - レイアウトのバランス。ウィンドウのレイアウトは、左右でおよそのバランスがとれているようにします。レイアウトが左に偏る場合は、コントロールを横に広げて、一部のコントロールを右方向へ移動させるようにします。

- レイアウトのサイズ変更。ウィンドウのサイズが変更可能で、データの表示が欠ける場合は、より多くのデータが表示されるようにウィンドウのサイズを大きくなります。データが欠けていると、ユーザーは、ウィンドウのサイズを変更してより多くの情報を表示させようと考えます。
- コンテンツが利用できなくなる下限のサイズがある場合は、ウィンドウの最小サイズを設定します。サイズ変更可能なコントロールに対しては、機能を維持できる最小サイズを設定します(リストビューが役目を果たすことができる最小の列幅など)。

## テキスト

- できる限り、会話で使用するような一般的な用語を使用します。テクノロジーではなく、ユーザーの目的を重視します。これは、複雑な技術的概念や操作を説明する場合に、特に効果的です。ユーザーの肩越しにのぞき込み、タスクの実行方法を説明している様子を想像します。
- ていねいに、サポートするように、また、やる気を起こさせるようにします。見下され、非難され、威圧されているとユーザーが感じることのないようにします。
- 冗長なテキストは削除します。ウィンドウ タイトル、メイン指示テキスト、補足指示テキスト、コンテンツエリア、コマンド リンク、コミット ボタンに冗長なテキストがないことを確認します。通常、メイン指示テキストと対話型コントロールのテキストはそのまま残し、他の部分にある冗長なテキストを削除します。
- タイトルにはタイトルスタイルの大文字化を、その他のすべての UI 要素にはセンテンス スタイルの大文字化を使用します。これにより、Windows® のトーンにより適したテキストになります。
  - 例外: 古いアプリケーションでは、大文字/小文字のスタイルが混在しないように、コマンド ボタン、メニュー、列見出しにタイトルスタイルの大文字化を採用することもできます。
- 機能名やテクノロジーネームについては、大文字化は控えめにします。一般には、主要なコンポーネントのみを大文字化します(タイトルスタイルの大文字化を使用)。
- 機能名およびテクノロジーネームについては、大文字化に一貫性を持たせます。その名前が 1 つの UI 画面に何度も出現する場合、常に同じように表示する必要があります。同様に、プログラム内のすべての UI 画面で、その名前を同じように表示する必要があります。
- 汎用的なユーザー インターフェイス要素名は大文字で表記しません。たとえば、"toolbar (ツールバー)"、"menu (メニュー)"、"scroll bar (スクロールバー)"、"button (ボタン)"、"icon (アイコン)" などがあります。
  - 例外: アドレスバー (Address bar)、リンクバー (Links bar)、リボン (ribbon)。
- キーボード上のキーを表すのにすべて大文字の表記は使用しません。標準キーボードで使用されている表記に従うか、ラベル表示のないキーボードキーの場合は小文字で表記します。
- 省略記号"..."は不完全であることを意味します。次の UI テキストで、省略記号を使用します。
  - コマンド: コマンドに追加の情報が必要であることを示します。省略記号は、操作によって別ウィンドウが表示される場合に使用するのではなく、追加の情報が必要な場合にのみ使用します。詳細設定、ヘルプ、オプション、プロパティ、設定など、ラベルの内容によって別のウィンドウが表示されることが明らかにわかるコマンドについては、省略記号を付ける必要はありません。
  - データ: テキストの一部が表示されていないことを示します。
  - ラベル: タスクが進行中であることを示します ("検索しています..." など)。

**ヒント:** ウィンドウやページに未使用の領域があるにもかかわらずテキストの切り詰めがある場合は、レイアウトに問題があるか、既定のウィンドウ サイズが小さすぎることを示しています。レイアウトや既定のウィンドウ サイズを修正して、テキストがすべて表示されるようにするか、より多くのテキストが表示されるようにしてください。詳細については、「[レイアウト](#)」を参照してください。

- リンク以外のテキストを青色にしないようにします。ユーザーがリンクと誤解する可能性があります。色付きテキストは使わずに、太字または灰色を使います。
- 太字は、必読のテキストに注目を集めるために、控えめに使用します。
- 表示されたウィンドウまたはページでユーザーがすべきことを簡潔に説明するために、メイン指示テキストを使用します。優れたメイン指示テキストは、単に UI 操作に注目させるだけではなく、ユーザーの目的を明確にするものになっています。
- メイン指示テキストは、必須の指示や具体的な質問の形式で表現します。
- コントロール ラベルやメイン指示テキストの最後には、句点を付けません。
- 文と文との間には空白を 1 つ配置します(英語の場合)。2 つではありません。

## コントロール

- 全般
  - すべてのコントロール、コントロールのグループにラベルを設定します。次の場合は例外です。
    - テキストボックスおよびドロップダウンリストには、プロンプトを使用してラベルを設定することができます。
    - 従属コントロールには、関連するコントロールのラベルを使用します。スピンコントロールは、常に従属コントロールになります。
  - すべてのコントロールの既定値は、最も安全(データの消失やシステムアクセスが失われるなどを防ぐ)で、最もセキュリティの高い値にします。安全性やセキュリティを考慮する必要がない場合は、最もよく使用される値か、最も便利な値を選択します。
  - できるだけ制約付きコントロールを使用します。テキスト入力の負担を減らすために、テキストボックスのような制約のないコントロールではなく、リストやスライダーといった制約付きコントロールをできる限り使用します。
  - 無効化されているコントロールは検討し直します。無効化されているコントロールがあると、ユーザーは無効化されている理由を推測しなければならなくなり、使用しにくくなる場合があります。ユーザーが使用することが見込まれ、無効になっている理由を簡単に推測できる場合に、コントロールを無効化します。ユーザーが有効にできないコントロール、またはユーザーが使用することが想定されないコントロールは削除するか、有効なままにして、誤って使用された場合にエラー メッセージが表示されるようにします。
    - ヒント: コントロールを無効にするか、またはエラー メッセージを表示するかで迷った場合は、まず実際にエラー メッセージを作成してみてください。対象ユーザーが簡単に思い付かないような役立つ情報がエラー メッセージに含まれている場合は、コントロールを有効にしたままで、エラー を表示するようにします。その他の場合は、コントロールを無効にします。
- コマンド ボタン
  - ラベルは汎用的な内容ではなく、具体的になるようにします。ラベル以外のものを読まなくてもユーザーが意味を理解できることが理想です。一般にユーザーは、静的テキストよりも、コマンド ボタンのラベルを優先的に読む傾向があります。
    - 例外: [キャンセル] ボタンのラベルは、何を取り消すのかがあいまいであっても変更しないでください。ユーザーがすべてのラベルを読まなくても、どのボタンが操作を取り消すものであるかがわかるようにするためにです。ただし、保留中の処理が複数ある場合など、どの操作が取り消されるのか明確でない場合は、別のラベルを使用してかまいません。
  - 問い合わせに対する応答ボタンは、その内容に合ったラベルにしてください。たとえば、"はい" か "いいえ" で答える質問であれば、[はい]/[いいえ] ボタンを用意します。
  - プロパティ シートやコントロール パネル アイテム以外のダイアログ ボックスで、[適用] ボタンを使用しないでください。[適用] ボタンをクリックすると、保留中の変更が適用されますが、ウィンドウは開いたままになります。こうすることで、ユーザーがウィンドウを閉じる前に変更内容を評価できるようになります。ただし、[適用] ボタンが必要なのは、プロパティ シートとコントロール パネル アイテムだけです。
  - "キャンセル" というボタンのラベルは、取り消すことによって環境を以前の状態に(副次的な影響を残さず)戻せる場合に使用します。これ以外の場合は、変更された現在の状態が維持されることを示すために、ボタンのラベルは "閉じる" (処理が完了した場合) または "停止" (処理が進行中の場合) を使用します。
- コマンド リンク
  - コマンド リンクは常に 2 つ以上のセットで提示します。応答が 1 種類しかない質問は論理的に意味をなしません。
  - 明示的な [キャンセル] ボタンを用意します。この用途でコマンド リンクを使用しないでください。タスクを実行する必要がないことにユーザーが気付くことはかなり頻繁にあります。コマンド リンクを取り消しに使用すると、どのコマンド リンクが取り消しを意味するか判断するために、すべてのコマンド リンクをよく読む必要があります。[キャンセル] ボタンがあれば、ユーザーは効率よくタスクを取り消すことができます。
  - 明示的な [キャンセル] ボタンを用意するとコマンド リンクが 1 つだけになってしまう場合は、取り消すためのコマンド リンクと [キャンセル] ボタンの両方を用意します。これにより、選択肢があることがはっきりとユーザーに伝わります。このコマンド リンクでは、単に "キャンセル" またはそれに類する語句ではなく、もう一方の選択肢との対比が明らかな語句を使用します。

- [今後、この<アイテム>を表示しない] チェック ボックス
  - [今後、この<アイテム>を表示しない] というオプションを使って、同じダイアログ ボックスが繰り返し表示されないようにユーザーが設定できるようにすることを検討します。ただし、これは他に良い手段がない場合に限ります。ユーザーが本当に必要としている場合は、ダイアログ ボックスを常に表示することをお勧めします。必要としていない場合は、単純に削除します。
  - <アイテム>を具体的なアイテムに置き換えます。たとえば、[今後、この通知を表示しない] とします。ダイアログ ボックスを指す場合は、一般的に、[今後、このメッセージを表示しない] を使用します。
  - ユーザー入力を今後の既定値として使用する場合は、その旨を明確に示します。オプションの下に、"選択内容は、今後既定値として使用されます" という文を追加します。
  - このオプションは既定でオンにしません。ダイアログ ボックスを本当に1回だけ表示する必要がある場合は、確認せずに1回だけ表示します。このオプションをユーザーの手をわざらわせることの言い訳に使用しないでください。既定の動作がわざらわしいものでないことを確認してください。
  - ユーザーがこのオプションをオンにして [キャンセル] をクリックした場合、このオプションは "有効" になります。この設定はメタオプションであるため、副次的な影響を残さないという [キャンセル] の標準の動作には従いません。ユーザーが今後そのダイアログ ボックスを表示しないようにする場合、そのダイアログ ボックス自体もキャンセルする可能性が高いことに注意してください。
- リンク
  - アクセス キーは割り当てません。リンクには Tab キーでアクセスできます。
  - リンク テキストに "クリック" や "ここをクリック" は追加しません。リンク自体がクリックすることを示唆しているため、こうした言葉は必要ありません。
- ツールヒント
  - ラベルのないコントロールにラベルを付けるには、ツールヒントを使用します。一貫性を保つためだけで、ラベル付きコントロールにツールヒントを設定する必要はありません。
  - ツールバーのボタンにラベルが付けられている場合でも、有益であれば、ツールヒントで詳細を提供してもかまいません。既にラベルにある情報を繰り返したり言い直すことはしません。
  - ユーザーが確認または操作しようとするオブジェクトがヒントで隠れないようにします。ポインターとヒントが離れることになっても、ヒントは常にオブジェクトの脇に配置します。オブジェクトとヒントとの関係性が明らかである限り、多少離れても問題はありません。
    - 例外: リストやツリーで使用される、完全な名前を表示するツールヒント。
  - アイテムの集まりに対しては、次に来るオブジェクトをユーザーが確認または操作する可能性がある場合、オブジェクトが隠れないようにします。水平にアイテムが並んでいる場合は、右にヒントを配置しないようにし、垂直にアイテムが並んでいる場合は、下にヒントを配置しないようにします。
- 段階的表示
  - ユーザーが通常は必要としない高度なオプションおよびコマンド、使用頻度の低いオプションおよびコマンド、または詳細情報を非表示にするには、詳細表示/簡易表示を切り替える段階的表示 ボタンを使用します。一般的に使用される項目は非表示にしないでください。ユーザーが見つけられない可能性があります。ただし、非表示になっているものには必ず価値があるようにします。
  - 何らかのオプション、コマンド、または詳細が常に表示される場合は、以下のラベルの組み合わせを使用します。
    - オプションの詳細表示/オプションの簡易表示。オプションに、またはオプション、コマンド、詳細が混在する場合に使用します。
    - コマンドの詳細表示/コマンドの簡易表示。コマンドにのみ使用します。
    - 詳細表示/簡易表示。情報にのみ使用します。
    - <オブジェクト名> の詳細表示/<オブジェクト名> の簡易表示。フォルダーなど、上記以外のオブジェクトの種類に使用します。
  - または:
    - オプションの表示/オプションの非表示。オプションに、またはオプション、コマンド、詳細が混在する場合に使用します。
    - コマンドの表示/コマンドの非表示。コマンドにのみ使用します。
    - 詳細の表示/詳細の非表示。情報にのみ使用します。
    - <オブジェクト名> の表示/<オブジェクト名> の非表示。フォルダーなど、上記以外のオブジェ

クトの種類に使用します。

- **進行状況バー**
  - 処理時間の長さが有限である場合は、確定型の進行状況バーを使用します。これは、時間を正確に予測できない場合であっても同じです。不確定型の進行状況バーでは、進行中であることが示されますが、その他の情報は示されません。正確に予測できない可能性があるという理由だけで、不確定型の進行状況バーを選択しないでください。
  - 残り時間を正確に見積もることができるのであれば、それを提供します。正確に見積もられた残り時間は有用ですが、見積もりがまったくの見当違いであったり、頻繁に大きく変わる場合は役に立ちません。正確な見積もりを提供するために、何らかの処理を実行する必要がある場合は、該当する場合は、処理の初期段階では、不正確な可能性のある見積もりを表示しないようにします。
  - バーの進行を再開させないでください。おそらく処理の 1 つのステップが完了したという理由だと思いますが、バーの進行を最初から再開させると、進行状況バーの価値は失われます。これでは、ユーザーは処理がいつ完了するのかわかりません。代わりに、処理内のすべてのステップで進行状況を分割し、進行状況バーを 1 回で完了させます。
  - 有用な進行状況の詳細を提供します。進行状況の追加情報を提供します。ただし、ユーザーがその情報に基づく何らかの作業を行うことができる場合に限ります。テキストは、ユーザーが読むのに十分な時間、表示されるようにします。
  - 進行状況バーとビギー ポインターを組み合わせて使用しません。どちらかのみを使用し、両方を同時に使用しないでください。
- **プロンプト**
  - テキスト ボックスがツールバーのような特別な領域にあり、ラベルや指示がない方が良い場合は、プロンプトを使用します。
  - プロンプトは、テキスト ボックスまたはコンボ ボックスの目的を省スペースで示す場合に、主に使用します。コントロールの使用中にユーザーが確認する必要のある重要な情報は、プロンプトでは示しません。
  - プロンプト テキストと実際に入力されたテキストが混同されないようにします。そのためには、次のこと留意します。
    - プロンプト テキストは灰色で表示し、実際に入力されたテキストは通常の黒色で表示します。
    - プロンプト テキストは編集可能にしません。ユーザーがテキスト ボックス内をクリックしたり、`Tab` キーでテキスト ボックスに移動したら、画面からプロンプト テキストを消去します。
      - 例外: テキスト ボックスに既定で入力フォーカスがある場合は、プロンプトを表示し、ユーザーが入力を開始したときに画面から消去します。
  - 末尾に句読点や省略記号は付けません。
- **通知**
  - 通知は、現在のユーザー操作に無関係で、ユーザーがすばやく対応する必要がなく、無視してもかまわないイベントに対して使用します。
  - 通知を乱用しないでください。
    - 通知は、必要な場合に限って使用します。通知を表示すると、ユーザーの作業を中断したり、ユーザーに不快感を与える可能性があります。正当な理由で中断が行われるようにします。
    - 通知は、ユーザーがすばやく対応する必要がない、重大ではないイベントまたは状況に使用します。ユーザーがすばやく対応する必要のある重大なイベントまたは状況の場合は、他の UI 要素 (モーダル ダイアログ ボックスなど) を使用します。
    - 機能の告知に通知を使用しないでください。

## キー ボード

- ユーザーが最初に操作する可能性の高いコントロールに最初の入力フォーカスを割り当てます。多くの場合、最初の対話型コントロールに割り当てます。最初の対話型コントロールが適していない場合、ウィンドウのレイアウトを変更することを検討します。
- 読み取り専用の編集ボックスを含む、すべての対話型コントロールにタブ ストップを割り当てます。次の場合は例外です。
  - 1 つのコントロールとして動作する、関連するコントロールのグループ (ラジオ ボタンなど)。このようなグループには 1 つのタブ ストップが割り当てられます。

- 方向キーでグループ内を前後に移動したときに、フォーカスがグループ内から外に出ないように、グループを適切に設定します。
- タブオーダーは左から右、上から下の順序にします。一般的に、タブオーダーは読む順序と同じにします。よく使用するコントロールは、例外的にタブオーダーの早い方に設定することを検討します。Tabキーでは、止まることなく両方向に、すべてのタブストップを循環するようにします。グループ内では、タブオーダーに例外を設けず、順番どおりにします。
- タブストップ内では、方向キーは、左から右、上から下へ移動するようにします。例外は設けません。方向キーでは、止まることなく両方向に、すべてのアイテムを循環するようにします。
- 次の順番でコミットボタンを配置します。
  - [OK]/[実行する]/[はい]
  - "実行しない"/[いいえ]
  - [キャンセル]
  - [適用] (該当する場合)

"実行する" および "実行しない" には、メイン指示テキストに対する具体的な応答が入ります。

- アクセスキーとショートカットキーを混同しないでください。アクセスキーとショートカットキーは両方ともキーボードで UI にアクセスできますが、それぞれ異なる目的とガイドラインがあります。
- 可能な限り、すべての対話型コントロールまたはそのラベルに一意のアクセスキーを割り当てます。[読み取り専用のテキストボックス](#)は対話型コントロールである (ユーザーがスクロールし、テキストをコピーできる) ため、アクセスキーを割り当てるメリットがあります。以下には、アクセスキーを割り当てないでください。
  - [OK]、[キャンセル] および [閉じる] の各ボタン。Enter キーおよび Esc キーが、それぞれのボタンのアクセスキーに使用されています。ただし、"OK" または "キャンセル" を意味する異なるラベルのコントロールには、アクセスキーを常に割り当てます。
  - ショートカットキーを最もよく使用するコマンドに割り当てます。使用頻度の低いプログラムや機能の場合、代わりにアクセスキーが使用されるので、ショートカットキーは必要ありません。
  - ショートカットキーがタスクを実行する唯一の方法にならないようにします。マウスやキーボードの Tab キー、方向キー、アクセスキーも使用できるようにする必要があります。
  - よく知られているショートカットキーに別の機能を割り当てないようにします。よく知られているショートカットキーはユーザーに記憶されているので、機能に一貫性がないと、ストレスやエラーの原因になります。Windows プログラムで使用され、よく知られているショートカットキーについては、「[Windows のキーボードショートカットキー](#)」を参照してください。
  - システム全体に作用するショートカットキーをプログラムに割り当てないようにします。プログラムのショートカットキーは、プログラムに入力フォーカスがある場合にのみ効果を発揮します。

## マウス

- オブジェクトがクリック可能かどうかを判断するために、実際にクリックすることをユーザーに要求しないようにします。オブジェクトを見ただけで、クリックできるかどうかが判断できるようにします。
  - プライマリ UI (コミットボタンなど) には、静的なクリックアフォーダンスを設定します。ユーザーは、プライマリ UI を判断するのにマウスポインターを合わせる必要がなくなります。
  - セカンダリ UI (副コマンドや段階的表示コントロールなど) では、ポイント時にクリックアフォーダンスを表示できます。
  - テキストリンクでは、リンクテキストを静的に示唆し、マウスポインターを合わせるとクリックアフォーダンス (ポインターを手の形に変更し、下線やその他の表示を変更する) を表示するようにします。
  - グラフィックリンクでは、ポイント時に手の形のポインターを表示するだけにします。
- 手の形の (または "リンク選択" の) ポインターは、テキストリンクおよびグラフィックリンクに対してのみ使用します。このようにしないと、ユーザーはオブジェクトがリンクであるかどうかを判断するために実際にクリックしてみなければならなくなります。

## ダイアログボックス

- モーダルダイアログボックスでは対話操作を要求するため、ユーザーがタスクを続行する前に応答する必要があるタスクに使用します。作業を続行する前に完了させる必要がある、重要または頻度が低い、1回限

りのタスクなど、正当な理由で中断が行われるようにします。それ以外のタスクでは、モードレス ダイアログ ボックスの使用を検討します。

- モードレス ダイアログ ボックスは対話操作を要求しないため、ダイアログ ボックスとオーナーウィンドウを切り替える必要がある場合に使用します。モードレス ダイアログ ボックスは、頻度が高く、繰り返し発生する、継続的なタスクに最も適しています。ただし、リボン、ツールバー、およびパレット ウィンドウの方が適切な方法であることもあります。

## プロパティ シート

- プロパティの必要性を確認します。デザイン上の難しい判断を敬遠して、プロパティ ページに不要なプロパティを詰め込まないようにしてください。
- テクノロジーではなく、ユーザーの目的という観点からプロパティを表示します。プロパティは特定のテクノロジーを構成するためのものですが、だからといって、そのテクノロジーの観点からプロパティを表示する必要はありません。
  - テクノロジーの観点から設定を表示する必要がある場合(ユーザーがテクノロジーの名前を把握している場合など)は、ユーザーの利点を簡潔に説明するテキストを追加してください。
- 限定的で意味のあるタブ ラベルを使用します。"全般"、"詳細設定"、"設定"など、どのタブにも当てはまるような汎用的なタブ ラベルの使用は避けます。
- [全般] ページはできるだけ使用しないようにします。[全般] ページは、必ずしも必要ではありません。[全般] ページは次の場合にのみ使用します。
  - プロパティが複数のタスクに適用され、ほとんどのユーザーに対して重要な場合。[全般] ページには、特殊なプロパティや詳細なプロパティを配置しません。ただし、[全般] ページのコマンド ボタンを通じてこれらにアクセスできるようにすることができます。
  - プロパティをより具体的なカテゴリに分けられない場合。カテゴリに分けられる場合は、その名前をページに使用します。
- [詳細設定] ページはできるだけ使用しないようにします。[詳細設定] ページは次の場合にのみ使用します。
  - プロパティが一般的ではないタスクに適用され、主に詳しい知識のあるユーザーにとって重要な場合。
  - プロパティをより具体的なカテゴリに分けられない場合。カテゴリに分けられる場合は、その名前をページに使用します。
- プロパティ ウィンドウにタブが1つしかなく、拡張不可能である場合は、タブは使用しません。代わりに、[OK] ボタン、[キャンセル] ボタン、そして場合によっては[適用] ボタンを持つ通常のダイアログ ボックスを使用します。ただし、拡張可能なプロパティ ウィンドウ(サードパーティによる拡張が可能なウィンドウ)の場合は、必ずタブを使用する必要があります。

## ウィザード

- 最初は、ダイアログ ボックス、作業ウィンドウ、単一ページなどの軽量コントロールについて検討します。ウィザードは重い UI であり、複数の手順から成る、あまり実行しないタスクに最も適しています。ウィザードは必ずしも使用する必要はなく、役立つ情報やヘルプはどんな UI でも提供できます。
- [次へ] は、未確定のまま次のページに移動する場合にのみ使用します。[戻る] や [キャンセル] をクリックして結果を元に戻せない場合は、次のページに進むことは確定と見なされます。
- [戻る] は、間違いを修正する場合にのみ使用します。間違いの修正以外では、タスクを進めるために[戻る] をクリックすることをユーザーに要求しないようにします。
- タスクを確定する場合、メイン指示テキストに対する具体的な応答であるコミット ボタン([印刷]、[接続]、[開始]など)を使用します。タスクの確定に、確定を意味しない[次へ] や、具体的ではない[完了]などの汎用的なラベルを使用しないでください。コミット ボタンのラベルは、単独で意味のわかる名前にする必要があります。コミット ボタンのラベルは常に動詞から始めます(英語の場合)。次の場合は例外です。
  - [保存]、[選択]、[取得]など、"具体的な応答" が汎用的な場合は、[完了] を使用します。
  - 具体的な設定/設定群を変更する場合は、[完了] を使用します。
- コマンド リンクは、確定ではなく選択する場合にのみ使用します。具体的なコミット ボタンは、ウィザードのコマンド リンクに比べて、確定する行為であるということをより明確に示すことができます。
- コマンド リンクを使用している場合、[次へ] ボタンは非表示にしますが、[キャンセル] ボタンはそのまま表示します。
- [閉じる] は、フォローアップ ページや完了ページで使用します。ウィンドウを閉じても、この時点までに行

われた変更や操作は破棄されないため、[キャンセル] は使用しません。[完了](Done) は命令形の動詞ではないため、使用しません。

- ・ ウィザード名には "ウィザード" は使用しません。たとえば、"ネットワーク セットアップ ウィザード" ではなく "ネットワークへの接続" を使用します。ただし、ウィザードに言及する際にはウィザードと表現することはできます。たとえば、"最初にネットワークをセットアップする際に、ネットワークへの接続 ウィザードを使用することができます" という表現を使うことができます。
- ・ ナビゲーション中にユーザーの選択を保持します。たとえば、ユーザーが変更を行い、[戻る] をクリックしてから [次へ] をクリックした場合、これらの変更を保持するようにします。明示的に変更を取り消していない限り、ユーザーは変更を再入力する必要があるとは想定していません。

## ウィザード ページ

- ・ 効率的な意思決定を重視します。ページ数を減らして、重要なもののだけを表示します。関連ページをまとめて、省略可能なページはメイン フローから外します。最初は、ユーザーがウィザードを最後まで [次へ] をクリックして完了できることが効果的なエクスペリエンスのように思われますが、ユーザーが既定値を変更する必要がまったくない場合、そうしたページは不要である可能性があります。
- ・ ウェルカム ページは使用しません。可能な限り、最初のページを機能的なものにします。省略可能な "作業の開始" ページは、次の場合にのみ使用します。
  - ・ ウィザードを正常に完了させるために必要な前提条件がある。
  - ・ ウィザードの最初の選択ページで、選択の判断基準となる ウィザードの目的をユーザーが理解していない可能性があるが、詳細を説明する場所がない。
  - ・ "作業の開始" ページのメイン指示テキストが、"開始する前に" である。
- ・ ユーザーの選択を必要とするページは、最も可能性の高い選択肢に最適化します。このようなページでは、指示だけではなく実際の選択肢を提示します。
  - ・ "作業の開始" ページを使用しない場合は、最初の選択ページの上部で ウィザードの目的を説明します。
- ・ コミット ページは、ユーザーがタスクを確定するタイミングを明らかにするために使用します。通常、コミット ページは最後の選択ページとなり、確定するタスクを示すものに [次へ] ボタンのラベルを変更します。
  - ・ タスクがリスクを伴うものである場合(セキュリティ 関連である、時間やコストがかかるなど)や、ユーザーが自分の選択について理解していない可能性があり、確認する必要がある場合を除き、ユーザーのこれまでの選択を単に要約するような "概要" ページは使用しません。
- ・ ウィザードを終了する以外に何もできない "設定完了" ページは使用しません。 ウィザードの結果がユーザーにとって明らかである場合、最後のコミット ボタンで ウィザードを終了するようにします。
  - ・ フォローアップ ページは、ユーザーがフォローアップとして実行する可能性のある関連タスクがある場合に使用します。"電子メール メッセージを送信する" などの一般的なフォローアップ タスクは行いません。
  - ・ "完了" ページは、結果が目に見えず、タスクが完了したというフィードバックを提供する方法が他にない場合にのみ使用します。
  - ・ 進行状況 ページが含まれている ウィザードでは、タスクが完了したことを示すために、"完了" ページか フォローアップ ページを使用する必要があります。時間がかかるタスクの場合、コミット ページで ウィザードを終了し、通知を使用してフィードバックを返します。

## エラー メッセージ

- ・ ユーザーがメッセージの結果に応じて、何らかの対処をするか、自分の操作を見直す可能性がない場合は、エラー メッセージを表示しません。ユーザーが実行できる対処法がなかったり、問題がそれほど重要でない場合は、エラー メッセージを表示しないようにします。
- ・ ユーザーが問題を解決できるように、可能な限り解決策を提案します。ただし、解決の見込みが十分にある解決策を示すようにしてください。解決につながる見込みの低い、憶測だけの情報でユーザーに無駄な時間をとらせないようにします。
- ・ 具体的にします。"構文エラー" や "無効な操作" など、あいまいな表現を避けます。関連するオブジェクトの名称、場所、値などを具体的に提示します。
- ・ ユーザーに責任を押し付けたり、ユーザーに非があるような言い回しを避けます。"あなた" という表現は使わないようにします。一般には能動態が好まれますが、ユーザーを主語にすることで エラーの責任がユー

ユーザーにあるかのように感じさせてしまう可能性がある場合は受動態を使用します。

- エラー メッセージに [OK] は使用しません。ユーザーはエラーを OK であるとは見なしません。エラー メッセージに直接的な操作が示されていない場合、代わりに [閉じる] を使用します。
- 次の言葉は使用しないでください。
  - エラー、障害 (代わりに "問題" を使用)
  - 失敗しました (代わりに "できません" を使用)
  - 不正な、無効な、悪い (代わりに "正しくありません" を使用)
  - 致命的な (代わりに "プログラムの終了" を使用)
  - 中止、キル、強制終了 (代わりに "停止" を使用)
  - 壊滅的な (代わりに "重大な" を使用)

以上に挙げた言葉は不要であり、推奨される Windows のトーンに反するものです。代わりに、エラー アイコンを [正しく使用](#)すれば、問題の発生を効果的に伝えることができます。

- エラー メッセージに効果音は使用しないでください。耳障りになることが多く、必要性もありません。

## 警告メッセージ

- 警告は、今後問題を引き起こす可能性のある状態を示します。警告はエラーや質問ではないため、日常的な操作に対する質問を警告として使用しないでください。
- ユーザーがメッセージの結果に応じて、何らかの対処をするか、自分の操作を見直す可能性がない場合は、警告メッセージを表示しません。ユーザーが実行できる対処法がなかったり、条件がそれほど重要でない場合は、エラー メッセージを表示しないようにします。

## 確認

- 不要な確認は使用しません。確認は、次の場合にのみ使用します。
  - 繼続しない明確な理由があり、ユーザーが継続しない可能性が十分にある。
  - 操作が重大な結果をもたらすか、簡単に元に戻すことができない。
  - 操作によって生じる結果を、ユーザーが認識していない可能性がある。
  - 操作の続行にはユーザーの選択が必要で、選択肢の中に既定とするのに適切なものがない。
  - 現在のコンテキストで、ユーザーがエラーになる操作を実行する可能性がある。
- 確認は、"はい" か "いいえ" で答える質問にし、"はい" か "いいえ" で答えられるようにします。他の種類のダイアログ ボックスとは異なり、確認は、ユーザーがすばやく決定を下せないようにデザインされています。ユーザーが応答内容について考えなければ、確認の意味がありません。

## アイコン

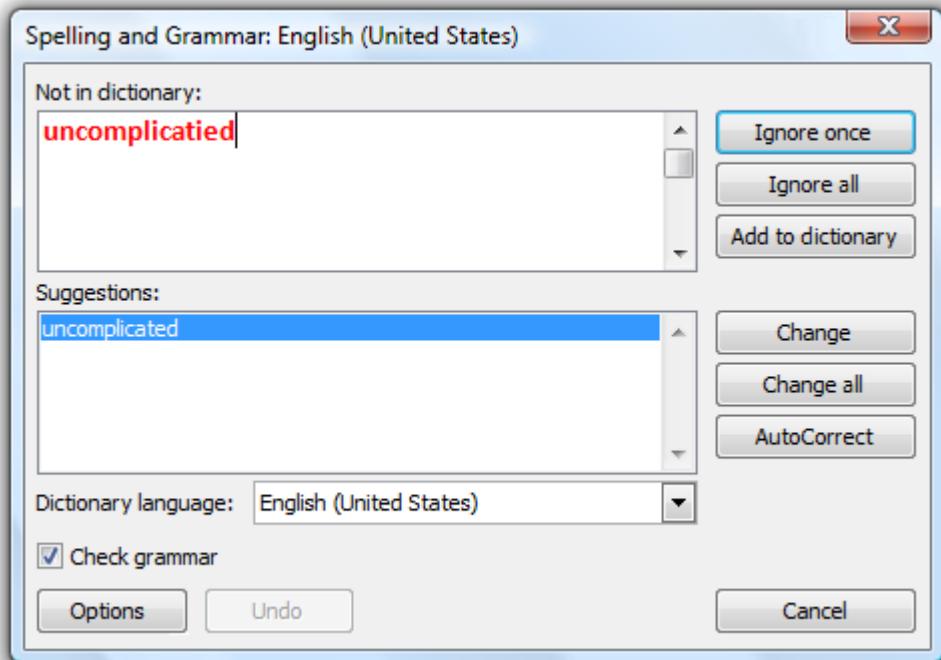
- すべてのアイコンが [Aero style アイコンのガイドライン](#)に準拠するようにします。Windows XP スタイルのアイコンはすべて置き換えます。
- 標準アイコンを選択する際には、その問題の重要度ではなく、メッセージの種類を基準にします。
  - エラー。発生したエラーまたは問題を示します。
  - 警告。今後問題を引き起こす可能性のある状態を示します。
  - 情報。有益な情報を示します。
- 問題が複数のメッセージの種類に該当する場合は、ユーザーが対処する必要のある最も重要な側面に焦点を当てます。
- アイコンは、メイン指示テキストまたはその他の対応するテキストと常に一致する必要があります。
- 重大ではないユーザー入力の問題にエラー アイコンは必要ありません。ただし、インプレース エラーにはアイコンが必要です。このようなコンテキストでのフィードバックは、アイコンがなければ容易に見過されてしまいます。
- 重大ではないエラーを "やわらかく伝える" 目的では、警告アイコンは使用しません。エラーは警告ではありません。エラー アイコンのガイドラインに従ってください。
- 質問のダイアログ ボックスでは、重大な結果を伴う質問に対してのみ警告アイコンを使用します。日常的な操作に対する質問には警告アイコンを使用しません。

## ヘルプ

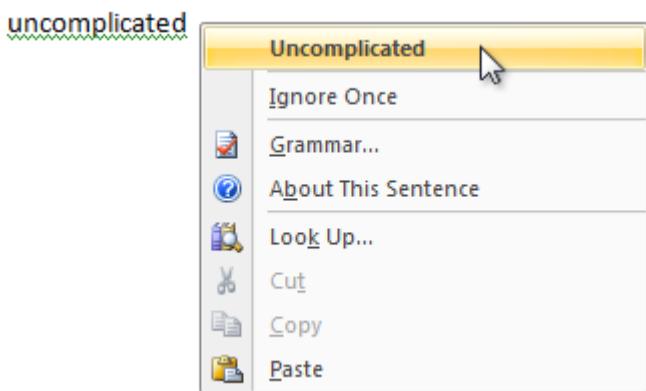
- 具体的で意味のあるヘルプトピックにリンクします。ヘルプのホームページ、目次、検索結果の一覧、または他のページにリンクしているだけのページにはリンクしません。よく寄せられる質問の膨大な一覧で構成されたページにはリンクさせないでください。こうすると、ユーザーはクリックしたリンクに合致する項目を探さなければならなくなります。具体的であっても、作業中のタスクに関連がなく、役に立たないヘルプトピックにはリンクしません。空のページにはリンクしないでください。
- 一貫性を持たせる目的で、すべてのウィンドウやページにヘルプリンクを配置することは避けます。1か所にヘルプリンクを配置したからといって、すべての場所に配置しなければならないわけではありません。
- 可能であれば常に、ヘルプリンクテキストは、メインの質問に対してヘルプコンテンツが回答を提供するような表現にします。"詳細情報の表示 (Learn more about)" や "ヘルプの表示 (Get help with this)" という表現は使用しません。
- キーワードだけでなく、ヘルプリンク全体をリンクテキストに使用します。
- 完全な文を使用します。
- 疑問符を除いて、末尾に句読点や省略記号は付けません。
- ヘルプコンテンツがオンライン上にある場合は、そのことをリンクテキストに明記します。そうすることによって、リンクの結果が予測しやすくなります。

## パワフルかつシンプルに

パワフル:



パワフルかつシンプル:



理想的な Windows® ベース アプリケーションは、パワフルでシンプルなものです。アプリケーションをパワフルにしたい、シンプルにしたいと考えるのは当然ですが、両方を実現することは可能でしょうか。この 2 つの目標はそもそも相反するものですが、必ずしも両立できないものではありません。機能の選択と提示のバランスを慎重にとることにより、パワフルでシンプルなアプリケーションを実現できます。

パワフルであること

ソフトウェアにおいて "パワー" とは何を意味するのでしょうか。あらゆるユーザーにとって万能であることを目指し、さまざまな機能を満載したアプリケーションがパワフルと見なされる場合もあるかもしれません。しかし、不必要的機能がユーザーのニーズを満たすことはないため、このようなデザインが成功しているとはいえません。これは、私たちが目指しているパワーではありません。

パワフルなアプリケーションとは、次の特徴を最適に組み合わせたアプリケーションです。

- ・ 有効性。そのアプリケーションがなければできないタスクを実行でき、目的を効果的に達成できることで、対象とするユーザーのニーズを満たす。
- ・ 効率性。そのアプリケーションなしには不可能なレベルの生産性と規模でタスクを実行できる。
- ・ 多目的性。さまざまな環境で幅広いタスクを効果的に実行できる。
- ・ 直接性。中断や不必要的手順を要求されることはなく、直接的に目的の達成を支援するものであると感じら

れる。ショートカット、キーボードアクセス、マクロなどの機能は、直接性の感覚を向上させます。

- ・柔軟性。作業を完全にきめ細かく制御できる。
- ・統合性。Microsoft® Windows® と十分に統合され、他のアプリケーションとデータを共有できる。
- ・先進性。競合他社のソリューションにはない、すばらしい革新的な最新機能を備えている。

こうした特徴のいくつかは、ユーザーの認識によって変わり、ユーザーのその時の能力に関連しています。何をパワフルと考えるかは時間と共に変化するため、今日の高度な検索機能が、明日には平凡でありふれた機能になっているということもあり得ます。

以上の特徴をすべて集約すると、パワフルなアプリケーションを次のように定義できます。

対象ユーザーがそのすべての可能性を効率的に実現できる場合、そのアプリケーションはパワフルである。

したがって、パワフルであることの最終的な基準は、機能の数ではなく生産性です。

ユーザーがそのすべての可能性を実現するために必要とする支援の方法は、ユーザーごとに異なります。あるユーザーにとっては有効でも、別のユーザーにとっては、多目的性、直接性、および制御性を損なうものである可能性もあります。適切にデザインされたソフトウェアは、これらの特徴のバランスが適切にとれています。たとえば、非専門家向けにデザインされたデスクトップパブリッシングシステムでは、ウィザードを使用して、ユーザーが複雑なタスクを処理できるようにしている場合があります。このようなウィザードでは、対象ユーザーは、それがなければ実行できないようなタスクを実行することができます。それに対して、プロフェッショナル用のデスクトップパブリッシングシステムでは、直接性、効率性、および完全制御性を重視します。このようなアプリケーションのユーザーにとっては、ウィザードは制限が多くてストレスを感じるかもしれません。

### 最も重要な点

対象ユーザーの目的を理解し、その目的を生産的に達成できる機能を作成します。

### シンプルであること

シンプルさは、次のように定義されます。

シンプルさとは、対象ユーザーが重要でないと感じたり判断したデザインの属性を減らしたり取り除くことである。

実際には、シンプルさは、適切な機能を選択したり、その機能を適切に表示することで実現できます。これにより、実際的にも感覚的にも重要でない要素を減らすことができます。

シンプルであるかどうかは、ユーザーの受け止め方にもります。次に、自動变速装置の効果をどのように感じるか、ユーザーの視点による違いを検討してみます。

- ・一般的なドライバー(対象ユーザー)の場合、自動变速装置があるとギアシフトやクラッチを手動で操作する必要がなくなるため、自動車の運転がとても楽になります。手動によるギアシフトやクラッチの操作は必ずしも運転に必要なタスクではないため、それらを削除することによりシンプルさを実現できます。
- ・プロのカーレーサーの場合、变速装置を直接制御する機能は勝利のためには欠かせません。自動变速装置は自動車のパフォーマンスにマイナスの影響を与えるため、自動变速装置によってシンプルさを実現できるとは見なされません。
- ・整備士の場合、自動变速装置は手動变速装置に比べてメカニズムが複雑であるため、修理や維持が難しくなります。整備士とは異なり、対象ユーザーは幸いなことにこうした内部の複雑さを意識することはありません。

自動变速装置をどう考えるかは、ユーザーによってさまざまですが、対象ユーザーにとっては、必ずしも重要ではない知識、スキル、および労力の必要性がなくなるという点で、成功といえます。一般ドライバーにとっては、自動变速装置は優れた機能です。

### シンプルさと使いやすさの違い

シンプルさは、正しく適用すれば、使いやすさにつながります。しかし、シンプルさと使いやすさは同じ概念ではありません。使いやすさを実現するとは、ユーザーが、適切な時間内に、困難や混乱を伴わ

ずに、自分自身で正常にタスクを実行できるようにするということです。使いやすさを実現する方法は数多くありますが、シンプルさ(重要でない要素の削除)は、そのうちの1つです。

すべてのユーザーは、たとえどんな上級ユーザーでも、不要な労力を最小限に抑えて作業を完了させたいと考えています。彼らの主な動機は、作業を完了させることであり、コンピューターやアプリケーションについて学ぶことではありません。

シンプルさは、使いやすさを実現する最も効果的な方法であり、使いやすいということは使われるということと同じです。複雑で使いにくい機能は使われなくなります。それに対して、シンプルで洗練されたデザインで、適切に機能を実行できれば、喜んで使われます。そのようなデザインに触ると、積極的に使いたいという気持ちになるものです。

たとえば、Microsoft Windows XP のワイヤレス ネットワークのサポートについて考えてみます。マイクロソフトでは、ユーザーが構成プロセスを処理できるウィザードを追加することもできましたが、このアプローチでは、使いやすさは実現できますが、シンプルではありません。重要でない機能(ウィザード)が追加されているからです。その代わり、ワイヤレス ネットワークが自動的に構成されるようにデザインしました。最終的にユーザーにとっては、信頼できて安全に機能するのであれば、構成の詳細は気にしません。ワイヤレス ネットワーク テクノロジーでは、このパワフルでシンプルなデザインの実現が、その人気と急速な普及につながっています。

### 最も重要な点

作業を適切に実行できる最もシンプルなデザインからデザイン プロセスを開始します。

現在のデザインに満足していない場合、重要でない要素をすべて取り除くことから始めます。通常、残った要素は優れたものだとわかるはずです。

## パワーを維持しながらシンプルさを実現する

### デザイン原則

シンプルさを実現するには、常に、可能性ではなく、確実性のためにデザインします。

### 可能性

可能であることに基づいたデザインの決定は、レジストリ エディターのような複雑なインターフェイスにつながります。このデザインでは、すべての操作が同じように可能になり、その結果、同じだけの労力を必要とします。すべての操作が可能なため、ユーザーの目的はデザインの決定では考慮されません。

### 確実性

高い確実性に基づいたデザインの決定は、簡潔な、目的ベースおよびタスクベースのソリューションにつながります。いかにもありそうなシナリオが注目され、最小限の労力で実行できます。

### シンプルなデザインの原則

シンプルさを実現するには、いかにもありそうなことに注目し、可能性の低いことは削減、非表示、または削除し、不可能なことは完全に排除する。

ユーザーがしたいことは、ユーザーがするかもしれないことよりも、デザインにとってはるかに重要です。

### デザインの手法

パワーを維持しながらシンプルさを実現するには、適切な機能セットを選択し、適切な場所に配置して、これらの機能セットを使用するための労力を軽減するようにします。このセクションでは、こうした目的を実現するための一般的な手法について、いくつか説明します。

### 適切な機能セットを選択する

"完璧であるということは、これ以上足すものがなくなったときではなく、これ以上取り除くものがなくなったときに実現する。"- アントワーヌ・ド・サン=テグジュペリ

実際に削減や削除を行ってシンプルさを実現しながら、必要な機能をユーザーに提供するデザイン手法を、次に挙げます。

- ユーザーが必要な機能を判断する。目標、シナリオ、およびタスク分析によってユーザーのニーズを理解し、これらの目的を実現する機能セットを判断します。
- 重要でない要素を削除する。使用されそうもない要素や、より望ましい代替手段のある要素を削除します。
- 重要でない冗長性を取り除く。タスクを実行するのに、複数の効果的な方法がある場合があります。シンプルさを実現するには、あらゆる方法を提供して選択できるようにするのではなく、厳しい決断をし、対象ユーザーにとって最も良い方法を選択します。
- 自動的に機能させる。ある要素が必要でも、許容できる既定の動作や構成があれば、それを機能させるためのユーザー操作は必要ありません。シンプルさを実現するには、要素が自動的に機能するようにし、ユーザーに対して完全に非表示にするか、あまり目に触れないようにします。

### 提示を合理化する

"簡素化する能力とは、必要なものを活かすために、不要なものを取り除くことである。"- ハンス・ホフマン

削減や削除の認識を通じてシンプルさを実現しながらパワーを維持するためのデザイン手法を、次に挙げます。

- まとめるべきものはまとめる。タスクを 1 か所で実行できるように、タスクをサポートする重要な機能をまとめて配置します。タスクの手順は統合し、合理化したフローにします。複雑なタスクは簡単でわかりやすい手順に分解し、ウィザードのように、"1 か所" に複数の UI 画面が表示されるようにします。
- 区別すべきものは区別する。すべてを 1 か所に表示できるとは限らないため、常にわかりやすい適切な境界を設定します。中心となるシナリオをサポートする機能を中心にはっきりと表示し、オプションの機能は非表示にするか、周辺に表示します。個々のタスクは区別して、関連タスクにはリンクを設定します。たとえば、写真の操作に関するタスクは、写真のコレクションの管理に関するタスクとははっきりと区別しますが、どちらからもすぐにアクセスできるようにします。
- 取り除けるものは取り除く。デザインを印刷し、最も重要なタスクを実行するために使用する要素を強調表示します。さらに、有益な情報を伝える UI テキスト内の個々の単語を強調表示します。これで、強調表示されていない要素を確認し、デザインから削除することを検討します。アイテムを削除しても問題ない場合は、削除します。  
一貫性、構成可能性、および一般化を図ることは、質の向上にとって望ましいことが多いのですが、それによって無意味な複雑さが生じることがあります。一貫性、一般化、および構成可能性に関して、冗長なテキストがある、2つで十分なタイムゾーンをいくつも設定する、ユーザーが変更する可能性のないオプションがあるなど、見当違いの設定がないかデザインを確認して、取り除けるものは取り除きます。
- 適切な場所に要素を配置する。ウインドウ内の要素の位置は、その用途に従って配置します。重要なコントロール、指示、および説明は、すべてコンテキストに沿って論理的な順序で配置します。他のオプションが必要な場合、シェプロンをクリックするか、それに似たような方法でコンテキスト内に表示します。他の情報が必要な場合は、マウス ポインターを重ねて情報ヒントを表示します。あまり重要でないタスク、オプション、およびヘルプ情報は、メインのフロー以外の場所(別のウインドウやページ)に表示します。必要に応じて追加の詳細を表示する手法を "段階的表示" といいます。
- 意味のある高度な組み合わせを使用する。個々の要素ではなく、関連する要素のグループを選択して操作する方が、簡単でスケーラブルであることはよくあります。高度な組み合わせの例には、フォルダー、テーマ、スタイル、ユーザー グループなどがあります。このような組み合わせは、多くの場合、個々の要素からはわからないような、ユーザーの目的や意図に対応します。たとえば、[ハイコントラスト 黒] の配色に込められた意図は、黒のウインドウ背景色に込められた意図に比べ、はるかに明白です。
- 適切なコントロールを選択する。デザイン要素は、使用されるコントロールによって具体化されるため、適切なコントロールを選択することは、効率的な提示にきわめて重要です。たとえば、Microsoft Word で使用されるフォントの選択ボックスには、フォントのプレビューと最近使用したフォントの両方が表示されます。同様に、Word でスペル ミスや文法上の間違いの可能性がある場所を表示する方法は、この記事の冒頭で示したように、ダイアログ ボックスで表示する方法に比べてはるかにシンプルです。

### 労力を軽減する

"簡単なものは簡単であるべきである。

複雑なものは実現できるべきである。" - アラン・ケイ

ユーザーの労力を軽減するデザイン手法を、次に挙げます。

- タスクを見つけやすく、見やすくする。すべてのタスク、特に頻繁に行うタスクは、ユーザーインターフェイス内ですぐに見つけられるようにします。タスクの実行に必要な手順は表示するようにし、記憶に頼らないで済むようにします。
- ユーザーが扱う分野の表現でタスクを提示する。複雑なソフトウェアでは、ユーザーが問題とテクノロジーをマッピングする必要があります。シンプルなソフトウェアでは、このマッピングは自然に提示されます。たとえば、赤目軽減機能は問題領域に直接マッピングされ、色合いやグラデーションなどの詳細についてユーザーが考える必要はありません。
- 専門知識をプログラムで利用できるようにする。アプリケーションを正常に使用するために、ユーザーが外部の情報にアクセスしなくて済むようにします。専門知識には、複雑なデータやアルゴリズムもあれば、単に有効な入力の種類を説明する情報もあります。
- ユーザーが理解できるテキストを使用する。よく練られたテキストは、ユーザーとの有効なコミュニケーションには重要です。ユーザーがよく知っている概念や用語を使用し、ユーザーが知識や情報に基づいた決定を行えるように、ユーザーの疑問をわかりやすい言葉で完全に説明します。
- 安全で、セキュリティで保護された、適用される可能性の高い既定値を使用する。ほとんどの環境で通常ユーザーに適用される値が設定に含まれており、その設定が安全でセキュリティで保護されている場合、その値を既定値として使用します。必要な場合のみ、ユーザーが値を指定するようにします。
- 制約する。タスクを実行する方法が数多くあり、その一部だけが正しい場合、タスクの実行は正しい方法のみに制限します。すぐに防げるミスをユーザーがしないようにします。

### シンプルさと単純さの違い

"何事もできる限り単純化しなければならないが、必要以上に単純化してはならない。" - アルバート・aignシュタイン

シンプルさは、効果的で望ましいユーザー エクスペリエンスにとって不可欠であると考えられます  
が、シンプルさが過剰になる可能性は常にあります。シンプルさで最も重要なのは、重要でないものを削減または削除することです。重要ななものまで削除すると、デザインが貧弱になってしまいます。"シンプルにすること" によって、ユーザーの不満や混乱を招いたり、自信がなくなったり、タスクをうまく完了できなくなったりした場合、それは削除しすぎているということです。

### シンプルさには開発者の努力が必要

"私がこんなに長い手紙を書いたのは、短くする時間がないからだ。" - ブライス・パスカル

パワーを維持しながらシンプルさを実現するには、内部をかなり複雑にしなければならないことがよくあります。通常、テクノロジーの設計をすべて公開するソフトウェアをデザインすることは、それを秘密にしているソフトウェアをデザインすることよりも簡単です。後者では、対象ユーザーとその目的を十分に理解している必要があります。実際には実用的でない斬新な機能を追加するかどうかを決定するのに規律が必要なのと同じように、機能を削除するのにも規律が必要です。シンプルさには、すべてを構成可能にする代わりに、厳しいデザインの選択が必要です。多くの場合、複雑なソフトウェアは、使ったことがない機能や理解できないほど複雑な機能に価値があるとするユーザーの誤解から生じています。

### パワフルかつシンプルに

パワーとは、ユーザーに可能性を与え、生産性を向上させるものにほかなりません。シンプルさとは、重要でないものを削除し、機能を正しい方法で提示することです。対象ユーザーを理解し、機能と提示の適切なバランスを図れば、パワフルでシンプルな Windows ベースのアプリケーションがデザインできます。

# Windows Presentation Foundation を使ってデザイン

UX ガイドの準拠

Windows Presentation Foundation の適切な使用

ガイドライン

テーマ

ソフトウェアのブランド化

カスタムコントロール

3D

アニメーション

動的動作

直接操作

ブラウザーでのホスティング

Windows Presentation Foundation (WPF) は、より高度な視覚効果を実現できるユーザーインターフェイス開発環境です。ドキュメント、メディア、2次元および3次元のグラフィック、アニメーション、Web の持つ機能などを組み込んだインターフェイスを作成できます。概要については、[Windows Presentation Foundation の概要に関するページ](#)を参照してください。

Windows® の WPF は、適切に使用すれば、対象ユーザーが望んでいるような魅力的で生産的なエクスペリエンスを実現できますが、誤って使用すると、作成したプログラムが期待に反した使いにくいものになってしまう可能性があります。ガイドラインに準拠することにより、その違いを理解し、WPF のテクノロジーを適切に使用することができます。

## UX ガイドへの準拠

Windows ユーザー エクスペリエンスのガイドライン(略して "UX ガイド")は Windows 専用に作成されましたが、このガイドラインのほぼすべての内容は、WPF を使用するプログラムにも適用されます。ただし、これはそれほど驚くことではありません。このガイドラインの主な目的は、実装方法を問わず、Windows ベースの "すべての" アプリケーションについて、質の高い、一貫性のある基準を確立することであるからです。

Windows Presentation Foundation では、従来の Microsoft® Windows® のものから高度にカスタマイズされたものまで、幅広いユーザー エクスペリエンスを作成できる柔軟性があり、WPF を使用することによって Windows の外観をあきらめなければならないということはありません。実際、WPF には従来の Windows の外観が既定で用意されており、Aero や Windows XP の高度なテーマも含まれています。WPF ベースのプログラムでは、プログラムの外観にかかわらず、豊富なアプリケーションモデル、動的なレイアウト、データ バインドなどの強力な機能を利用できます。

## Windows Presentation Foundation の適切な使用

特に WPF では、プログラムの外観を完全に変更して企業ブランドに合うようにしたり、カスタム操作モデルを使用してプログラムに "独自の" 操作を追加することもできます。ドロップダウンリストをカスタマイズして、スライダーのようにすることもできます。ただし、このような従来のエクスペリエンスを変更することが本当に適切かどうかは別の問題です。

"斬新" なプログラムとは

WPF には、すばらしい高度な機能が用意されています。使い慣れてくると、より優れた "斬新" なソフトウェアを作成したいという欲求が出てきます。こうした試みは、多くの場合、うまくいかないようです。その理由を知るために、プログラムが斬新といえる場合とそういえない場合の違いを見てみましょう。

プログラムが真に "斬新" といえるのは、次の場合です。

- 機能がプログラムとその対象ユーザーに適している。
- 外観が(多くの場合、細かな点において)美的満足をもたらしている。
- パフォーマンスを損なわずに、ユーザビリティやフローが向上している。
- 良い印象がずっと続く(何回見ても、初めて見たときのように楽しめる)。

プログラムが "斬新" とはいえないのは、次の場合です。

- テクノロジーを使用できるというだけで使用/乱用している。

- ・機能のユーザビリティ、フロー、またはパフォーマンスが低下している。
- ・ユーザーにとって不要な注意を頻繁に喚起する。
- ・良い印象が続かない。最初は魅力的でも、すぐに薄れる。

画家のパレットの色に本来良いも悪いもないと同じように、WPF の機能にも、本来良いも悪いもありません。しかし、具体的なプログラムやその対象ユーザーについていえば、適切なデザインと不適切なデザインがあると考えられます。真に斬新なプログラムは、使用できるからという理由ではなく、必要だからという理由で、テクノロジーを使用します。

斬新さを実現するには、テクノロジーによってできることを考えるのではなく、対象ユーザーが本当に必要としていることを中心に考えることから始めます。“斬新な”機能を追加する前に、それを支える明確な [ユーザー シナリオ](#) があることを確認してください。

## 対象ユーザー

優れたソフトウェアをデザインするとは、要するに、どういった機能をユーザーに提供し、その機能をどのように提示するかという一連の決定を下していくことです。優れたソフトウェアでは、こうした決定を、ソフトウェアとその対象となるユーザーの目的に基づいて行う必要があります。適切なデザイン上の決定は、自分のためでも、プラットフォームで簡単に作成できるからという理由からではなく、対象ユーザーのメリットために行われる必要があります。

対象ユーザーを理解するには、対象ユーザーの知識、目的、好みを一覧にしてみます。プログラムを使用する動機、使用頻度、作業環境について理解する必要があります。また、実際のユーザーのクラスを表すために、実際のデータからデザインされた架空のユーザーである、[ペルソナ](#)を作成してみるのも効果的です。

## プログラムの種類

作成するプログラムの種類もまた、決定の要因になります。一般的なプログラムの種類の特徴を、次に挙げます。

## 生産性アプリケーション

- ・対象ユーザー: ナレッジ ワーカー
- ・目的: 生産性の向上、コスト削減
- ・使用頻度: 通常は長時間、場合によっては終日
- ・ユーザーの期待: 使い慣れている、一貫性がある、すぐに生産性を向上できる
- ・WPF の適切な使用方法: ユーザーの生産的な作業遂行を支援
- ・例: Microsoft Office、基幹業務アプリケーション

## 消費者向けアプリケーション

- ・対象ユーザー: 消費者
- ・目的: 具体的なタスクの実行(ユーザーの場合)、売上の向上(ISV の場合)
- ・使用頻度: 短期間でときどき(1週間に1回など)
- ・ユーザーの期待: 対象となるタスクを適切に実行する快適なエクスペリエンス
- ・WPF の適切な使用方法: ユーザーのタスク実行を支援、ブランド化
- ・例: マルチメディア参照アプリケーション、メディアプレーヤーおよびメディアツール、セキュリティツール

## ゲーム

- ・対象ユーザー: 消費者
- ・目的: エンターテインメント
- ・使用頻度: 中期間でときどき
- ・ユーザーの期待: 浸入型エクスペリエンス

- WPF の適切な使用方法: 外観のカスタマイズ、(可能な場合) 操作のカスタマイズ
- 例: 簡単なゲーム、オンライン ゲーム

## キオスク

- 対象ユーザー: 1 回限りのユーザー
- 目的: 具体的なタスクの実行、情報入手
- 使用頻度: 1 回
- ユーザーの期待: 学習なしで、タスクをすぐに実行できること
- WPF の適切な使用方法: 外観のカスタマイズ、(可能な場合) 操作のカスタマイズ (タッチ、ドラッグ アンド ドロップ)、視覚的な説明のアニメーション化
- 例: 空港のキオスク、美術館のキオスク

## IT プロ ユーティリティ

- 対象ユーザー: IT プロフェッショナル (開発、使用)
- 目的: タスクの実行または迅速な情報入手
- 使用頻度: 必要に応じて、通常は短期間
- ユーザーの期待: ジョブの完了
- WPF の適切な使用方法: IT プロの迅速な UI 開発およびテストを支援

## ユーザビリティの向上

デザインの概念が優れているかどうかは、単純に、ユーザビリティを向上させるものかどうかによります。WPF を使用してユーザビリティを向上させる一般的な方法を、次に挙げます。

- 現実世界をモデルにします。カスタムの外観や操作を使用して、特定のコントロールに現実世界と同じような外観や動作を設定することができます。この手法は、ユーザーが現実世界のオブジェクトに慣れ親しんでおり、現実世界のアプローチがタスクを実行するのに最適で、最も効率的な方法である場合に、最も効果的です。たとえば、電卓のような簡単なユーティリティは、現実世界の電卓をモデルにすることで、より効果を発揮します。
- 説明する代わりに表示します。アニメーションや切り替え効果を使用して、関係、原因、効果を示すことができます。この手法は、アニメーションや切り替え効果がない場合に説明のテキストが必要となる情報や、ユーザーが見落とす可能性のある情報を表示する場合に、最も効果的です。たとえば、幼児向けの本では、ページめくりをアニメーション化してコントロールの機能を示すことができます。通常のページめくりは、幼児にとっては理解しにくい可能性があります。
- アフォーダンスを強化します。アフォーダンスは、ラベルを使用してオブジェクトを説明するのに対して、オブジェクトの使用方法を示唆するオブジェクトのプロパティです。カスタムのコントロールの外観やアニメーションを使用して、標準以外のコントロールの使用方法を示唆することができます。
- 自然なマッピングを使用します。自然なマッピングとは、ユーザーがやりたいこととその方法との明白な関係のことです。カスタムの外観や操作を使用して、標準のコモン コントロールではできない自然なマッピングを作成できます。
- 情報を減らします。カスタムの操作を使用して、操作の実行方法の数やタスクの実行に必要な情報量を制限できます。
- フィードバックを強化します。カスタムのコントロールの表示やアニメーションを使用して、フィードバックを行い、作業が正しく行われているかどうかを示したり、進捗状況を表示することができます。たとえば、Windows Internet Explorer® のアドレスバーでは、ページの読み込みの進捗状況を背景に表示します。
- オブジェクトを操作しやすくします。フィットの法則では、対象のクリックに必要な労力は、その距離に比例し、サイズに反比例するとされています。たとえば、アニメーションを使用して、ポインターが近づいたときにオブジェクトが大きくなり、離れたときに小さくなるようにすることができます。そうすることで、オブジェクトをクリックしやすくなります。また、通常はオブジェクトを小さくしておくことで、画面領域をより効率的に使用することもできます。
- 強調表示します。豊富なレイアウトとカスタムの外観を使用して、タスクに重要な画面要素を強調表示し、2 次的な要素は強調しないようにすることができます。

## 決定を下す

ここで、これまでの決定要因をすべてまとめてみましょう。切り替え効果の概念をプログラムで採用すべきかどうかを決定する場合、それが可能かどうか、独創的かどうか、簡単かどうかということからスタートしないようにしてください。こうした決定要因は対象ユーザーにとってはどうでもよいことです。

そうではなく、機能自体のユーザビリティについて検討してください。たとえばそれによってオブジェクトの切り替え前または後が明確になるなど、その機能に切り替え効果のメリットがあるかどうか、切り替えの速度はどれくらいなのか、プログラム全体のパフォーマンスに影響しないかどうかなどについて検討します。

対象ユーザーについても検討してください。たまにしかタスクを実行しないかどうか、時間の制約により切り替え効果がわざらわしいものにならないかについて検証し、最後に、この切り替え効果がプログラムの種類に適しているかどうかを検討します。ユーザビリティに役立つ簡単な切り替え効果は、ほとんどすべてのプログラムに適していますが、ユーザビリティに役立たない複雑なアニメーションが適しているのは、ゲームなどの娯楽向けのプログラムくらいです。

適切な質問を自分自身に問いかけて、よく考えて答を出すことにより、WPF の機能を適切に使用して、すばらしいユーザー エクスペリエンスを作成できます。

## 結論

マイクロソフトには、ソフトウェアは役に立つもの、使用に適しているもの、望ましいもの、ふさわしいものであるべきという 4 つのデザイン基本原則があります。これらの基本原則について、内容も順序も適切だと考えています。Windows Presentation Foundation を適切に使用すれば、これらの 4 つの基本原則をすべて満たすことができます。

## ガイドライン

### テーマ

一般に、[アプリケーション テーマ](#)は、生産性よりも全体のエクスペリエンスが重要なプログラムに適しています。高度なテーマを持つアプリケーションは、没入型で、短期間のみ使用するものに限ります。テーマは、ゲームやキオスク アプリケーションには適しているが、生産性アプリケーションには適していません。

ただし、テーマの作成は、行うか行わないかの 2 択ではありません。カスタマイズされたコントロールの外観を特定のコントロールに使用して、カスタムの外観を設定できます。ブランド化された消費者向けアプリケーションでは、シンプルで控え目なアプリケーション テーマを使用して、ブランド感覚を創出できます。

### 必要事項

- 生産性アプリケーションには標準の Windows テーマを使用します。
- 消費者向けアプリケーションには、控え目な、ブランドに関連したアプリケーション テーマの使用を検討します。
- ゲームおよびキオスク アプリケーションには、没入型エクスペリエンスを実現するために、重厚なアプリケーション テーマを使用することもできます。
- テーマは、選択的に、控え目に、節度を持って作成します。テーマの使用を決定することは、あらゆるものについてテーマを作成したり、まったく違う外観にしたりしなければならないということではありません。
- プログラムが現実世界のオブジェクトをモデルにしている場合、カスタムの外観によってコントロールの外観と動作を現実世界のオブジェクトと同じように作成することを検討しますが、これは、この現実世界のアプローチがタスクを実行するのに最適で、最も効率的な方法である場合に限ります(これはかなり厳しい基準です)。
- タスクにとって重要な画面要素を強調表示し、2 次的な要素は強調されないようにテーマを作成することを検討します。
- プロのグラフィック アーティストにテーマの作成を依頼します。テーマを成功させるには、色、フォーカス、コントラスト、テクスチャ、および次元の使い方について理解する必要があります。

## 禁止事項

- ・ アプリケーションが没入型エクスペリエンスで、他に実行中のプログラムがなく全画面で実行している場合を除き、ウィンドウの非クライアント領域にはテーマを作成しません。
- ・ 疑問がある場合、テーマは使用しません。

## ソフトウェアのブランド化

競争の激しい市場では、企業は自社の製品をブランド化することにより、競合他社と差を付けることができます。ただし、プログラムの外観や動作を風変わりなものにしても、ブランドの独自性を強化することにはなりません。それよりも、受け入れられやすく、同時に他より目立つ、特性のあるプログラムを作成することを目指します。

最終的には、ユーザーは、その目的を十分に果たすことのできる質の高いプログラムに対して、最も良い印象を受けます。ユーザーの気を散らせたり、ユーザビリティやパフォーマンスに悪影響を与えるブランド化では、ユーザーから好印象を得られる可能性は低くなります。

## 必要事項

- ・ ブランドの基盤となるような優れた製品名とロゴを選択します。テーマはブランドの基盤にはなりません。
- ・ [バージョン情報] ダイアログ ボックスやヘルプ ファイルに製品 Web サイトへのリンクを表示することを検討します。製品 Web サイトは、製品自体のブランド化よりもはるかに効果的にブランドを売り込み、製品をサポートすることができます。
- ・ ブランド化が必要な場合、次のような影響の低いブランド化を検討します。
  - ・ ワークフロー外に配置され、小さく控え目に表示された会社や製品のロゴ。
  - ・ 会社や製品の色を示唆する、小さく控え目なテーマ カラーの変更。

## 禁止事項

- ・ ユーザーの気を散らせたり、ユーザビリティやパフォーマンスに悪影響を与えるブランド化は使用しません。
- ・ カスタム コントロールはブランド化に使用しません。カスタム コントロールは、特殊な没入型エクスペリエンスを作り出すために必要な場合、または特別な機能を必要とする場合に使用します。
- ・ アニメーション化された[スプラッシュ スクリーン](#)はブランド化に使用しません。実際、読み込みが速いプログラムではスプラッシュ スクリーンを使用しません。
- ・ アニメーション効果付きのロゴは使用しません。
- ・ UI 画面ごとに会社や製品のロゴを表示することは避けます。
  - ・ 製品や会社のロゴの使用は、メイン ウィンドウまたはホームページと [バージョン情報] ボックスなど、最大 2 か所に限定します。
  - ・ 製品や会社のロゴの使用は、どのような画面においても 2 回までとします。
  - ・ 製品名や会社名の使用は、どのような画面においても 3 回までとします。

その他のガイドラインと例については、「[ブランド化](#)」を参照してください。

## カスタム コントロール

Windows のコモン コントロールは、使い慣れており、一貫性があって柔軟性が高く、利用しやすいものです。経験を積んだ Windows ユーザーから学ばなくても、ほとんどすべての状況で適切に操作できます。

そうはいっても、コモン コントロールが最も機能するのは、想定された使用パターンにおいてです。たとえば、スライダー コントロールが標準になる前は、代わりにスクロールバーが使用されていたことがありました。しかし、スクロールバーは、ドキュメントのスクロールを想定しており、連続範囲からの値の選択を想定したものではありません。スライダー コントロールが標準になる前は、この操作にはカスタム コントロールを使用することが良しとされていました。

## 必要事項

- ・ カスタム コントロールは、コモン コントロールでサポートされていない特殊な動作に使用します。

- カスタム コントロールでは、システムのメトリックスと色をサポートし、すべてのユーザーがこれらの設定を変更することを考慮するようにします。
- カスタム コントロールは Windows アクセシビリティ のガイドラインに準拠するようにします。

#### 禁止事項

- コモン コントロールには、標準以外の動作を割り当てません。可能であれば標準的な動作を使用し、必要な場合にのみカスタム コントロールを使用します。
- カスタム コントロールを正しく使用するために必要な作業を過小評価しないようにします。

### 3D

#### 必要事項

- 3D グラフィックスは、3 次元のオブジェクト、グラフ、グラフィックを視覚化し、ユーザーが検証および操作できるようにするために使用します。
- 3D グラフィックス機能の必要性を支える明確なユーザー シナリオがあることを確認します。

#### 禁止事項

- できるからという利用だけで 3D グラフィックスを使用しないでください。

### アニメーション

5 つの基本的なアニメーションの種類を次に挙げます。

- 説明: 言葉ではなく視覚的に情報を伝えるアニメーションです。
- 効果: 現実世界の操作をモデルにした操作を作成するアニメーションです。
- 関係: オブジェクト間の関係(どこから来てどこへ行くか)を表わすアニメーションです。
- 切り替え: 主な UI の状態の変化を表わすアニメーションです。
- フィードバック: 作業が正しく行われているかどうかを示したり、進捗状況を表示するアニメーションです。

人間の目は、(特に視界周辺の)動きに敏感です。何かに注意を引くためにアニメーションを使用する場合は、ユーザーの思考の流れを中断してまでその注意を引く意義や価値があることを確認してください。

アニメーションによってうまく注意を引き付けられなくてもかまいません。実際、多くの効果的なアニメーションは、ユーザーが気付かないくらい自然です。

#### 必要事項

#### 説明アニメーション

- 1 回だけ説明を行う説明アニメーションを使用します。混乱を招くようなら、あまり効果がないということです。
- ユーザーを圧倒しないように、一度に 1 つだけ表示します。
- 最適な速度で再生します。早すぎると理解しにくくなり、遅すぎると見るのは退屈します。
- 繰り返しのアニメーションは、徐々に速度を上げます。視聴者は既にアニメーションを見慣れているため、速度をゆっくり上げることで快適に感じます。
- 重要な部分で速度を落とすなどして、重要な点を強調するようにタイミングを調整します。

#### 効果アニメーション

- 効果アニメーションは、ユーザーが現在操作中のオブジェクトに対して使用します。このようなアニメーションでは、ユーザーが既にオブジェクトに集中しているため、気が散ることはありません。
- 状態を示す効果アニメーションの使用は最小限に抑えます。次の点に留意します。
  - ユーザーが実際に利用できる追加情報を提供することで、真の効果が発揮されるようにします。この例としては、一時的な状態の変化や緊急事態などがあります。

- 控え目なものにします。
- 実行時間を短くし、ほとんどの時間を実行していない状態にします。
- 無効にできるようにします。
- 効果アニメーションは、あまり注意を引き付けないように、目立たないようにします。まったく移動させないか、少しだけ移動させるようにし、それよりもフェードを使用したりオーバーレイで変化させたりするようになります。

#### 関係アニメーション

- 選択したオブジェクトで開始または終了させます。ユーザーが現在操作していないオブジェクト間の関係は表示しません。
- 2分の1秒以内で完了するようにします。

#### 切り替えアニメーション

- 状態間の関係を表示するために使用します。状態の変化をアニメーション化すると、理解しやすくなり、表示が滑らかになります。
- 切り替えが自然なマッピングになるようにします。たとえば、ウィンドウを開く切り替えは、上方に拡大されるようにし、ウィンドウを閉じる切り替えは、下方に縮小されるようにします。
- 2分の1秒以内で完了するようにします。

#### フィードバック アニメーション

- 完了とエラーの状態をはっきりと特定できるようにします。
- 基となるプロセスが進行していない場合、進捗状況を表示しないようにします。

#### 禁止事項

- パフォーマンスに著しく影響を与えるアニメーションは使用しません。遅いネットワーク接続を使用する場合や、多くのオブジェクトが関わる場合のパフォーマンスについて考慮します。
- 注目させる価値がないものに注意を引かないようにします。

#### 動的動作

WPFでは、[段階的表示](#)を使用して、追加情報を表示または非表示にすることができます。段階的表示を使用することで、本質的要素を中心に置きながら、簡単な操作で必要に応じて追加の詳細を表示できます。

段階的表示コントロールには、通常、動作を説明するラベルはありません。そのため、コントロールの外観のみに基づいて、ユーザーが以下を判断できるようにする必要があります。

- コントロールで段階的表示が提供されていること。
- 現在、展開されている状態か、または折りたたまれている状態か。
- 追加情報、オプション、またはコマンドがタスクの実行に必要なものか。
- 必要な場合に元の状態に戻すための方法。

上記の判断にユーザーによる試行錯誤が必要になるようであれば、試さなくても判断できるように工夫してください。

#### 必要事項

- 段階的表示の機能が常に表示されているようにします。
- 適切なグリフを使用します。非表示のコンテンツ内にあるアイテムを表示するには、スライドして開く方向を示すシェvronまたは単一の山かっこを使用します。
- 正しい方向を示すグリフを使用します。シェvronの向きはアクションが発生する方向を示します。

#### 禁止事項

- マウスポインターを合わせなくとも段階的表示であることがわかるようにします。

## 直接操作

WPF では、[直接操作](#)を使用すると、キーボードで間接的にダイアログ ボックスやメニューを操作するのではなく、マウスで直接オブジェクトを操作できるようになります。

直接操作は、多くのタスクを実行する上での自然な方法です。習得しやすく、便利です。しかし、多くの課題があります。最も重要なことは、大切なデータが誤って操作されるのを防ぐことです。

### 必要事項

- 直接操作は、次の方法で示します。
  - ポイントするとマウス ポインターを手の形に変更する。
  - オブジェクトの選択時にドラッグハンドルを表示する。
  - オブジェクトがドラッグされたら移動の状態を表示し、その後、適切なドロップ対象を表示する。また、オブジェクトが正常にドロップされた状態やドロップが取り消された状態も明確に表示する。
  - 名前の変更のためのダブルクリックで、インプレース テキスト ボックスを表示する。
  - 最も役に立つ属性をツールヒントで表示する。
- 操作ミスは、次の方法で防ぎます。
  - 重要な、または誤って操作される可能性のあるオブジェクトを既定でロックする。また、オブジェクトのコンテキストメニューでロック解除できるようにする。
  - 操作ミスを元に戻すための明確な方法を用意する。
- 別の方を常に用意しておくことで、直接操作を行いやすくします。

### 禁止事項

- 必要性のない場面で直接操作を提供するなど、直接操作を乱用することのないようにします。乱用することによって、UI の価値が薄れ、ユーザーはしぶしぶ操作するようになります。

## ブラウザーでのホスティング

WPF ベースのプログラムは、ブラウザーでホストすることができます。

### 必要事項

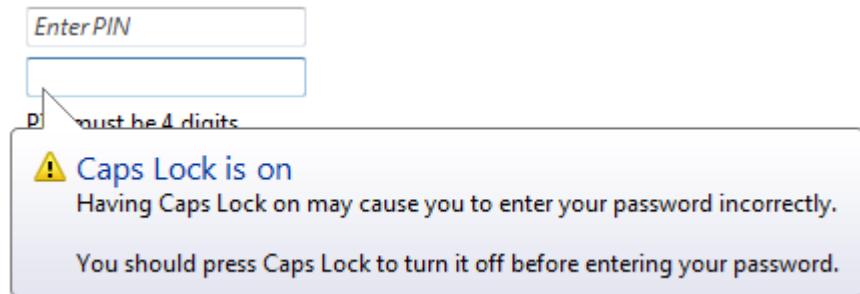
- Web ページからプログラムにナビゲートするために、プログラムをブラウザーでホストします。

### 禁止事項

- 以下のプログラムはブラウザーでホストしないでください。
  - リッチ コンテンツを編集するために使用される (ワード プロセッサなど)。
  - 複数のウィンドウを必要とする。
  - ユーザーが別のページに移動すると切断/中断されるバックグラウンド タスク (メディア プレーヤーなど)。

## コントロール

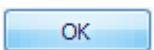
以下は、Windows® のコントロールの例です。画像をクリックすると、それぞれのコントロールのガイドライン項目に移動します。



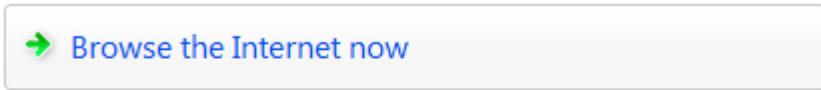
バルーンは、コントロールでの緊急性の低い問題や特殊な状態を知らせます。

- 
- Clock
  - Volume
  - Network
  - Power

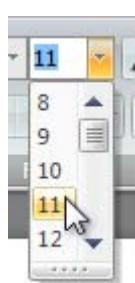
チェック ボックスを使用すると、2つまたはそれ以上の、明白に異なる複数の選択肢から意思決定できます。



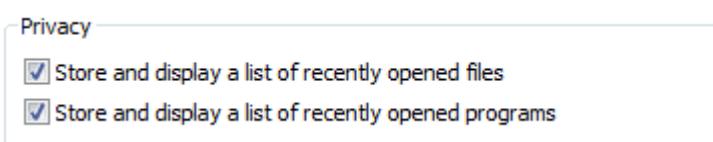
コマンド ボタンを使用すると、直接アクションを実行できます。



コマンド リンクを使用すると、関連する相互に排他的な一連の選択肢から選択できます。



ドロップダウン リストとコンボ ボックスを使用すると、相互に排他的な値の一覧から選択できます。

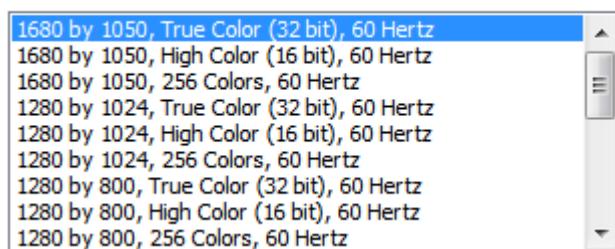


グループ ボックスを使用すると、一連のコントロールの関連性を示すことができます。



リンクを使用すると、別のページ、ウィンドウ、またはヘルプトピックへの移動、定義の表示、コマンドの開始、またはオプションの選択を実行できます。

List of valid modes:



リストボックスを使用すると、常に表示されている一覧に示された一連の値から選択できます。“単一選択リストボックス”では、相互に排他的な値の一覧からアイテムを1つ選択します。“複数選択リストボックス”では、値の一覧からアイテムを選択しないことも、1個以上選択することもできます。

Name	Date taken	Tags	Size	>>
Autumn Leaves				
Creek				
Desert Landscape				
Dock				
Forest				
Forest Flowers				
Frangipani Flowers				

リストビューを使用すると、単一選択または複数選択によるデータオブジェクトのコレクションの表示と操作が可能になります。



通知は、現在のユーザー操作とは無関係のイベントについて知らせます。

from Pictures (D:\User...\Pictures) to Pictures (D:\User...\Pictures)  
Calculating time remaining...



進行状況バーを使用すると、長い時間がかかる処理の進行状況を確認できます。



段階的表示コントロールを使用すると、データ、オプション、コマンドなどの追加情報の表示/非表示を切り替えることができます。

- Display as a link
- Display as a menu
- Don't display this item

ラジオボタンを使用すると、関連する相互に排他的な一連の選択肢から選択できます。



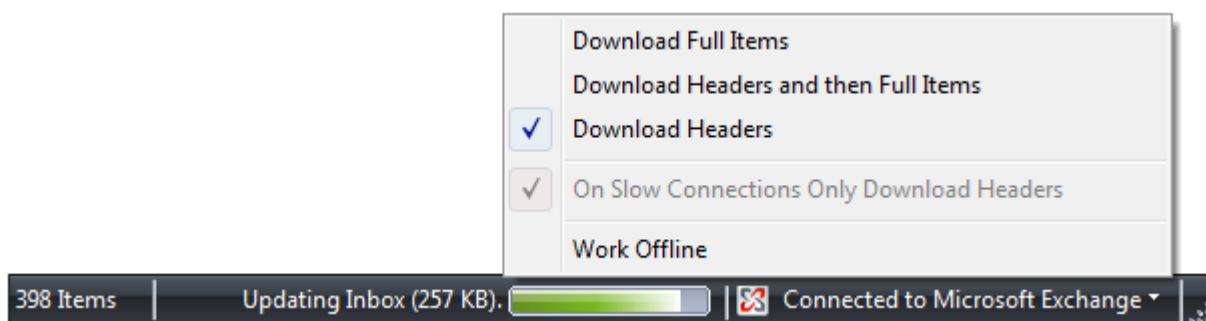
検索ボックスを使用すると、特定のオブジェクトやテキストをすばやく検索できます。



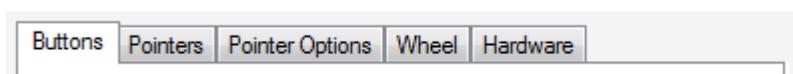
スライダーを使用すると、連続的な値範囲から選択できます。



スピンコントロールを使用すると、関連付けられた数値テキストボックス内の値を 1 単位ずつ変更できます。



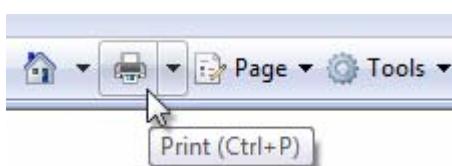
ステータスバーには、現在のウィンドウ、バックグラウンドタスク、またはその他のコンテキスト情報の状態に関する情報が表示されます。



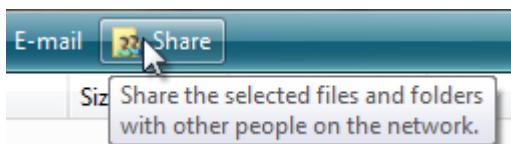
タブを使用すると、関連する情報をラベル付きの個別ページで表示できます。



テキストボックスを使用すると、テキストまたは数値の表示、入力、または編集が可能になります。



ツールヒントは、ラベル付けされていないコントロールのラベルとして機能します。



情報ヒントは、ユーザーがポイントしているオブジェクトについての説明を表示します。



ツリービューを使用すると、単一選択または複数選択による、オブジェクトの階層型コレクションの表示の操作が可能になります。

## バルーン

適切なコントロールかどうかの判断基準

使用パターン

ガイドライン

表示のタイミング

表示する時間の長さ

表示方法

パスワードや PIN 用のテキスト ボックス

その他のテキスト ボックス

対話操作

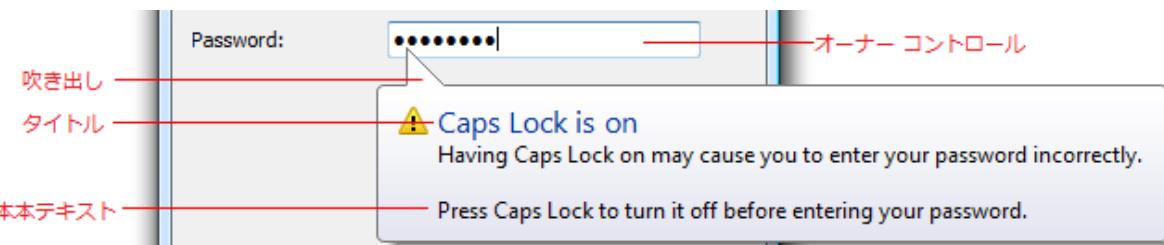
アイコン

アクセシビリティ

テキスト

ドキュメント

"バルーン" は、小さなポップアップ ウィンドウです。コントロールにおいて、それほど重大ではない問題や特殊な状態が発生していることをユーザーに通知します。



### 典型的なバルーン

バルーンは、アイコン、タイトル、および本文テキストで構成されます。この 3 つはいずれも省略可能です。ツールヒントおよび情報ヒントとは異なり、バルーンは吹き出しへになっているので、ソースがすぐにわかります。ソースは、通常、コントロールです。その場合、バルーンのソースとなっているコントロールは、[オーナー コントロール](#)と呼ばれます。

バルーンはそれほど重大ではない問題を通知するものであり、問題を防ぐものではありません。問題を防ぐ機能は、オーナー コントロールで提供することができます。ユーザーが操作をコミットする際、未対応の問題がある場合は、所有元のユーザーインターフェイス (UI) で対応する必要があります。

バルーンは、通常、テキスト ボックス、または値の変更にテキスト ボックスを使用するコントロール（コンボ ボックス、リスト ビュー、ツリー ビューなど）と共に使用します。ここに挙げた以外のコントロールでは適切な制約が課されるので、追加のフィードバック バルーンを提供する必要はありません。また、そのようなコントロールでは、問題があったとしても複数のコントロール間の不整合が関係していることが多いので、バルーンの使用は適切ではありません。制約がなく、[単一点エラー](#)がよく起こるという 2 つの条件に該当するのは、テキスト入力用のコントロールだけです。

通知は、[通知領域](#)のアイコンに対して表示される特別な種類のバルーンです。

注: [通知](#)、[ツールヒント](#)と[情報ヒント](#)、および[エラー メッセージ](#)に関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- 提供する情報が、問題または特殊な状態の説明かどうか。該当しない場合は、別のコントロールを使用します。コントロールに関する補足情報を表示する場合は、バルーンを使用するのではなく、代わりに[静的テキスト](#)、[情報ヒント](#)、[段階的表示](#)、または[プロンプト](#)を使用するようにします。
- その問題または特殊な状態が、すぐに検出できるものかどうか。入力時、または入力フォーカスがオーナー コントロールから外れたときに検出できるかどうかを判断します。該当しない場合は、[タスク ダイアログ](#)または[メッセージ ボックス](#)でエラー メッセージを表示するようにします。
- 問題の場合、重大な問題かどうか。該当する場合は、タスク ダイアログまたはメッセージ ボックスでエラー メッセージを表示するようにします。このようなエラー メッセージは対話操作が必須となるので、重大なエラーに適しています。バルーンでは対話操作は必須ではありません。
- 特殊な状態の場合、有効ではあるものの、意図せず設定された可能性が高い状態かどうか。該当する場合は、バルーンが適しています。有効ではない状態の場合は、その状態の発生を防ぐことが先決です。意図的に設定された可能性

が高い状態の場合は、何もする必要はありません。

- その問題または特殊な状態が、簡潔に説明できるものかどうか。該当しない場合は、別のコントロールを使用します。バルーンでは詳細な説明や補足情報は提供できません。
- 提供する情報が、現在ポインターがあるコントロールの説明かどうか。該当する場合は、対話操作の必要性がない限り、ヒントを使用します。
- 提供する情報が、ユーザーの現在のアクティビティに関連しているかどうか。該当しない場合は、代わりに[通知](#)または[ダイアログ ボックス](#)を使用するようにします。ユーザーの現在のアクティビティの外で表示されるバルーンは見落とされがちです。既定では、バルーンは 10 秒でタイムアウトになります。
- 情報のソースを特定の 1 つに限定できるかどうか。問題または状態のソースが複数あったり特定できない場合は、代わりに[インプレース メッセージ](#)または[ダイアログ ボックス](#)を使用します。

間違った例:



この例では、問題はユーザー名にあることもパスワードにあることも考えられます。しかし、バルーンで問題を報告すると、パスワードだけに問題があるように見え、間違ったユーザー名を入力したことに対するフィードバックが適切に返されません。

バルーンは、情報ヒント、ダイアログ ボックス、およびインプレース メッセージの代わりに使用できます。ツールヒントおよび情報ヒントとの違いは次のとおりです。

- バルーンは、現在のポインターの場所と関係なく表示できます(ソースを示す吹き出しがあります)。
- バルーンは、タイトル、本文テキスト、およびアイコンで構成されます。
- バルーンは対話的にクリックできますが、ヒントはクリックできません。

モーダル ダイアログ ボックスとの違いは次のとおりです。

- バルーンに入力フォーカスが移動したり、バルーンに対話操作が必須になることはありません。
- バルーンのソースは特定の 1 つに限定されます。モーダル ダイアログのソースは複数あったり特定のソースというものが存在しないことがあります。

インプレース メッセージとの違いは次のとおりです。

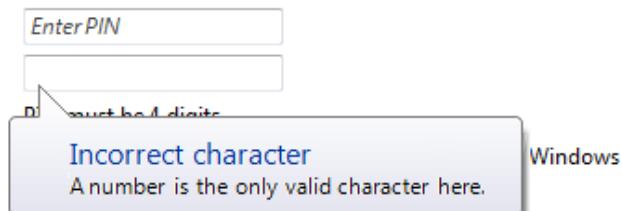
- インプレース メッセージよりバルーンのほうが目に付きます。
- インプレース メッセージの表示に必要な空き画面領域または動的レイアウトは、バルーンには必要ありません。
- バルーンはタイムアウトになると自動的に消えます。

## 使用パターン

バルーンの使用パターンを以下に示します。

入力の問題 オーナー コントロールに入力フォーカスがある状態でバルーンをエラー メッセージの表示に使用する单一のオーナー と、入力フォーカスは移動しないまま、ユーザーは問題にすぐに気付くことができます。問題を修正するには、ユーザーが入力内容を変更するか最初から入力し直します。間違った入力がオーナー コントロールで無視される場合、ユーザーが変更する必要はありません。問題がそれほど重大なものではないボックス)を ので、エラー アイコンは不要です。

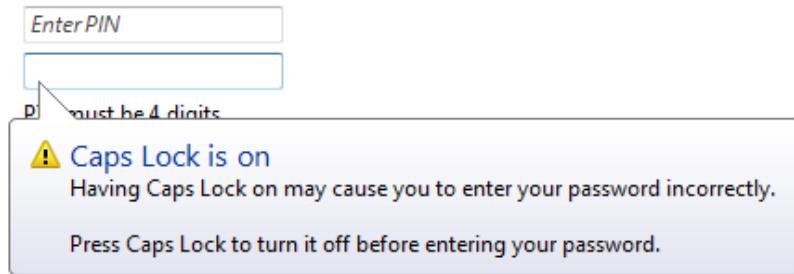
ソースとする、  
それほど重大で  
はないユーザー  
入力の問題を示  
すパターン。



それほど重大ではないユーザー入力の問題を報告するバルーン。

特殊な状態 最大入力文字数を超えたときや、誤って Caps Lock が設定されているときなど、特殊な状態が発生した  
オーナー コント ときにはすぐバルーンでユーザーに警告すると、ユーザーが余計なストレスを感じることがなくなりま  
ロールが、入力 す。意図的に設定された状態の可能性もあるため、入力フォーカスの移動や対話操作の強制をしない  
に何らかの影響 フィードバックにすることが大切です。特にパスワードや PIN 用のボックスでは、ユーザーが間違いに  
を及ぼす状態に 気付きにくいので、バルーンは重要視されます。このパターンのバルーンには警告アイコンが付きます  
あることを示す す。

パターン(意図せ  
ず設定された可  
能性の高い状態  
で、ユーザーが  
入力への影響を  
認識していない  
可能性がある)。



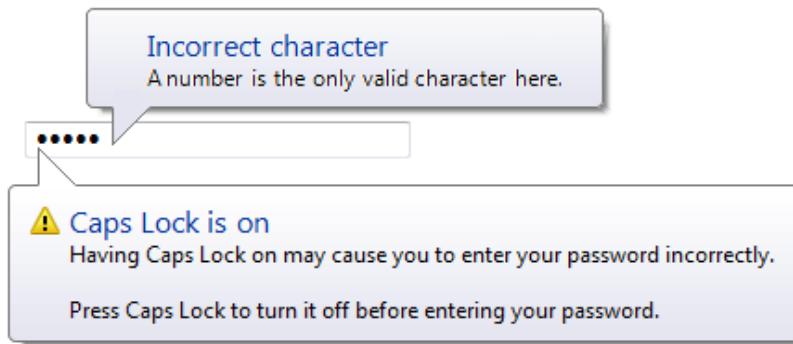
特殊な状態を報告するバルーン。

## ガイドライン

### 表示のタイミング

- バルーンは、問題または特殊な状態が検出されたときにすぐ、間を置かずに表示します（繰り返しの表示になってしまいません）。
  - 個々の文字や最大入力文字数に関する問題の場合は、入力時にバルーンを表示します。
  - 空白文字を含む値が禁止される場合など、入力値に関する問題の場合は、入力フォーカスがオーナー コントロールから移動するときにバルーンを表示します。これ以外のタイミングで、ユーザーが有効な値にするつもりで入力している最中にバルーンを表示すると、操作の妨げになります。
- 一度に 1 つのバルーンだけを表示します。複数のバルーンを一度に表示するとユーザーは圧倒されます。1 つのイベントから複数の問題が発生した場合は、すべての問題を 1 回で示すか、最も重要な問題だけを報告します。

間違った例：



この例では、2 つの問題が一度に報告されており、不適切な使用といえます。

### 表示する時間の長さ

- 次の状態になったらバルーンを消します。
  - 問題が解決するか、特殊な状態ではなくなったとき。
  - ユーザーが有効なデータを入力したとき（入力の問題を示す場合）。
  - バルーンがタイムアウトになったとき。既定では、10 秒たつとバルーンは消えます。ただしユーザーは SPI\_MESSAGE DURATION システム パラメーターを変更することでこの秒数を変更できます。

- 問題が解決されるまでユーザーが操作を続行できない場合は、タイムアウトを外します。開発者向け情報: Win32 では、TTM\_SETDELAYTIME メッセージで表示時間を設定できます。

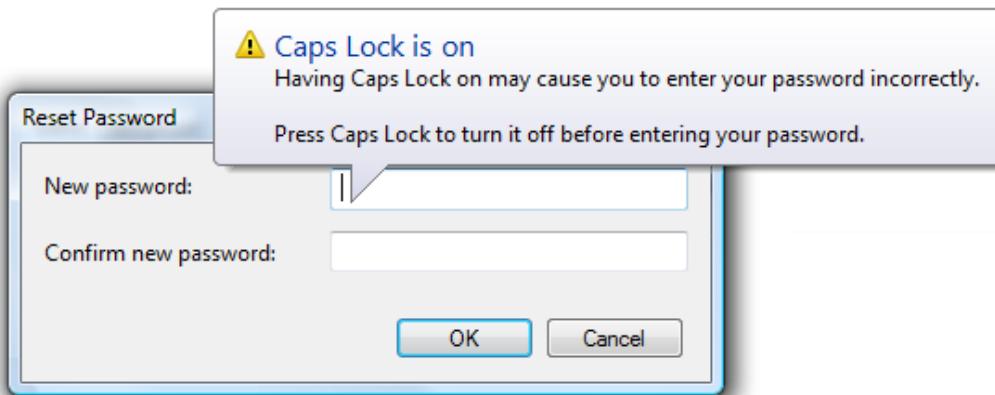
## 表示方法

- バルーンは、オーナー コントロールの下に表示します。オーナー コントロールの下に表示することで、ユーザーはオーナー コントロールとそのラベルを含むコンテキストを確認できます。Microsoft® Windows® では、バルーン全体が画面に表示されるように、バルーンの位置が自動的に調整されます。既定の動作では、通知の場合のように、バルーンはオーナー コントロールの上に表示されます。

正しい例:



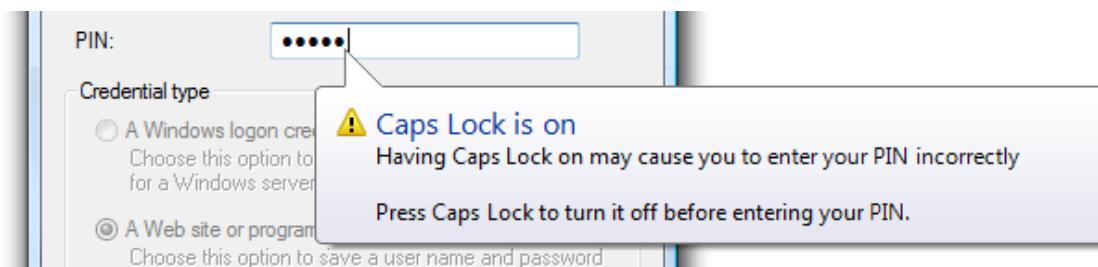
間違った例:



間違った例では、バルーンはオーナー コントロールの上の、最適ではない位置に表示されています。

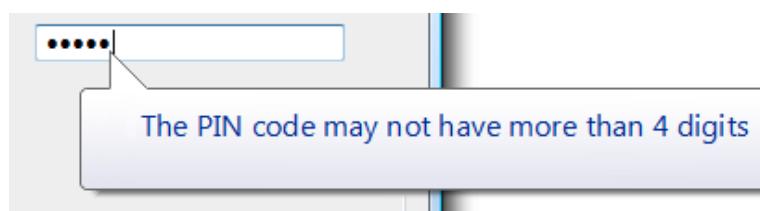
## パスワードや PIN 用のテキスト ボックス

- Caps Lock が設定されていることを示すときにバルーンを使用します。テキストは、次の例のようにします。



この例では、PIN テキスト ボックスで、Caps Lock が設定されていることを示すバルーンを表示しています。

- ユーザーが最大入力文字数を超えて入力しようとしたときにバルーンを表示します。パスワードや PIN 用のテキスト ボックスの場合、通常のテキスト ボックスよりも、最大入力文字数を超えたことがわかりにくくなっています。



この例では、ユーザーが最大入力文字数を超えて入力しようとしたときにバルーンを表示しています。

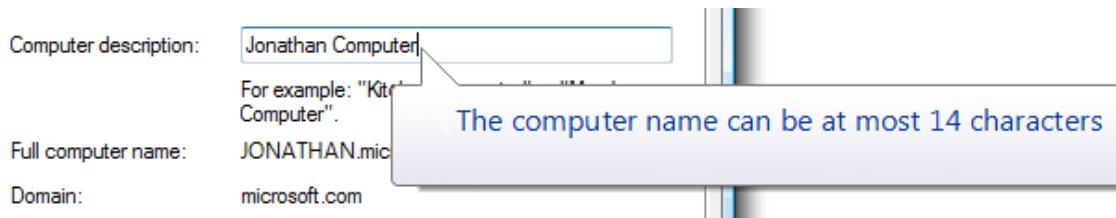
- ユーザーが正しくない文字を入力したときにバルーンを表示します。ただし使用文字の制限を示すことは、パスワードや PIN のセキュリティが低くなるのでお勧めしません。情報の漏えいを防ぐには、バルーンで示す内容を、有効なパスワードまたは PIN に関するドキュメント化された事実だけにします。



この例では、PIN が数値でなければならないことを示すバルーンを表示しています。

## その他のテキスト ボックス

- 重要な短いテキスト ボックスには、使い慣れないユーザー向けに、最大入力文字数を超えて入力しようとしたときにバルーンを表示するようにします。例としては、ユーザー名やアカウント名などのテキスト ボックスが挙げられます。ユーザーが最大入力文字数を超えて入力しようとすると、テキスト ボックスでは警告音が鳴りますが、使い慣れないユーザーは警告音の意味を理解しない可能性があります。



この例では、ユーザーが最大入力文字数を超えて入力しようとしたときにバルーンを表示しています。

## 対話操作

- ユーザーがバルーンをクリックしたら、単純にバルーンを消します(他の UI を表示したり他の効果を発生させたりしません)。通知とは異なり、バルーンに閉じるボタンは使用しません。

## アイコン

- 使用パターンに応じてアイコンを選択します。

パターン	アイコン
入力の問題	アイコンを使用しません。このパターンでエラー アイコンを使用しないことは、Windows のトーンに関するガイドラインとも一致します。
特殊な状態	標準の 16 × 16 ピクセルの警告アイコンを使用します。

## アクセシビリティ

適切に使用するバルーンは、アクセシビリティを高めます。アクセシビリティを高めるバルーンにするには、以下のことに留意します。

- ユーザーの現在のアクティビティに関連するバルーンだけを表示します。
- バルーン テキストは常に簡潔にします。簡潔なバルーン テキストは、低視力のユーザーにとっても読みやすく、スクリーン リーダーで読み取られる場合も介入が最小限となります。
- 問題や状態が再発したときは常にバルーンを再表示します。

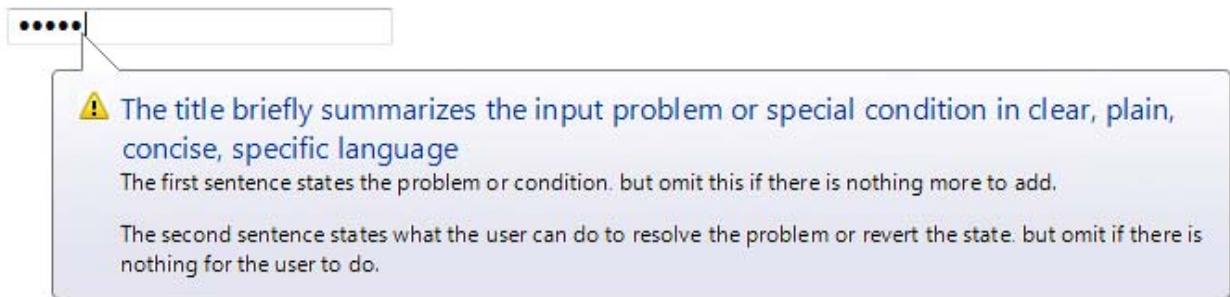
## テキスト

### タイトル テキスト

- 入力の問題または特殊な状態を、明確でわかりやすい言葉で短く簡潔に要約したタイトル テキストを使用します。ユーザーが最小の努力でバルーンの目的をすばやく理解できるテキストである必要があります。
- 語句または文を使用し、末尾に句読点は付けません。
- センテンス スタイルの大文字化を使用します。
- ローカライズを想定して、英語の場合の文字数は半角換算で 48 文字以内とします。タイトルの最大文字数は半角 63 文字ですが、ローカライズ用に 30% 以上の空白を残しておく必要があります。

## 本文テキスト

- ・本文テキストの1文目では、ユーザーとの関連をはっきりと示す形で問題または状態を説明します。タイトルに含まれている情報は繰り返しません。追加する情報がない場合、本文テキストは省略します。
- ・2文目では、問題の解決または状態の解消のためにユーザーができるることを説明します。[スタイルとトーン](#)に関するガイドラインにもあるように、この文であまり丁寧すぎる言葉を使用する必要はありません。1文目と2文目の間には、改行を2つ挿入します。



この例では、バルーンテキストは標準的なレイアウトで構成されています。

- ・問題の解決方法や状態の解消方法を、明白な場合でも説明します。ただし、問題の説明と解決方法の間で冗長になる部分があれば省きます。次の場合は例外です。
  - ・簡潔に説明できない場合や、かなり冗長になる場合は、解決方法を省略します。
  - ・正しくない文字が無視される場合など、ユーザーの対応が必要ない場合は、解決方法を省略します。
- ・文を使用し、末尾に句点を付けます。
- ・センテンススタイルの大文字化を使用します。
- ・ローカライズを想定して、英語の場合の文字数は半角換算で200文字以内とします。本文テキストの最大文字数は半角255文字ですが、ローカライズ用に30%以上の空白を残しておく必要があります。

## ドキュメント

バルーンに言及するときは、以下のこと留意します。

- ・大文字と小文字の区別を含め、タイトルテキストを正確に引用します。
- ・"通知"や"警告"、"アラート"ではなく、"バルーン"としてコンポーネントを示します。
- ・タイトルテキストを二重引用符 (" ") で囲み、可能な場合は太字にします。

## チェック ボックス

適切なコントロールかどうかの判断基準

使用パターン

ガイドライン

全般

今後、この<アイテム>を表示しない

従属コントロール

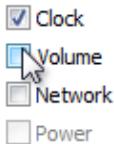
既定値

推奨されるサイズと間隔

ラベル

ドキュメント

"チェック ボックス" を使うと、ユーザーは正反対の 2 つの選択肢のうちどちらかを選ぶことができます。チェック ボックスのラベルは、これをオンにした場合の状態を表します。オフは、オンになっているときと正反対の状態を表します。したがって、チェック ボックスは、オプションのオンとオフを切り替える場合、またはあるアイテムを選択するかどうかを選ぶ場合にのみ使用する必要があります。



### 典型的なチェック ボックスのグループ

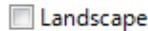
注: [レイアウト](#)に関するガイドラインは、別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- チェック ボックスが、オプションのオンとオフの切り替え、またはあるアイテムの選択/選択解除に使用されるかどうか。該当しない場合は、別のコントロールを使用します。
- オンにしたときの状態とオフにしたときの状態が正反対で、選択肢があいまいではないかどうか。該当しない場合は、[ラジオ ボタン](#)または[ドロップダウンリスト](#)を使って、それぞれの状態ごとにラベル付けできます。
- グループ化して使用する場合、グループが独立型選択肢で構成され、ユーザーが選択しないでおく、または 1 件以上選択することができるかどうか。該当しない場合は、ラジオ ボタンや[チェック ボックスのツリービュー](#)など、依存型選択肢向けのコントロールを検討します。
- グループ化して使用する場合、グループが依存型選択肢で構成され、ユーザーが 1 つ以上選択する必要があるかどうか。該当する場合は、チェック ボックスのグループを使用し、オプションが何も選択されていない場合にエラーとして処理されるようにします。
- グループ内のオプションの数が 10 項目以下であるかどうか。使用する画面領域はオプションの数に比例するので、チェック ボックスの数は 10 個以下になるようにします。オプションが 10 項目を超える場合は[チェック ボックスリスト](#)を使用します。
- ラジオ ボタンの方が適していないか。チェック ボックスが適しているのは、オプションのオン/オフを切り替える場合のみです。オプションがまったく別の内容である場合は、ラジオ ボタンを使用できます。チェック ボックスとラジオ ボタンの両方が考えられる場合は、以下に従います。
  - チェック ボックスがオフになっていることによって示される意味がはっきりしない場合は、ラジオ ボタンを使用します。

間違った例:



この例では、[横長] の反対の選択肢が明確ではないので、チェック ボックスは適しません。

正しい例:



この例では、選択肢が相反するものではないため、ラジオ ボタンの方が適しています。

- 通常はチェック ボックスを使ってもかまわないような場合でも、可能な選択肢をはっきりと示すために、 ウィザード ページではラジオ ボタンを使います。
- 十分な画面領域を確保でき、画面領域を多く使用するだけの重要性を持つオプションであれば、ラジオ ボタンを使用します。それ以外の場合は、チェック ボックスまたはドロップダウン リストを使用します。

間違った例:

- Show this again  
 Don't show this again

この例のオプションは、ラジオ ボタンを使うほど重要ではありません。

正しい例:

- Don't show this message again

この例のオプションは、それほど重要ではないので、チェック ボックスを使うのが効率の良い画面領域の使い方です。

- ウィンドウ上に他にもチェック ボックスがある場合は、チェック ボックスを使用します。
- オプションの内容がデータではないプログラム オプションかどうか。オプションの値は、コンテキストや他のデータに基づかない値である必要があります。データの場合は、チェック ボックス リストまたは[複数選択リスト](#)を使用します。

## 使用パターン

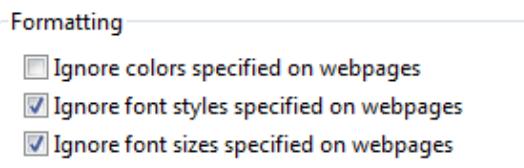
チェック ボックスにはいくつかの使用パターンがあります。

単独の選択  Remind me one week before this change occurs

単独の選択には、単独のチェック ボックスを使用します。  
 は、単独のチェック ボックスを使用します。  
 チェック ボックスを使用しま  
 す。

独立型選択肢 ラジオ ボタンなどの逐一のコントロールとは異なり、ユーザーはチェック ボックスのグループの中から、各オプションを好きに組み合わせて選択できます。  
 (選択しない、ま  
 たは 1 項目以上

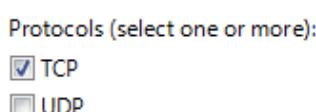
の選択が可能) 何も選択しな  
 い、または 1 件  
 以上を選択でき  
 るようにする場  
 合は、チェック  
 ボックスのグ  
 ループを使  
 用します。



独立型選択肢の場合は、チェック ボックスのグループを使用します。

依存型選択肢 (1 件以上の選択が必要) 依存型選択肢を 1 項目以上選択するよう示しておくことが必要な場合があります。Microsoft® Windows® にはこのような入力を直接サポートするコントロールがないため、チェック ボックスのグループを使用して、オプションがまったく選択されていない場合にはエラーとして処理するのが最も良い方法です。

する場合も、  
 チェック ボック  
 スのグループを  
 使用できます。



チェック ボックスのグループが使用されています。ここではプロトコルを少なくとも 1 つ選択する必要があります。

選択の混在  Attributes: Read-only

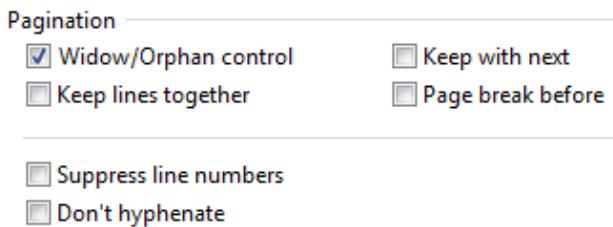
チェック ボック  
 スにはオン/オ  
 フの状態のほ  
 フの状態のほ

か、複数の項目に対する選択が混在していることを示す状態もあります。オブジェクトのすべてではなく、一部についてオプションが設定されていることを表します。

## ガイドライン

### 全般

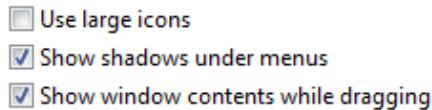
- 関連するチェックボックスをグループ化します。関連性のあるオプションをまとめ、関連性のないオプションを別にすることで、10個以下のオプションを含むグループにし、必要に応じて複数のグループを使用します。



関連する独立したオプションのグループの例。

- チェックボックスのグループを **グループボックス** を使って整理することを再検討します。これは画面上に不要な混乱を招きやすくなるので注意します。
- チェックボックスを論理的な順序で一覧にします。関連性が高いオプションをまとめてグループにする、最も一般的なオプションを先頭にするなど、自然な順序に従います。アルファベット順は言語に依存し、ローカライズ時に順序を再現できないのでお勧めしません。
- チェックボックスを並べる方向は、横ではなく縦にします。横方向に並べると読みにくくなります。

正しい例:



この例では、チェックボックスが正しく並んでいます。

間違った例:



この例では、横方向に並んでいるため読みにくくなっています。

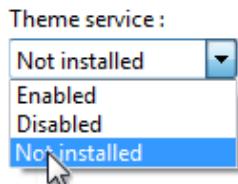
- 第3の状態を表す目的には、混在状態を使用しません。混在状態は、オプションが、子オブジェクトのすべてではなく、一部にだけ設定されていることを表すために使用します。混在状態は、ユーザーが自らこれを設定できるようになるのではなく、子オブジェクトの選択によって反映されるようにする必要があります。混在状態は、単独のアイテムの第3の状態としては使用しません。第3の状態を表すには、代わりにラジオボタンまたはドロップダウンリストを使います。

間違った例:



この例では、混在状態を使用して、テーマサービスがインストールされていないことを示そうとしています。

正しい例:



この例では、ユーザーは 3 つの明確なオプションから選択できます。

- 混在状態のチェックボックスをクリックすると、すべて選択、すべてクリア、元の混在状態が繰り返されるようにします。ユーザーにとって設定が複雑であったり、未知のものであったりする場合があるため、寛容性を確保する観点から、元の混在状態に戻せるようにしておくことが重要です。これが不可能な場合、確実に元の混在状態に戻すには、タスクを取り消して最初からやり直すしかありません。
- チェックボックスは、進行状況のインジケーターとして使用しません。代わりに、[進行状況のインジケーター](#)コントロールを使います。

間違った例:

Removing program...

- Shutting down program
- Removing program files
- Removing registry settings
- Removing desktop and Start menu items

これは、チェックボックスを進行状況のインジケーターとして使った間違った例です。

正しい例:

Removing program...



典型的な進行状況バーの例です。

- 無効化されたチェックボックスを正しい選択状態で表示します。無効化されたチェックボックスはユーザーが選択状態を変更することはできませんが、無効化されたチェックボックスの状態がユーザーにわかり、その状態と合致する結果が得られるようにしておく必要があります。

間違った例:

Windows can read and highlight this list of tools automatically. Press the spacebar to select the highlighted option.

- Always read this section aloud
- Always scan and highlight each option

この例では、チェックボックスが無効化されている場合に音声は読み上げられないため、[このセクションの内容を常に音声で読み上げます] チェックボックスはオフになっている必要があります。

- チェックボックスによる選択は、以下の目的には使用しません。
  - コマンドの実行。
  - 追加の情報入力を行うためのダイアログボックスなど、他のウィンドウの表示。
  - 選択されているコントロールに関連する、他のコントロールの動的な表示(このようなイベントは、スクリーンリーダーで検出できません)。

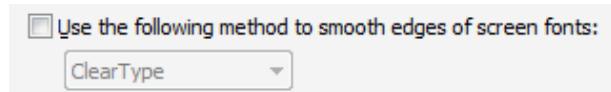
今後、この <アイテム> を表示しない

- "今後、この<アイテム>を表示しない" というオプションを使って、同じダイアログボックスが繰り返し表示されないようにユーザーが設定できるようにすることを検討します。ただし、これは他に良い手段がない場合に限ります。まずは、そのダイアログがユーザーにとって本当に必要かどうかを判断します。ユーザーにとって本当に必要なダイアログであれば常に表示し、そうでなければそのダイアログを省きます。

ガイドラインの詳細と例については、「[ダイアログボックス](#)」を参照してください。

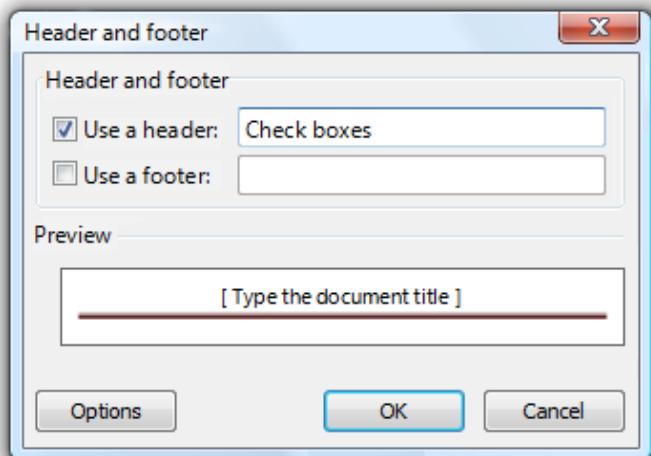
従属コントロール

従属コントロールは、チェック ボックスおよびそのラベルの右側または下方に配置します。下方に配置する際は、インデントしてチェック ボックスのラベルに揃えます。チェック ボックスのラベルの末尾には、コロンを付けます。



この例では、チェック ボックスおよびその従属コントロールは、ラジオ ボタンのラベルおよびそのアクセキーを共有しています。

- 従属している編集可能なテキスト ボックスおよびドロップダウン リストがチェック ボックスのラベルを共有している場合は、それらのボックスを有効なままにしておきます。ユーザーがボックスに入力したり何かを貼り付けたりしたときに、対応するオプションが自動的に選択されるようにし、対話操作を単純化します。



この例では、ヘッダーやフッターを入力すると、オプションが自動的に選択されます。

- チェック ボックスを、ラジオ ボタンまたは他のチェック ボックスと入れ子にする場合、上位レベルのオプションが選択されるまで、従属コントロールを無効にしておきます。こうすることで、従属コントロールの意図に関する混乱を回避できます。
- チェック ボックスとその従属コントロールは、タブ オーダーで隣接させます。
- オプションを選択すると従属チェック ボックスがオンになることを示す場合は、関係を明確にするために、これらのチェック ボックスを明示的にオンにします。

間違った例:

Filter web content

Choose a web restriction level.

Highest restriction - Only websites on the Allowed websites list  
 High restriction - Kids websites only  
 Medium Restriction  
 Low Restriction  
 Custom

Check the content you want to block:

How do these categories work?

<input type="checkbox"/> Alcohol	<input type="checkbox"/> Pornography
<input type="checkbox"/> Bomb making	<input type="checkbox"/> Sex education
<input type="checkbox"/> Drugs	<input type="checkbox"/> Tobacco
<input type="checkbox"/> Gambling	<input type="checkbox"/> Weapons
<input type="checkbox"/> Hate speech	<input type="checkbox"/> Web e-mail
<input type="checkbox"/> Mature content	<input type="checkbox"/> Web chat

この例では、従属チェック ボックスがオンになっていません。

正しい例:

## Filter web content

Choose a web restriction level.

- Highest restriction - Only websites on the Allowed websites list
- High restriction - Kids websites only
- Medium Restriction
- Low Restriction
- Custom

Check the content you want to block:

How do these categories work?

- |  |   |
|--|---|
| <input checked="" type="checkbox"/> Alcohol        | <input checked="" type="checkbox"/> Pornography   |
| <input checked="" type="checkbox"/> Bomb making    | <input checked="" type="checkbox"/> Sex education |
| <input checked="" type="checkbox"/> Drugs          | <input checked="" type="checkbox"/> Tobacco       |
| <input checked="" type="checkbox"/> Gambling       | <input checked="" type="checkbox"/> Weapons       |
| <input checked="" type="checkbox"/> Hate speech    | <input checked="" type="checkbox"/> Web e-mail    |
| <input checked="" type="checkbox"/> Mature content | <input checked="" type="checkbox"/> Web chat      |

この例では、従属チェックボックスがオンになっているので、選択したオプションとの関係が明確になっています。

- 他の方法ではかえって複雑になる場合は、依存型チェックボックスを使用します。チェックボックスは独立型のオプションに使用することが推奨されますが、ラジオボタンなど他の方法を用いるとかえって複雑になることがあります。

正しい例:

### Effects

- |  |  |
|--|--|
| <input checked="" type="radio"/> No strikethrough      | <input checked="" type="radio"/> No effect |
| <input type="radio"/> Strikethrough                    | <input type="radio"/> Shadow or outline    |
| <input type="radio"/> Double strikethrough             | <input type="checkbox"/> Shadow            |
|  | <input type="checkbox"/> Outline           |
| <input checked="" type="radio"/> No super or subscript | <input type="radio"/> Emboss               |
| <input type="radio"/> Superscript                      | <input type="radio"/> Engrave              |
| <input type="radio"/> Subscript                        |  |

この例では、ラジオボタンが正しく使用されていますが、無駄に複雑になっています。

より良い例:

### Effects

- |   |  |
|---|--|
| <input checked="" type="checkbox"/> Strikethrough | <input checked="" type="checkbox"/> Shadow |
| <input type="checkbox"/> Double strikethrough     | <input type="checkbox"/> Outline           |
| <input type="checkbox"/> Superscript              | <input type="checkbox"/> Emboss            |
| <input type="checkbox"/> Subscript                | <input type="checkbox"/> Engrave           |

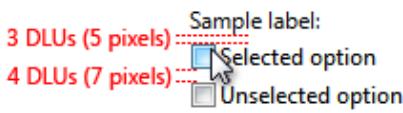
この例では、チェックボックスが使用されてシンプルになっているので、ユーザーはオプション間の複雑な関係に惑わされることなく、目的のオプションの選択に集中できます。

**重要:** このガイドラインを適用できる状況は、きわめて限られています。依存関係を示すことが明確さの助けにはならず、かえってひどく複雑になることになる場合にのみ適用します。上記の例では、ユーザーが上付きと下付きを同時に選択しようとすることはあまりなく、仮にこれを行おうとしても、2つのオプションが排他的なものであることは容易にわかります。

## 既定値

- ユーザー オプションのチェックボックスでは、最も安全(データの消失やシステムアクセスが失われることを防ぐため)で、最もセキュリティの高い、プライベートな状態を既定に設定します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用される値または最も便利な値を選択します。

## 推奨されるサイズと間隔



Check1 10 DLUs (17 pixels)

チェック ボックスに推奨されるサイズと間隔

## ラベル

チェック ボックスのラベル

- すべてのチェック ボックスにラベルを付けます。
- 各ラベルに、一意な[アクセス キー](#)を割り当てます。ガイドラインについては、「[キーボード](#)」を参照してください。
- [センテンス スタイルの大文字化](#)を使用します。
- ラベルには語句または "...する"などの言い切りの文を使用し、末尾に句読点は付けません。
  - 例外: チェック ボックスのラベルが従属コントロールのラベルでもあり、ラベルの後ろに従属コントロールが続く場合は、ラベルの末尾にコロンを使用します。
- ラベルはそのチェック ボックスがオンのときの状態を説明するものにします。
- チェック ボックスのグループでは、同じ文法構造の表現を使用し、すべてのラベルの長さがおおよそ同じになるようにします。
- チェック ボックスのグループでは、オプション間の違いがわかるラベルテキストにします。すべてのオプションが同じテキストで始まっている場合は、そのテキストをグループ ラベルに移動します。
- 肯定文を使用します。チェック ボックスをオンにするとあるアクションが実行されないなどの言い回しはラベルに使用しません。
  - 例外: [今後、この<アイテム>を表示しない] チェック ボックス。

間違った例:

Turn off System Restore on all drives

この例のオプションでは、肯定文が用いられていません。

- ラベルではオプションのみを説明します。ラベルを簡潔にすることで、メッセージおよびドキュメントで言及することが容易になります。オプションに詳細な説明が必要な場合は、句点を付けた完全な文として[静的テキスト](#) コントロール内で説明します。

注: 1つのチェック ボックスに説明を追加したからといって、すべてのチェック ボックスに説明を追加する必要はありません。関連情報は可能な限りラベルで提供し、必要がある場合にのみ説明を追加します。ラベルの言い回しを変えただけの説明は、一貫性を保つためであっても追加しないでください。

Hide modes that this monitor cannot display

Clearing this check box allows you to select display modes that this monitor cannot display correctly. This may lead to an unusable display or damaged hardware.

この例では、チェック ボックスのラベルの下に説明テキストが追加されています。

- 強く推奨されるオプションに対しては、ラベルに "(推奨)" を追加することを検討します。補足的なメモに対してではなく、コントロール ラベルに追加するようにします。
- 複数行にまたがるラベルを使用する必要がある場合は、ラベルの先頭をチェック ボックスと揃えます。
- 従属コントロール、それに含まれる値、または単位のラベルを使用して、文や句を作成しないでください。文の構造は言語ごとに異なるので、このように設計するとローカライズできなくなります。

間違った例:

Create a computer account in the  domain

この例では、チェック ボックスのラベル内にテキスト ボックスが配置されているので不適切です。

## チェック ボックス グループのラベル

- それぞれのラベルの末尾にコロンを付けます。
- ラベルにはアクセス キーを割り当てません。ラベルにアクセス キーは不要であり、割り当てるに他のアクセス キーの割り当てが困難になります。
- 依存型選択肢を 1 つ以上選択する場合は、ラベルで条件を説明します。

正しい例:

Protocols:



この例では、1 つしか選択できないという印象をユーザーに与えることがあります。

より良い例:

Protocols (select one or more):



この例では、複数選択が可能であることがユーザーにはっきりと伝わります。

## ドキュメント

チェック ボックスに言及するときは、以下のことに留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、アクセス キーを示すかっこや下線付き文字、およびコロンは含めません。また、"チェック ボックス" という語を含めます。
- チェック ボックスに言及する場合は、"チェック ボックス" と記述します。"オプション"、"チェックボックス" (スペースなし)、または単に "ボックス" とは記述しません。"ボックス" だけではローカライズ時にあいまいに受け取られます。
- ユーザー操作を説明する場合は、"オンにする" と "オフにする" を使用します。
- ラベルは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例: [下線] チェック ボックスをオンにします。

## コマンド ボタン

適切なコントロールかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

全般

分割ボタン

既定値

推奨されるサイズと間隔

ラベル

ドキュメント

"コマンド ボタン" は、操作をすぐに開始するために使います。

OK

### 典型的なコマンド ボタン

既定のコマンド ボタンは、ユーザーが **Enter** キーを押すと呼び出されます。既定のコマンド ボタンは開発者によって割り当てられますが、ユーザーは **Tab** キーを押すことにより、"既定のコマンド ボタン" を順次変更できます。

注: [レイアウト](#)に関するガイドラインは、別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- コマンド ボタンが、すぐに操作を開始するために使われるものであるかどうか。該当しない場合は、別のコントロールを使用します。
- リンクの方が適していないか。次のような場合はリンクを使ってください。
  - 他のページ、ウィンドウ、またはヘルプ トピックに移動するための操作である場合。  
例外: ウィザードでは [戻る]、[次へ] のコマンド ボタンを使います。
  - コマンドがテキスト本文内に埋め込まれる場合。
  - コマンドが副次的な性質のものである場合。つまり、ウィンドウの主目的に関連しないコマンドである場合。この場合は、軽量コマンド ボタンまたはリンクの方が適切です。
  - コマンドがメニューまたは関連するリンク グループの一部である場合。
  - ラベルが長い (5 語あるいは15文字以上) ため、コマンド ボタンの外観が不自然になる場合。
- ラジオ ボタンと汎用コマンド ボタンを組み合わせた方が適切ではないかどうか。以下のいずれかに該当する場合、特定のコマンド ボタンのセットの代わりに、ラジオ ボタンを汎用コマンド ボタン ([OK]、[キャンセル]) と組み合わせて使用することがよくあります。
  - 5 つ以上の操作が考えられる場合。
  - ユーザーが判断の前に追加の情報を確認する必要がある場合。
  - ユーザーが決定前に (追加情報を見るなど) 選択肢について何らかのやり取りを行う必要がある場合。
  - ユーザーが別のコマンドの代わりに、オプションとして選択肢を確認する場合。

正しい例:



この例では、ラジオ ボタンを [OK]、[キャンセル] ボタンと組み合わせ、オプションに関する追加情報が提供されています。

間違った例:



この例では、コマンド ボタンが単独で使われているので、ユーザーが情報に基づいた判断を行うのが難しくなっています。

注: より詳細な比較については「[コマンド ボタンとリンクの比較](#)」を参照してください。

## デザイン コンセプト

### 省略記号の使用

コマンド ボタンは直ちに開始する操作のために使うものですが、その操作を実行するために、さらに詳しい情報が必要な場合があります。追加の情報(確認を含む)が必要なコマンドであることは、ボタンラベルの末尾に省略記号 (...) を付けて示します。

[Print...](#)

この例では、[印刷...] コマンドをクリックすると、情報を入力するための [印刷] ダイアログ ボックスが表示されます。

[Print](#)

この例では、[印刷] コマンドをクリックすると、ユーザーの追加操作なしでドキュメントのコピーが既定のプリンターに 1 部印刷されます。

操作の実行前に、さらに細かい選択ができるよう、場合によっては操作を完全に取り消すことができるよう、省略記号を適切に使うことが大切です。省略記号によって与えられた視覚的な手掛けにより、ユーザーは安心してソフトウェアを使用できます。

ただし、新しいウィンドウを表示する操作すべてで省略記号を使う必要はありません。操作を実行するのに追加の情報入力が必要な場合だけに使います。したがって、その動詞によって別のウィンドウが表示されることが明らかにわかるコマンドボタンでは、省略記号を付ける必要がありません。バージョン情報、詳細設定、ヘルプ(またはヘルプトピックにリンクするその他のコマンド)、オプション、プロパティ、設定などのコマンドがこれに当たります。

一般に、ユーザーインターフェイスでは、省略記号は未完了であることを示すために使われます。他のウィンドウを表示するコマンドは未完了であり、つまり、別のウィンドウを表示する必要があるということです。その操作を実行するために追加の情報は必要ありません。これは省略記号を使用する価値がほとんどない状況で画面が輻輳するのを防ぐための方針です。

注: 省略記号が必要かどうか判断する際、[権限の昇格](#)について考慮する必要はありません。コマンドの実行に必要な情報ではなく、実行権限を確認するためのものだからです。権限の昇格が必要である旨はセキュリティシールドで示されます。

### 最も重要な点

コマンドボタンには処理内容を一言で表す簡潔なラベルを付け、省略記号を適切に使ってください。

## 使用パターン

コマンドボタンにはいくつかの使用パターンがあります。

標準コマンドボタン  
すぐに操作を開始するために使用できます。



既定のコマンドボタン  
ウィンドウ内  
で、ユーザーが  
Enterキーを押し  
たときにアク  
ティブになるコ  
マンドボタンを  
示します。



既定のコマンドボタン  
ユーザーは Tab キーを押すことにより、既定のコマンドボタンを順次変更できます。入力フォーカスがコマンドボタン以外にある場合は、既定のボタンの属性を持つコマンドボタンが既定になります。ウィンドウ内で既定にできるのは、1つのコマンドボタンだけです。

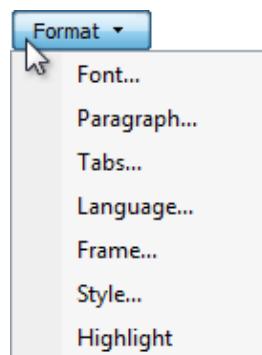
軽量コマンドボタン  
標準コマンドボタンと似ていますが、マウスポインターが重ねない限り、(リンクに似た)非常に軽い外観をしています。マウスカーソルが上に置かれるとボタン枠が描画されます。



この例のように、ユーザーがマウスポインターを重ねない限り、(リンクに似た)非常に軽い外観をしています。マウスカーソルが上に置かれるとボタン枠が描画されます。

軽量コマンドボタンは、標準コマンドボタンと同じような状況で使いますが、枠を常に表示したくない場合に使います。あまり強調したくないが、リンクを使用するのが適切ではないような場合に向いています。

メニュー ボタン  
関連するいくつかのコマンドをまとめるメ  
ニューが必要な場合に使います。



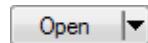
関連するいくつかのコマンドをまとめたメニュー ボタン

ダイアログボックス、ツールバー、その他のメニュー バーを持たないウィンドウなど、メニュー バー

が使いにくい場合に向いています。下向きの三角形は、これをクリックすると下にメニューが現れることを表します。

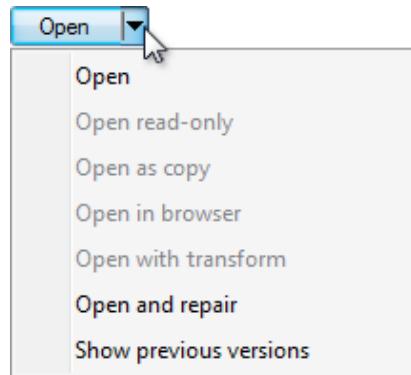
#### 分割ボタン

種類が異なるコマンドをセットにまとめるもので、特にそのうちの1つを頻繁に呼び出す場合に使用します。



折りたたまれた状態の分割ボタン

右端にある下向きの三角形をクリックすると、メニュー ボタンと同様にメニューが現れることを表します。



ドロップダウンされた状態の分割ボタン

この例では、6種類の"開く"コマンドをまとめるために使われています。通常の"開く"コマンドが最も頻繁に使われるため、通常はそれ以外のコマンドが見える必要はありません。分割ボタンを使うと、画面上の占有面積を大幅に抑えつつ、強力な選択肢も提供できます。

メニュー ボタンと違って、ボタンの左側部分をクリックすると、ラベルに示された操作が直ちに実行されます。分割ボタンは、特定のツールを使って実行する次の処理が、直前の処理と同じであるような状況にも向いています。この場合、ラベルは直前の操作に応じて変わります。カラー ピッカーはその典型的な例です。



この例では、直前の操作に応じてラベルが変わっています。

#### 参照ボタン

ダイアログ ボックスを表示して、有効な値を選択しやすくします。

[参照] ボタンで起動されるダイアログ ボックスにより、ユーザーは、ファイル、フォルダー、コンピューター、ユーザー、色などを選択しやすくなります。一般的に、テキスト ボックスなどの制約のないコントロールと組み合わせて使います。通常、[参照]、[その他]、[詳細]などのラベルが付けられ、必ず省略記号を付加して、さらに情報入力が必要である旨を表します。



[参照] ボタン付きのテキスト ボックス。

同じウィンドウ上に [参照] ボタンを多数置く場合、省略形を使うことができます。



省略形の参照ボタン

段階的表示 ボタン  
あまり使われないオプションの表示と非表示を切り替えるために使います。

あまり使われないオプションを非表示にしておき、必要なときに表示することを"段階的表示"と言います。シェブロンは段階的表示があることを示すために使われ、その向きに表示されたり、非表示になったりします。



ボタンをクリックするとラベルが変わり、次にクリックしたときに反対の効果があることが示されます。

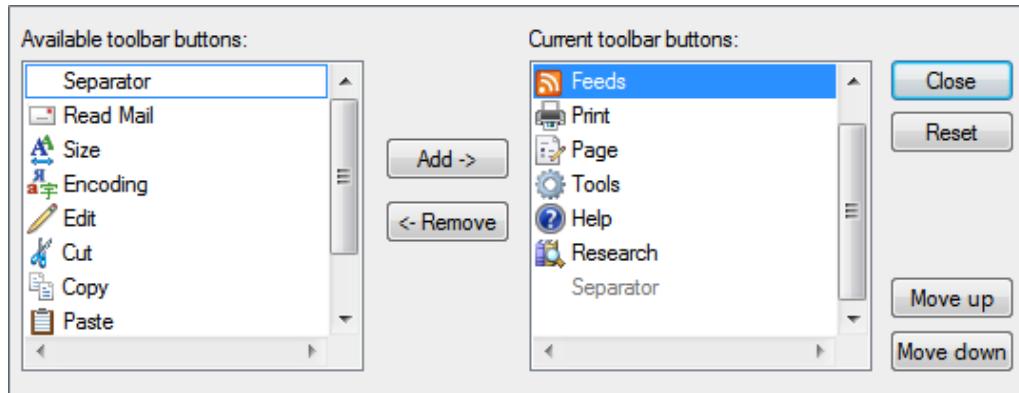


詳しい情報と例については、「[段階的表示コントロール](#)」を参照してください。

方向ボタン  
操作の向きを表すために使いま

この場合、シェブロンではなく単一の山かっこを使って矢印にします。

す。



方向ボタンは操作の向きを示しています。

## ガイドライン

### 全般

- コマンド ボタンをクリックした結果がすぐに表示されない場合は、ビジー ポインターを表示します。何もメッセージが表示されないと、ユーザーはクリックされなかったと見なして、再度クリックする可能性があります。
- いくつかのウィンドウに同じコマンド ボタンを配置する場合、ラベル テキスト、アクセス キー、ウィンドウ上の位置はできるだけ揃えます。
- テキスト ラベル付きのコマンド ボタンのサイズについては、幅は最小にし、高さは標準のコマンド ボタンと同じにします。詳細については、「[推奨されるサイズと間隔](#)」を参照してください。
- 各ウィンドウでコマンド ボタンの幅を揃えます。それが難しい場合でも、幅の種類が 3 とおり以上にはならないようにします。
- 参照ボタン付きテキスト ボックスのように、他のコントロールがコマンド ボタンと連携する場合は、両者の関係がわかりやすいよう、次の 3 つの場所のいずれかにコマンド ボタンを配置します。
  - 他のコントロールの右側に、上辺を揃えて配置。
  - 他のコントロールの下に、左辺を揃えて配置。
  - それぞれの垂直方向の中央を揃えて配置(2 つの連携するリスト ボックス間の [追加]、[削除] ボタンなど)。
- 同じコントロールと連携するコマンド ボタンが複数ある場合は、縦に積み重ね、コントロールの右側に上辺を揃えて配置するか、または横に並べ、その下側に左辺を揃えて配置します。
- コマンド ボタンが他のコントロールに従属する場合は、上記と同様の配置にし、上位コントロールがオフである間は無効にします。
- テキスト ラベル付きの、狭い/短い/高いコマンド ボタンは使用しないでください。洗練されていない印象を与えがちです。既定の幅や高さを使用するようにします。

### 正しい例:

OK

この例では、ボタンは標準的な大きさで、洗練された外観になっています。

### 間違った例:

OK

この例では、ボタンは小さすぎます。

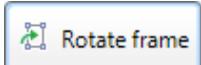
### 間違った例:

OK

この例では、ボタン ラベルの周囲の余白が広すぎます。

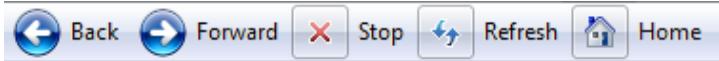
- コマンド ボタンにテキスト ラベルと画像を混在させることは避けます。ほとんどの場合、テキストを画像を組み合わせると不必要に見た目が錯綜し、ユーザーの理解の助けにはなりません。テキストと画像の併用を検討するのには、画像がそのコマンドに対する標準的な記号であったり、ユーザーがコマンドの結果をイメージしやすくなるような場合など、理解の助けになる場合のみです。それ以外の場合はテキストか画像のいずれかを使用しますが、テキストを用いることをお勧めします。

正しい例:



この例では、矢印の画像によってユーザーがコマンドの結果をイメージしやすくなっています。

正しい例:



この例では、標準的なシンボル画像をテキストと組み合わせて表示し、ユーザーが理解しやすいようにしています。

間違った例:



この例では、"取り消し"を表す画像が添えられていますが、テキスト以上の情報が追加されているわけではありません。

- 状態の設定にはコマンド ボタンを使いません。代わりにラジオ ボタンやチェック ボックスを使います。コマンド ボタンは操作を開始するためだけに使います。

## 分割ボタン

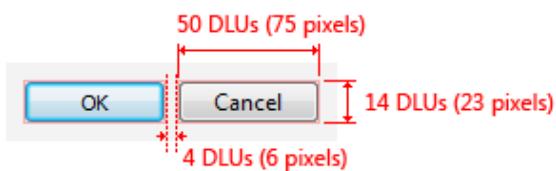
- 最もよく使われるコマンドを既定の動作にします。よく使われるコマンドが複数ある場合は、追加の情報を必要としないものを選択します。
- 最もよく使われるコマンドがユーザーの直前の選択である場合は、ボタン ラベルを直前の選択に変更します。
- 既定のコマンドはメニューに太字で表示します。既定のコマンドが動的に変わる場合や、テキストの代わりに画像が使われている分割ボタンの場合は特に、どれが既定のコマンドであるか、ユーザーが把握しやすくなります。

## 既定値

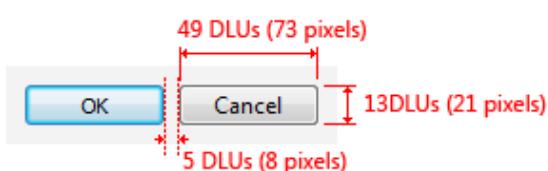
- ダイアログ ボックスには必ず 1 つ、既定のコマンド ボタンを置きます。この際、最も安全(データの消失やシステム アクセスが失われることを防ぐため)で、最もセキュリティの高いコマンドを選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用されるコマンドまたは最も便利なコマンドを選択します。
- リスクのある操作は既定のコマンド ボタンにしません。ただしそのコマンドの処理を容易に元に戻す方法がある場合を除きます。

## 推奨されるサイズと間隔

実際のコントロールのサイズ:



表示されるサイズ:



コントロールの外周に透明な1ピクセルの  
ボーダーがあるため、表示されるサイズは  
コントロールのサイズより小さい

## コマンド ボタンに推奨されるサイズと間隔

## ラベル

- コマンド ボタンにはすべてラベルを付けます。
- ラベルとして画像のみを使用する場合は、適切なテキスト ラベルを名前プロパティに割り当てます。この方法に従うと、スクリーン リーダーなどの支援テクノロジー製品で、画像に関する代替情報を提供できるようになります。



これは画像のみのボタンですが、内部的に "戻る(P)"、"次へ(N)"、"コピー(C)" というラベルが付けられています。

- 省略形の参照ボタン（「...」というラベルが付けられています）の内部ラベルは "参照" にします。
- 一意な [アクセスキー](#)を割り当てます。ガイドラインについては、「[キーボード](#)」を参照してください。

次の場合は例外です。

- [OK]、[キャンセル] ボタンにはアクセスキーを割り当てません。これは、Enter キーが既定のボタン（通常は [OK] ボタン）のアクセスキー、Esc キーが [キャンセル] ボタンのアクセスキーになっていることが理由です。これらのボタンにアクセスキーを割り当てないことで、他のアクセスキーの割り当てが容易になります。
- 省略形のブラウズ ボタン（「...」というラベルが付けられます）にはアクセスキーを割り当てません。これは、一意な割り当てができないためです。
- ラベルは汎用的な内容ではなく、具体的になるようにします。ラベル以外のものを読まなくてもユーザーが意味を理解できることが理想です。一般にユーザーは、静的テキストよりも、コマンド ボタンのラベルを優先的に読む傾向があります。
  - 例外: [キャンセル] ボタンのラベルは、何を取り消すのかがあいまいであっても変更しないでください。ユーザーがすべてのラベルを読まなくても、どのボタンが操作を取り消すものであるかがわかるようにするためです。ただし、保留中の処理が複数ある場合など、どの操作が取り消されるのか明確でない場合は、別のラベルを使用してかまいません。

許容される例:



この例では、[OK] および [キャンセル] ボタンは許容できますが、内容が具体的ではありません。

より良い例:



この例では、[Burn CD] の方が、[OK] よりも具体的です。

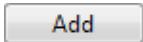
間違った例:



この例では、[Don't burn CD] ではなく [Cancel] を使うべきです。

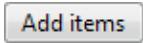
- ラベルは動詞の命令形で始め（英語の場合）、ボタンで実行される操作を明確に記述します。末尾に句読点は付けません。
  - 例外: 次のような標準的なラベルには上記の制限はありません: [詳細設定]、[戻る]、[詳細]、[進む]、[簡易表示]、[詳細表示]、[新規]、[次へ]、[いいえ]、[OK]、[オプション]、[前へ]、[プロパティ]、[設定]、[はい]
- 短いラベルをお勧めしますが、コマンドを十分に説明するものでなければなりません。対象がコンテキストからはっきりわからない場合は、直接目的語（動詞の前に名詞）を使用します。ラベル以外のものを読まなくてもユーザーが意味を理解できることが理想です。

許容される例:



このように短いラベルは、その意味がコンテキストでわかる場合は許容できます。

より良い例: ([追加] では明確でない場合)



この例では、動詞の前に名詞（目的語）を添えて、処理対象を明確にしています。

最も良い例: ([追加] や [アイテムの追加] では明確でない場合)



この例では、ラベルだけで処理内容が明確に言い表されています。

センテンススタイルの大文字化を使用します。これは Windows のトーンに従ったものです。また、コマンドボタンには短い語句を使います。

- 例外: 古いアプリケーションでは、大文字/小文字の形式が混在しないように、[タイトルスタイルの大文字化](#)を採用することもできます。
- コマンドボタンのラベルに "今すぐ" という語を入れないでください。コマンドボタンである以上、即座に実行されることは明らかです。
  - 例外: タスクが即座に実行されるコマンドとそうでないコマンドを区別する必要がある場合には、"今すぐ" という語を使用できます。

[Download](#)

この例では、コマンドボタンをクリックすると、ダウンロード用のウィンドウまたはページに切り替わります。

[Download now](#)

このコマンドボタンをクリックすると、直ちにダウンロードが始まります。

"今すぐ" というラベルを付けることができるるのは、各タスクフローに 1 つのコマンドだけです。たとえば、[今すぐダウンロード] コマンドの後に別の [今すぐダウンロード] コマンドが続かないようにします。

- 当面は処理をしない(後回しにする)という意味のコマンドボタンのラベルに、"後で" という語を使わないでください。たとえば "後でインストール" というラベル("今すぐインストール" と対比)は、このコマンドを実行してしばらくすると自動的にインストールされる場合にのみ使用します。代わりに "インストールしない" または "キャンセル" を使用します。

間違った例:

Do you want to restart your computer now?

[Restart now](#) [Restart later](#)

この例の [後で再起動] は、しばらくすると自動的に再起動される、と誤解されるおそれがあります。

- [詳細設定] ボタンは、詳しい知識のあるユーザー向けか、または高度な知識が必要なオプションにのみ使用します。技術的に高度な機能を呼び出すボタンという意味では使わないでください。たとえばプリンターのホチキス綴じ機能は、ここでいう詳細設定オプションではなく、カラー管理システムは詳細設定オプションです。

間違った例: (上記の意味での詳細設定オプションではない場合)

[Advanced](#)

この例の [詳細設定] というラベルは誤解を与えます。

正しい例:

[More options](#)

このラベルはより意味が具体的で正確です。

- 他のウィンドウを開くコマンドボタンについては、そのウィンドウのタイトルバーに表示される文字列、またはその一部をラベルに入れてください。たとえば、"フォルダーを参照" というダイアログボックスを開くコマンドボタンは、ラベルを "参照" とする、ということです。このように同じ語句を一貫して入れておくと、ユーザーにとって操作の流れがわかりやすくなります。
- 問い合わせに対する応答ボタンは、その内容に合ったラベルにしてください。はい/いいえで答える問い合わせに対して、[OK]/[キャンセル] ボタンは使用しません。

正しい例:

Are you sure you want to send "Fabrikam" to the Recycle Bin?

[Yes](#)

[No](#)

このラベルは問い合わせに対する応答ボタンとして適切です。

#### 間違った例:

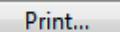
Are you sure you want to send "Fabrikam" to the Recycle Bin?



このラベルは問い合わせに応答するボタンになっていません。

- さらに情報を入力するよう求める場合は、ラベルの末尾に省略記号を付けます。
  - 例外: 画像ラベルには省略記号を付けません。

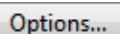
#### 正しい例: ([印刷オプション] ダイアログが表示される場合)



この例のボタンをクリックすると [印刷オプション] ダイアログが現れ、追加情報を入力するよう求められます。

- 他のウィンドウを表示すること自体が目的であるボタンのラベルには、省略記号を付けません。[バージョン情報]、[詳細設定]、[オプション]、[プロパティ]、[ヘルプ]などのコマンドに省略記号を付けることはありません。

#### 間違った例:



このボタンをクリックすると [オプション] ダイアログが表示されますが、ユーザーがさらに情報入力を求められることはできません。

- 省略記号を付けるかどうか判断に迷う場合(ラベルに動詞がないなど)は、ユーザーが何を目的としてこのコマンドを実行しようとするかを基準として判断します。最もありがちな目的が、単にウィンドウに表示される情報を見る事であれば、省略記号は不要です。

#### 正しい例:

他の色...

バージョン情報

1つ目の例では、ユーザーは一般に色の選択を目的としてこのコマンドを実行するので、省略記号を付けています。2つ目の例では、多くの場合ユーザーはバージョン情報を見ようとしてこのコマンドを実行するので、省略記号は不要です。

- 同じウィンドウ上に参照ボタンが複数ある場合は、省略形のボタン("..."というラベルが付けられます)を使用します。参照ボタンを並べて表示する場合は、常に省略形のボタンを使います。
- 方向ボタンについては、单一の山かっこを使って処理の向きを表します。

コミットボタン([OK]、[キャンセル]、[はい]、[いいえ]、[閉じる]、[停止]、[適用]、[完了]、[終了]など)のラベルに関するガイドラインは、「[ユーザーインターフェイスのテキスト](#)」を参照してください。

## ドキュメント

コマンドボタンに言及するときは、以下のこと留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、アクセスキーを示すかっこや下線付き文字、および省略記号は含めません。"ボタン"という語は含めません。
- ユーザー操作を説明する場合は、"クリック"を使用します。
- ラベルは半角の角かっこ([ ])で囲み、可能な場合は太字にします。

例: [印刷] をクリックするとドキュメントが印刷されます。

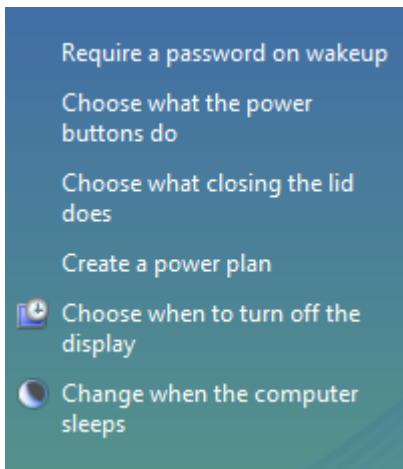
## コマンド ボタンとリンクの比較

従来、**コマンド ボタン**は操作を開始するために使われ、一方で**リンク**は他の場所に移動するために使われていました。ウェブの普及に伴いリンクの用途が広がり、コマンドの開始やオプションの選択にも使用されるようになってきました。どのコマンドにコマンド ボタンを使うか、またはリンクを使うかをここで検討します。アフォーダンスの概念や、主コマンドと副コマンドの違いに着目すると、いくつかのガイドラインが浮かび上がります。

### アフォーダンス

コマンド ボタンは、**アフォーダンス**という点においても優れています。コマンド ボタンは押すことができそうな外観になっているので、ユーザーは使い方がわかります。これに対し、リンクはアフォーダンスの面で劣り、経験に頼ることになります。また、アンダーラインやシステム標準のリンク色がないと、普通のテキストと同じように見えます。リンクとして動作することを確認するには、コンテキストやマウス カーソルをリンクの上に置くほかありません。

このように、アフォーダンスや見つけやすさの点で劣るため積極的に利用しない方がよいように思えますが、ドロップダウン メニューやコンテキスト メニューなど、操作を起動する他のしくみにも同じような問題があることを忘れてはなりません。特定のオブジェクトについて、ユーザーは関連するコマンドをメニュー バーから探したり、コンテキスト メニューから見つけるため右クリックしてみることがあります。しかもメニューの見た目は、使い方を示すようなものではないので、経験から学ぶしかありません。一方で、個々のメニュー項目に使い方を示すアフォーダンスは必要ありません。メニュー全体としてその使い方が示されているからです。したがって、関連するコマンドのメニューが見つかれば、その機能がわかります。



この例では、コマンドのメニューとしてリンクが使われています。メニューのコンテキストでわかるため、コマンド ボタンのようなアフォーダンスが不要です。

### 主コマンドか副コマンドか

主コマンドとは、ウィンドウの主たる目的のために必要なコマンドのことです。たとえば、[印刷] ダイアログ ボックスでは、"印刷" が主コマンドになります。コマンド ボタンを使用すると、主コマンドであることがユーザーにはっきりわかります。副コマンドは、便利ではあるが、そのウィンドウの目的の本質ではない周辺的な操作です。たとえば、[印刷] ダイアログ ボックスでは、"プリンターの検索"、"プリンターのインストール" が副コマンドです。副コマンドには、あまり強調されないようにリンクを使用することを検討します。

### 情報収集か関連ウィンドウの表示か

情報入力を求めたり、タスクをすぐに選択するための、ウィンドウを表示することが目的の場合は、コマンド ボタンを使います。このような操作は、コマンドに似た感じがあります。これに対し、関連はあっても独立するウィンドウを表示するのであれば、リンクを使うことができます。このような操作には、移動のような印象を受けるからです。

したがって、[開く]、[保存]、[参照]などのコマンドについては、副コマンドであっても、リンクではなくコマンド ボタンを使用します。使い分けに迷う場合は、モーダル ウィンドウを表示するのであればコマンド ボタンを、独立したウィンドウであればリンクを使用してください。

## 元の状態に戻せなくなる操作か、いつでも元に戻れる移動か

ユーザーはリンクを見ると、ウェブのページ移動を思い浮かべます。安全性レベルに関してもウェブと同様に考えます。つまり、ブラウザーでは、ユーザーはいつでも [戻る] ボタンを使って元に戻ることができます。さらに、ブラウジングの際、重要な確認のほとんどは、リンクではなくコマンドボタンを使って行います。

このような予測に合わせるため、リスクのある操作や元の状態に戻せない操作など、重要な結果を伴うコマンドにはリンクを使いません。同様に、[ウィザードやタスクフロー](#)では、確認にはコマンドボタンを使用し、リンクは選択や次の手順への移動に使うようにします。

## ラベルの長さ

単純に言うと、コマンドボタンの枠が大きすぎると不自然で目障りになってしまいます。コマンドボタンのラベルが短ければ(通常は4語以下)外観が最もよくなります。画面要素が重ならないように、主コマンドのラベルはできるだけ簡潔にします。

## ガイドライン

- 次の場合には、コマンドボタンを使用します。
  - 主コマンド。
  - 副コマンドであっても、情報収集や選択を行うために使用するウィンドウを表示する場合。
  - リスクのある操作または元の状態に戻せない操作、ウィザードやページフローにおける確認。
- リンクは、他のページやウィンドウ、ヘルプトピックに移動するため、または定義の表示やコマンドメニューに使います。
- 副コマンドはあまり強調されないよう、リンクを使用することを検討します。
- コマンドボタンのラベルは4語あるいは12文字以下の短いものを使用します。リンクはラベルがそれより長くてもかまいません。

## コマンド リンク

適切なコントロールかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

対話操作

提示方法

アイコン

既定値

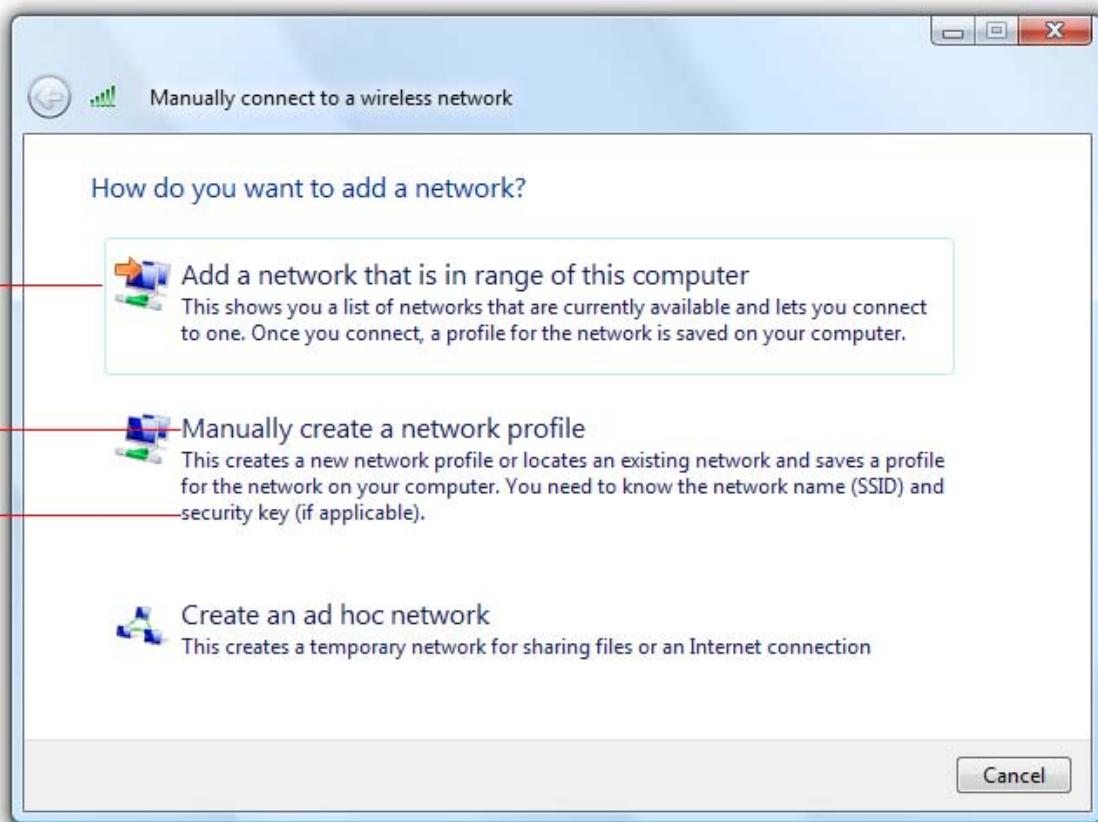
推奨されるサイズと間隔

ラベル

ドキュメント

"コマンド リンク" を使って、ユーザーはメイン指示テキストに対して応答を 1 つ選んで、タスクの次の手順に進みます。

コマンド リンクは外観がすっきりとして見やすいため説明的なラベルを付けることができ、標準的な矢印アイコンまたは独自のアイコン、および補足説明を添えて表示します。



### 典型的なコマンド リンク セット

相互に排他的な選択肢から選ぶ場合に使用するという点で、コマンド リンクは **ラジオ ボタン** に似ています。コマンド リンクは単独で現れることはなく、常にいくつか組にして使用される点もラジオ ボタンと同じです。コマンド リンクの外観は普通の **リンク** と似たすっきりとしたもので、フレームはなく、クリック可能であることを強調する **アフォーダンス** も使われません。また、コマンド リンクは既定の "コマンド ボタン" として使用できる点や、アクセスキーを割り当てるができる点で、**コマンド ボタン**にも似ています。**コミット ボタン** と同様、クリックすると、ウィンドウが閉じられたり(ダイアログ ボックスの場合)、次のページに進んだりします(ウィザードおよびページ フローの場合)。

注: **リンク** および **レイアウト** に関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

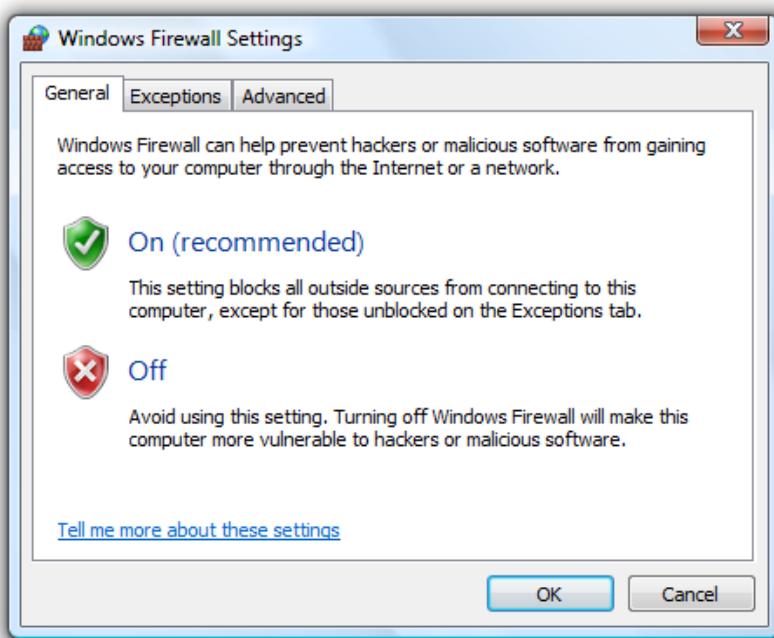
- オプションがメイン指示テキストに対する応答であり、ウィンドウやページの主な目的に関係しているかどうか。別のページに移動するだけでなく、それらに応答してユーザーが何らかの処理を行う必要があるかどうかを検討します。該当しない場合は、コマンド ボタンやリンクなどの他のコントロールを使います。コマンド リンクは、副次的な選択肢や必須でない選択肢、単純なページ移動には適しません。



コントロールパネルの個人設定の各アイテムでは、コマンドリンクが使われているように見えますが、このオプションは通常のリンクです。これは各ページに移動するための[ハブページ](#)であるためです。

- 相互に排他的な応答のセットから1つの応答を選択するために使用されるコントロールであるかどうか。該当しない場合は、別のコントロールを使用します。個別のコマンドを選べるようにするには、コマンドボタンまたはリンクを使います。
- ダイアログボックスの場合、そのコントロールをクリックすることによりウィンドウが閉じられるかどうか。該当しない場合は、ラジオボタン、コマンドボタン、リンクなど、ウィンドウを閉じる必要がないコントロールを使います。

間違った例:



プロパティ ウィンドウやタブ付きダイアログボックスでは、コマンドリンクは使うことができません。クリックするとウィンドウが閉じてしまうからです。

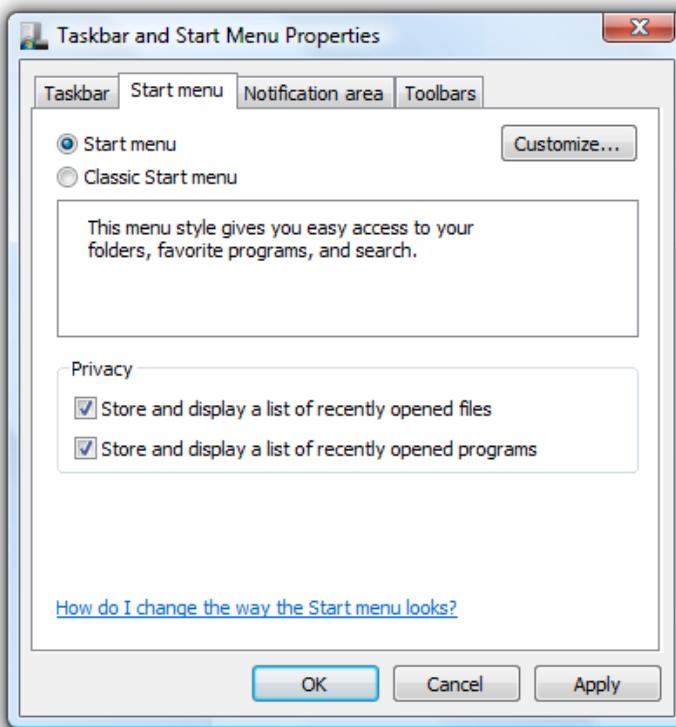
- ウィザードまたはページフローの場合、クリックすることにより未確定のまま次のページへ移動するかどうか。タスクのコミットには、コマンドリンクではなくコミットボタンを使います。コマンドリンクは外観が通常のリンクと似ていてユーザーはページフロー内で移動するためのリンクだと思ってしまいます。したがって、ユーザーがいつでも元に戻れるようにする必要があり、リンクは[コミットページ](#)には向いていません。
- ウィザードまたはページフローの場合、他のページでコマンドリンクを使っているかどうか。これに該当し、他の要素が同じである場合、ページ間で一貫性を保つため、コマンドリンクを使うことをお勧めします。

- 応答の数が2～5個の範囲か。コマンドリンクを単独で使うことはありません。コマンドリンクはサイズが大きいコントロールであり、使用する画面領域がオプションの数に比例するので、応答の数は5項目以下にします。オプションが6項目以上になる場合は、ラジオボタン、通常のリンク、単一選択リストビューを使います。



この例では、Microsoft® Windows® の自動再生機能でリストビューが使われています。

- ラジオボタンとコミットボタンを組み合わせる方が適切かどうか。以下に該当する場合はラジオボタンの方が適しています。
  - ほとんどのユーザーに選択を強く推奨する既定オプションがある場合。ラジオボタンの場合、コマンドリンクに比べ、ユーザーは既定値を変更しようとしない傾向があります。特にウィザードの場合、既定値をそのまま受け入れて[次へ]をクリックすることに慣れています。これに対し、明示的な選択を促す場合は、コマンドリンクの方が向いています。
  - ユーザーが決定前に(追加情報を見るなど)選択肢について何らかのやり取りを行う必要がある場合。たとえば、ラジオボタンを選択すると、そのオプションに関する説明が表示されるようにする場合です。



この例では、ラジオボタンを選択すると、そのオプションの説明が表示されます。

- 同一ページ内に二次的なオプションまたは関連するオプションがある場合。コマンドリンクはページ上で目立つため、他のオプションが見落とされやすくなります。しかも、コマンドリンクをクリックすると、二次的なオプションを選択できなくなります。

間違った例:



この例では、メイン指示テキストに対して 2 つの異なる応答方法があります。二次的なオプションの選択が難しくなってしまうため、最初の応答でコマンド リンクが使われることはないと考えられます。

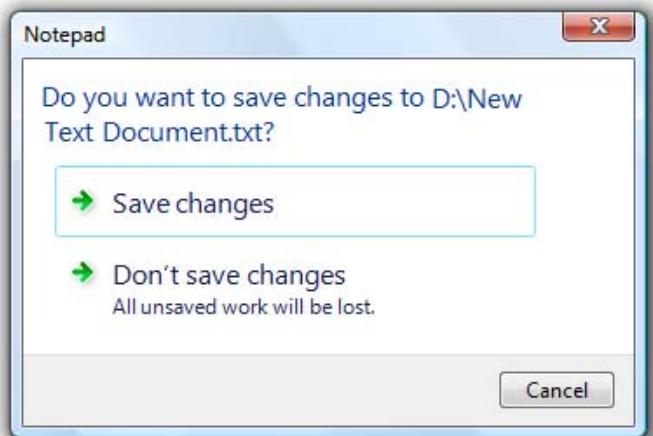
正しい例:



この例では、ユーザーが二次的なオプションを選択できるようになっている一方、ラジオ ボタンによって応答が明確になっています。

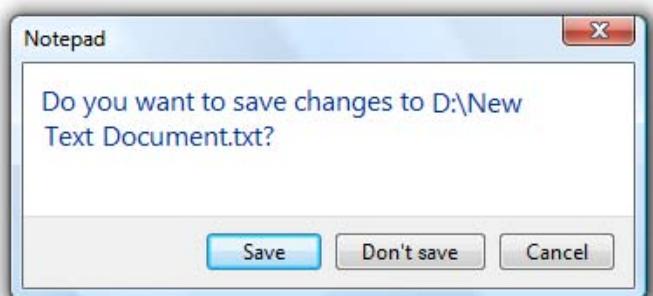
- ダイアログ ボックスの場合、コミット ボタンのグループの方が適切かどうか。コマンド リンクは、それぞれのオプションで、長めの説明的な応答や補足説明が必要な場合に向いています。一方、コミット ボタンのグループは、単純なオプションが数個ある場合に適しています。

間違った例:



この例では、単純なコマンドに対してコマンド リンクを使っているので、ダイアログ ボックスがいたずらに複雑になっています。

正しい例:



この例では、単純なコミット ボタンが的確に使用されています。

一方、コミット ボタンの説明にテキストが使われる場合は、見てすぐにわかるコマンド リンクの方が常に適しています。

間違った例:



この例では、コミット ボタンの説明にテキストが使われています。

正しい例:



この例では、コマンド リンク自体が説明になっています。

注: コマンド リンクが動作するには Windows Vista® 以降が必要であるため、古いバージョンの Windows には適しません。代わりに通常のリンクを使用できます。

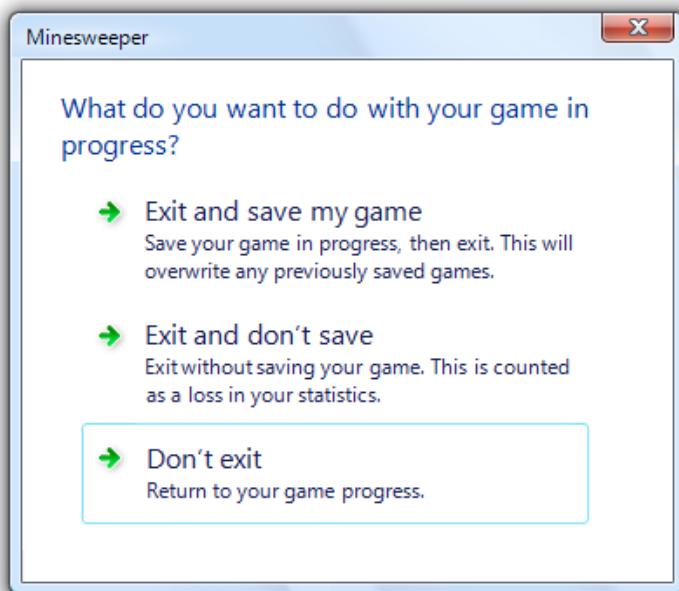


この例では、通常のリンクにアイコンと補足説明を添えて、Windows XP でコマンド リンクの代替として使われています。

## デザイン コンセプト

コマンド リンクでは説明的なラベルおよび任意の補足説明を付けることができますが、だからといって必ずそうしなければならないということではありません。次の例を考えてみます。

間違った例:



このダイアログ ボックスは情報過剰です。

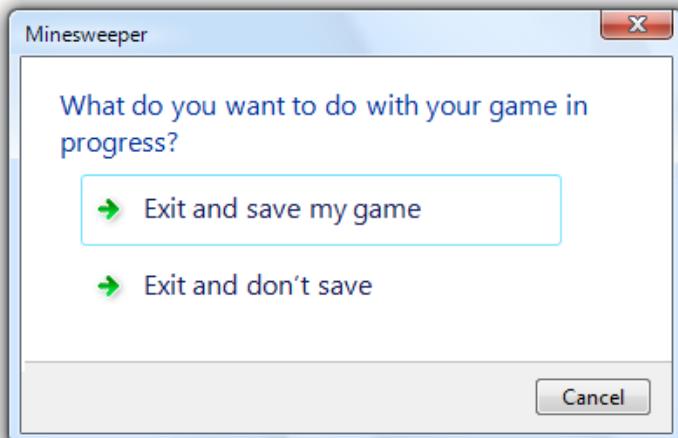
このダイアログで行っているのは単純な質問ですが、コマンド リンクのテキストを使うことによって無駄に複雑になっています。このような単純な質問では、ユーザーはすべてのテキストを読む気になりません。

コマンド リンクに関する以下の 3 つのガイドラインを適用することにより、簡潔なダイアログ ボックスにすることができます。

- コマンド リンクの表現の言い換えになるような補足説明は使用しません。補足説明は、コマンド リンクだけではわかりにくい場合に限って使います。あるコマンド リンクに補足説明を付けたからといって、他のすべてのコマンド リンクに対しても補足説明が必要になるわけではありません。
- この際、最も安全（データの消失やシステム アクセスが失われることを防ぐため）で、最もセキュリティの高い応答を既定で選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用される応答または最も便利な応答を選択します。
- 明示的な [キャンセル] ボタンを用意します。この用途でコマンド リンクを使用しないでください。

以上のガイドラインを適用することで、不必要的補足説明をなくし、最も便利な応答を既定にし、明示的な [キャンセル] ボタンを提供できます。

より良い例：

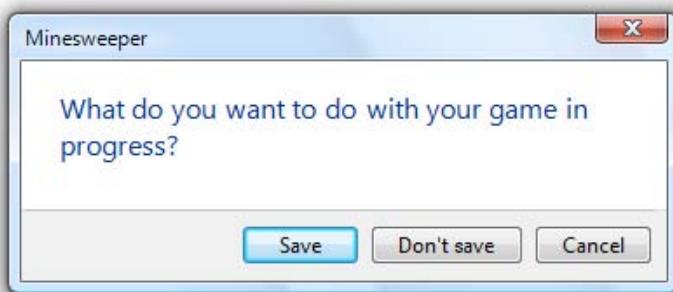


より簡潔なコマンド リンクを使って、改善されたダイアログ ボックス。

この例では、確かに保存しなければデータが失われる場合があることの説明は明記されていませんが、この情報によって判断を変えるユーザーはあまりいないので、このようにするのが良い選択です。

この状況でコマンド リンクが適しているかどうかを分析することで、このダイアログ ボックスはさらに良いものになる可能性があります。長くて説明的な応答は必要ないので、コミット ボタンの方が実際にはより適切です。

最も良い例：



この例では、コミット ボタンが的確に使用されています。

コマンド リンクには多くの長所がありますが、うまく利用しないと情報が過剰になります。ダイアログ ボックスの場合は、まずコミット ボタンを使用することを検討し、コマンド リンクはコミット ボタンではうまくいかない場合に限って使用します。

コマンド リンクは、適切に使えばユーザー インターフェイスが簡潔でわかりやすいものになります。わかりにくいユーザー インターフェイスになってしまった場合は、さかのぼって代替案を検討し、ユーザーに本当に伝える必要がある内容に注目します。

#### 最も重要な点

コマンド リンクを使うことにより情報が過剰にならないようにします。コマンド リンクは簡潔で明確に情報を伝えるためのものであって、複雑にするためのものではありません。

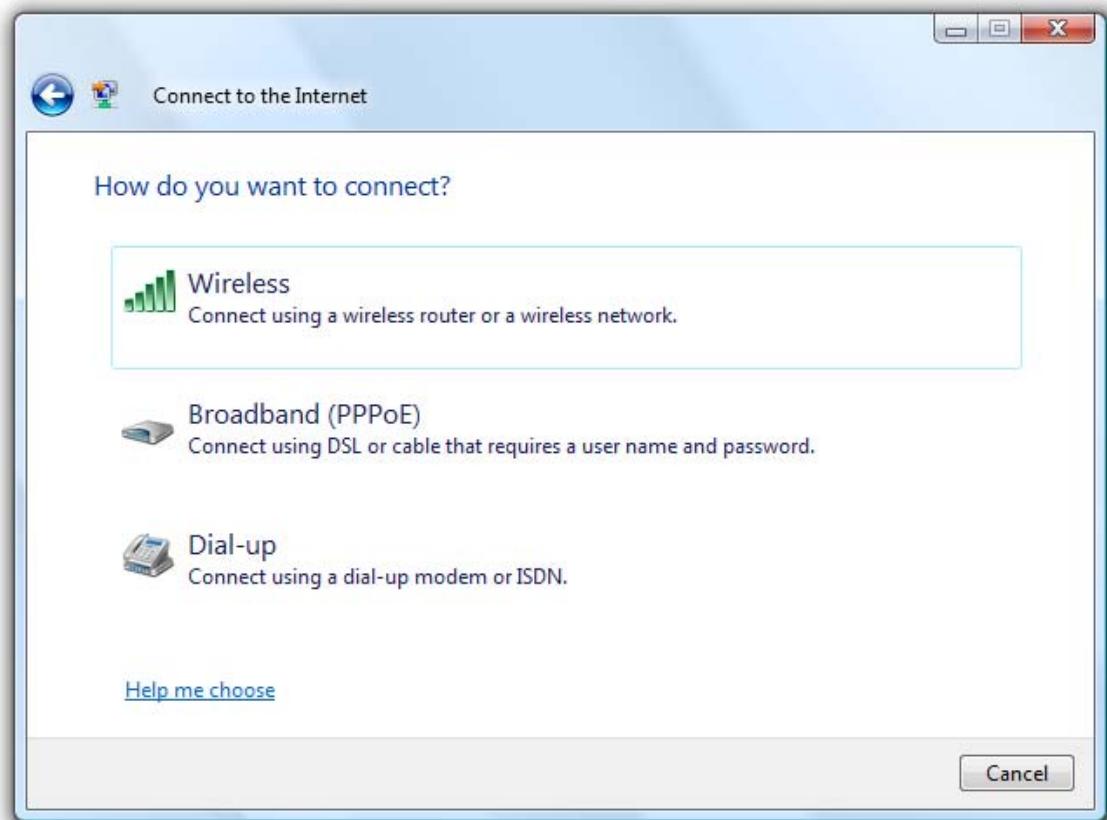
## 使用パターン

コマンド リンクにはいくつかの使用パターンがあります。

**ページ応答** このパターンでは、コマンドリンクは [次へ] ボタンの代わりに使いますが、[キャンセル] ボタンはそのまま使います。

**コマンドリンクは、メイン指示**

**テキストに応答** ページ応答には、コミットの意味はありません。コマンドリンクは通常のリンクと外観が似ていて、して次のページ ユーザーは通常のリンクをページフロー内での移動を連想するため、リンクは、コミットページにはに進むために使 向いていません。ユーザーがいつでも元に戻れるようにしておく必要があります。われます。



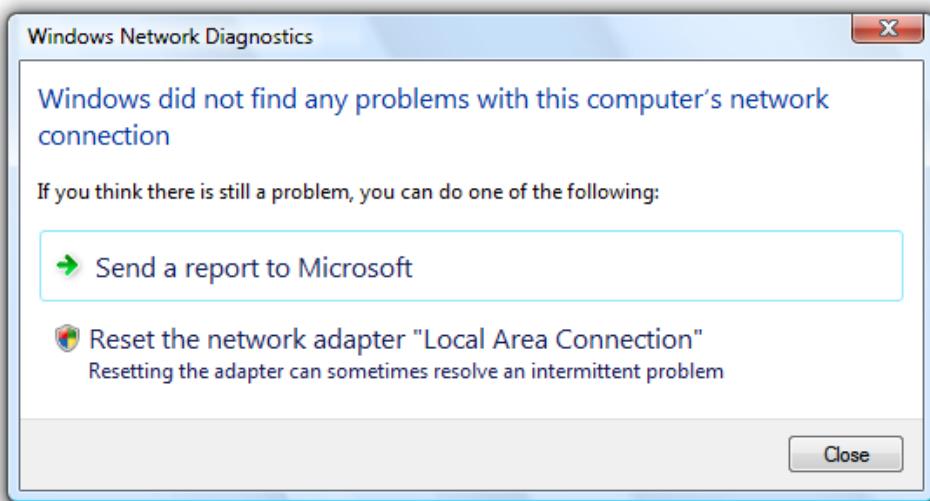
この例では、コマンドリンクがメイン指示テキストに対する説明的な応答を提供するために、使われています。ここではラジオボタンを使用できますが、コマンドリンクを使うと、ユーザーは1回のクリックで応答できます。

**ダイアログボックス応答** このパターンでは、コマンドリンクはコミットボタン ([OK]など) の代わりに使いますが、[キャンセル]ボタンはそのまま使います。

**コマンドリンク**

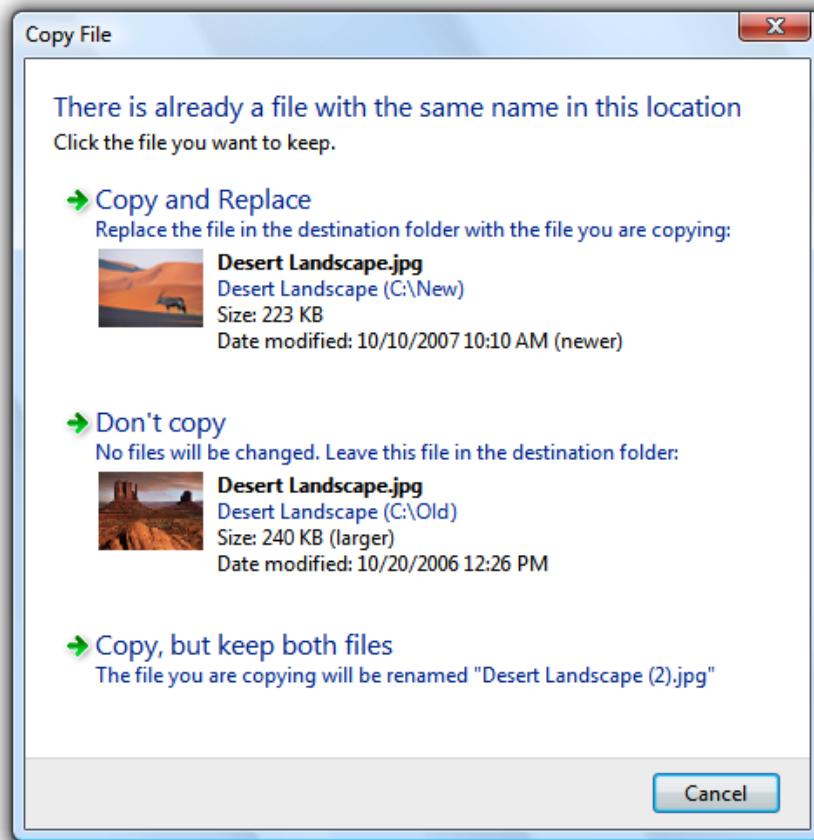
**は、メイン指示** ページフローと違い、ダイアログボックスベースの応答がいったん行われると元に戻る手段があります。したがって、ダイアログボックスのコマンドリンクには、暗黙のうちに"コミット"の意味が含まれています。

**テキストに応答** ダイアログボックスを閉じるために使われます。



この例では、コマンドリンクがメイン指示テキストに対する説明的な応答を提供するために、使われています。ここではラジオボタンを使用できますが、コマンドリンクを使うと、ユーザーは1回のクリックで選択できます。

詳細な応答 ユーザーが応答を選択するために、詳細な情報を必要とする場合があります。  
詳細な情報が含まれるページ応答やダイアログ応答です。



この例では、ユーザーが情報を踏まえて判断できるよう、詳細なコマンドリンクが使われています。サムネイルとファイルの詳細情報が、ユーザーの判断に役立ちます。

## ガイドライン

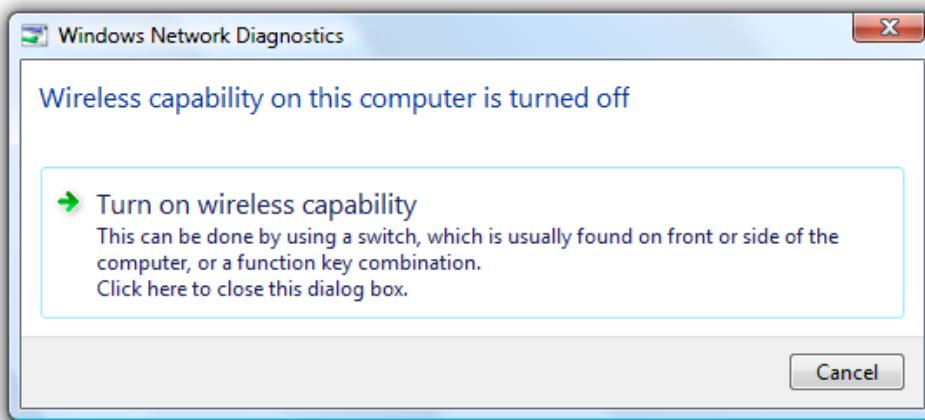
### 対話操作

- コマンドリンクをクリックした結果がすぐに表示されない場合は、ビジー ポインターを表示します。何もメッセージが表示されないと、ユーザーはクリックされなかったと見なして、再度クリックする可能性があります。

### 提示方法

- コマンドリンクは常に 2 つ以上のセットで提示します。応答が 1 種類しかない質問は論理的に意味をなしません。

間違った例:

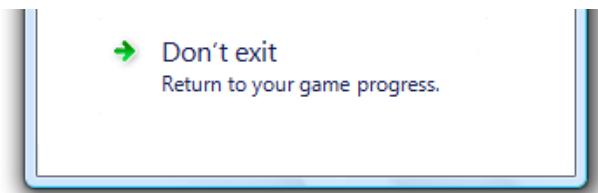


この例では、ユーザーに選択肢を示すためにダイアログ ボックスが表示されますが、単なる指示に過ぎません。このような場合は、代わりに情報ダイアログを使用します。

- 最も一般的に使われるコマンドリンクを先頭に置きます。使用される頻度順に主に従って並べますが、論理的にも自然になるようにします。
  - 例外: "すべて実行する" 旨のコマンドリンクは先頭に置きます。

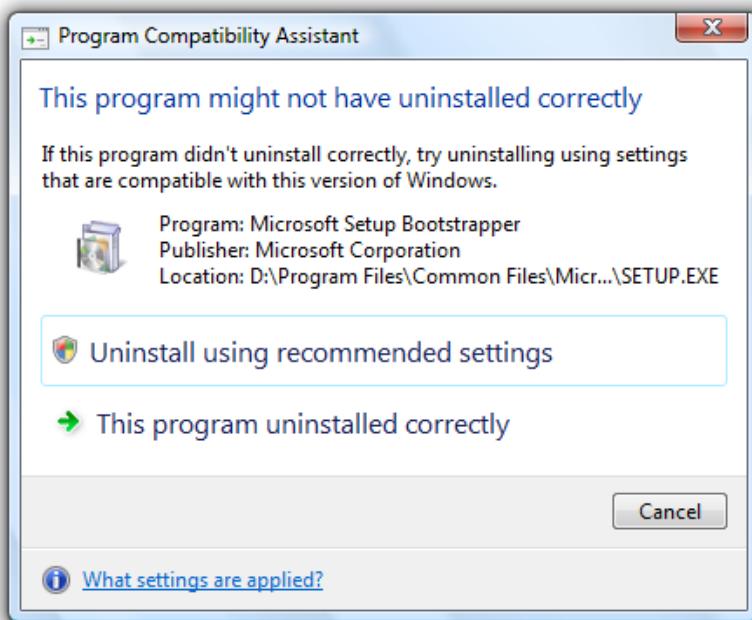
- 明示的な [キャンセル] ボタンを用意します。この用途でコマンドリンクを使用しないでください。タスクを実行する必要がないことにユーザーが気付くことはかなり頻繁にあります。コマンドリンクを取り消しに使用すると、どのコマンドリンクが取り消しを意味するか判断するために、すべてのコマンドリンクをよく読む必要があります。[キャンセル] ボタンがあれば、ユーザーは効率よくタスクを取り消すことができます。

間違った例:



この例の [終了しない] コマンドリンクは、[キャンセル] ボタンにする必要があります。

- 明示的な [キャンセル] ボタンを用意するとコマンドリンクが 1 つだけになってしまう場合は、取り消すためのコマンドリンクと [キャンセル] ボタンの両方を用意します。これにより、選択肢があることがはっきりとユーザーに伝わります。このコマンドリンクでは、単に "キャンセル" またはそれに類する語句ではなく、もう一方の選択肢との対比が明らかな語句を使用します。



この例では、2 つ目のコマンドリンクで選択肢があることが示されていますが、これによって実行されることは取り消しだけですが、1 つ目のコマンドリンクとの違いがわかるような語を使って記述されています。

- 環境を (副次的な影響を残さず) 元の状態に戻せない場合は、[キャンセル] ではなく [閉じる] を使います。
- 無効になっているコマンドリンクは表示しません。現在のコンテキストに該当しないコマンドリンクは、無効にするではなく削除します。該当しないコマンドリンクをすべて削除すると、残るコマンドリンクが 1 つだけになる場合は、そのウィンドウまたはページを削除するか、明示的なユーザーの同意が必要であれば確認画面を表示します。

## アイコン

- すべてのコマンドリンクにアイコンが必要です。アイコンにより、通常のリンクやユーザーインターフェイス テキストとコマンドリンクを区別しやすくなります。
- 矢印アイコンはコマンドリンクのみに使用します。Windows XP でコマンドリンクの代わりとして使用される場合を除いて、通常のリンクでは矢印アイコンは使用しません。
- 権限昇格がすぐに必要な応答であることを示す場合はセキュリティ シールド アイコンを使用します。セキュリティ シールド アイコンの使用に関する追加のガイドラインについては、「[ユーザー アカウント 制御](#)」を参照してください。
- カスタム アイコンは、ユーザーがオプションを視覚的に判別したり区別するのに役立つ場合に限って使います。すぐに判別したり意味がわかるカスタム アイコンでなければ使わないでください。

間違った例:



**Invite someone you trust to help you**  
If you've used Remote Assistance before, you can use a previous invitation to get help from the same person.



**Offer to help someone**  
Offer Remote Assistance to someone or open a Remote Assistance invitation.

この例では、カスタム アイコンからすぐに判別できません。

- カスタム アイコンを使用する場合は、 $16 \times 16$  ピクセルまたは  $32 \times 32$  ピクセルのサイズにします。画面領域が十分にあり、サ イズを大きくする方が視覚的に有利な場合に、サイズが大きいアイコンを使います。セキュリティ シールドのオーバーレイ表示を使用する必要がある場合は、 $32 \times 32$  ピクセルまたは  $48 \times 48$  ピクセルのアイコンにします。



**Wireless**

Connect using a wireless router or a wireless network.



**Broadband (PPPoE)**

Connect using DSL or cable that requires a user name and password.



**Dial-up**

Connect using a dial-up modem or ISDN.

この例では  $32 \times 32$  ピクセルのカスタム アイコンを使っています。



**Home**

Choose this for a home or similar location. Your computer is discoverable and you can see other computers and devices.



**Work**

Choose this for a workplace or similar location. Your computer is discoverable and you can see other computers and devices.

この例では、セキュリティ シールドのオーバーレイ表示付きで  $48 \times 48$  ピクセルのアイコンが表示されています。

- ウィンドウまたはページ上では、カスタム アイコンと標準の矢印アイコンの混在を避けます。カスタム アイコンを使う場合は、すべてカスタム アイコンを使用するようにします。ただし、無意味にカスタム アイコンを使うより、標準の矢印アイコンの方が適切です。

## 既定値

- この際、最も安全 (データの消失やシステム アクセスが失われることを防ぐため) で、最もセキュリティの高い応答を既定で選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用される応答または最も便利な応答を選択します。
- 実用的な場合は、1つ目の応答を既定のオプションにします。順序が論理的である限り、ユーザーは、通常そのように想定します。
- ダイアログ ボックスの場合、リスクのある操作を既定のコマンド リンクにしません。ただし操作を容易に元に戻す方法がある場合を除きます。

## 推奨されるサイズと間隔

If you think there is still a problem, you can do one of the following:

9 DLUs (15 pixels)	
25 DLUs (41 pixels)	Send a report to Microsoft
35 DLUs (58 pixels)	Reset the network adapter "Local Area Connection" Resetting the adapter can sometimes resolve an intermittent problem

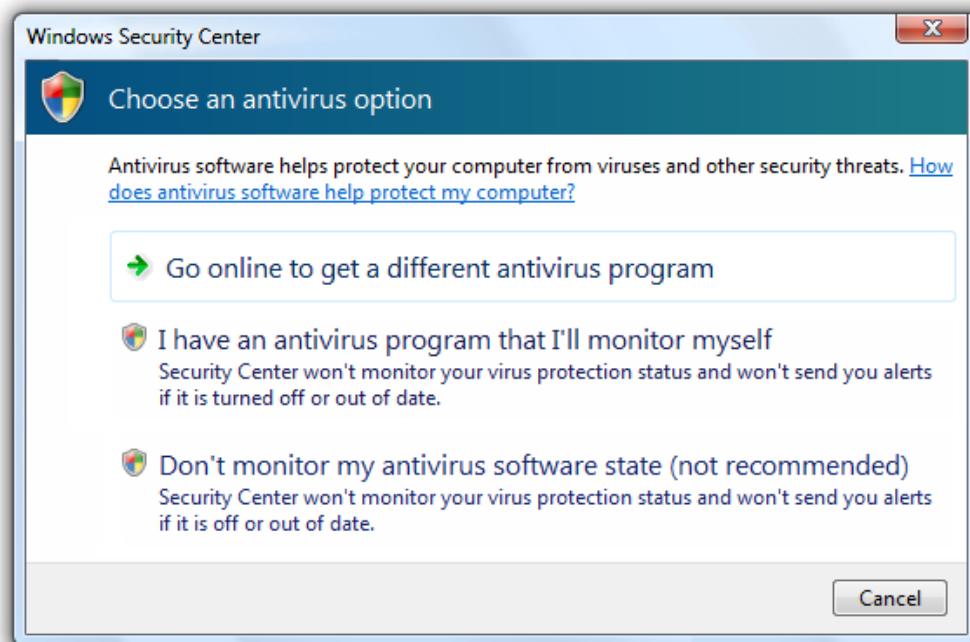
## ラベル

注: コマンド リンクはメイン指示テキストに応答するためのものなので、まず適切なメイン指示テキストを作成してから、それに対する応答を決定してください。

## コマンド リンクのラベル

- コマンド リンクの処理の内容と他のコマンド リンクとの違いがはっきりわかる、簡潔なラベルにします。これだけで内容が理解できる、メイン指示テキストに対応した表現である必要があります。他の応答との違いが明確なラベルであることも重要です。コマンド リンクの本来の意味や、他のコマンド リンクとの違いを、ユーザーが考えこむようなものにはしないでください。

間違った例:



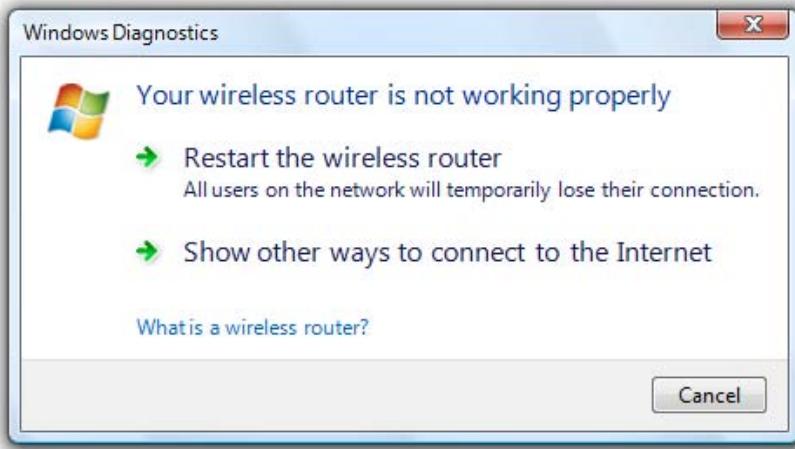
この例では、2番目と3番目の応答の違いがわかりません。また、[キャンセル] ボタンも適切ではありません。

- コマンド リンクのラベルは、ユーザーの正しい判断を助ける内容にします。選択に影響のない詳細は省きます。何が起きるかの完全な説明をラベルに記述する必要はありません。
- コマンド リンクのラベルには動作を示す語句を含めます。ただし、"クリックする" という動詞は使いません。コマンド リンクの処理内容を記述する場所であって、どのように操作するかを記述する場所ではありません。
  - 例外: コマンド リンクがすべて同じ語句で始まる場合は、冗長な語句を省略します。
- 全般的に、肯定的な言い回しを使います(何かを実行する選択肢を提示します)。否定的な言い回し(何かを実行しない選択肢を提示する)も、その方がラベルがわかりやすくなる場合は使ってかまいません。
- ラベルには同じ文法構造の表現を使用し、1行で記述します。長いラベルは読む気が損なわれるものであり、また必要でもありません。適度な長さのラベルは、ドキュメントで言及するのも容易です。
- センテンススタイルの大文字化を使用します。
- ラベルが質問である場合を除いて、句読点は使用しません。
- 一意なアクセスキーを割り当てます。ガイドラインについては、「キーボード」を参照してください。
- 省略記号は使いません。省略記号は、操作の実行にさらに情報が必要になる可能性を示します。コマンド リンクは直ちに反映されるので、コマンド リンクが適切に使われていれば省略記号は必要ありません。
- 強く推奨される応答に対しては、ラベルに"(推奨)" を追加します。補足説明に対してではなく、ラベルに追加するようにします。
- 詳しい知識のあるユーザー向けの応答に対しては、ラベルに"(詳細設定)" を追加することを検討します。補足説明に対してではなく、ラベルに追加するようにします。

ヒント: 友人がメイン指示テキストを読み上げて自分がコマンド リンクで応答する場面を想像すれば、コマンド リンクが適切かどうか評価できます。コマンド リンクによる応答が不自然であったり使いにくかったりする場合は、コマンド リンク、場合によってはメイン指示テキストを変更します。

## 補足説明

- コマンド リンクにさらに詳しい説明が必要な場合は、補足説明を記述します。補足説明には、その応答をどのような場合に選択する必要があるか、選択すると何が起こるかを記載します。



この例では、オプションの影響が補足説明で示されています。

- コマンドリンクの表現の言い換えになるような補足説明は使用しません。補足説明は、コマンドリンクだけではわかりにくい場合に限って使います。あるコマンドリンクに補足説明を付けたからと言って、他のすべてのコマンドリンクに対しても補足説明が必要になるわけではありません。
- 補足説明は、ユーザーの正しい判断を助ける内容にします。選択に影響のない詳細は省きます。何が起きるかの完全な説明を補足説明に記述する必要はありません。
- 同じ文法構造の表現を使用し、多くても3行までにします。長い補足説明は読む気が損なわれるものであり、また必要でもありません。
- 文を使用し、末尾に句点を付けます。

#### コマンドリンク グループのラベル

- グループラベルは使いません。メイン指示テキスト自体がコマンドリンクのグループラベルとしての機能を果たしています。

#### ドキュメント

コマンドリンクに言及するときは、以下のことに留意します。

- ラベルのテキストは正確に記述しますが、アクセスキーを示すかっこや下線付き文字は含めません。
- ラベルにオブジェクト名が含まれている場合は、オブジェクト名は削除するかプレースホルダー テキストを使います。
- ユーザー操作を説明するには、"クリック"を使用します。
- ラベルは半角の角かっこ([ ])で囲み、可能な場合は太字にします。

例:

図をコピーするには、[コピーして置き換える]をクリックします。

[ネットワークアダプターをリセットする]をクリックします(コマンドリンクのラベルが"ネットワークアダプター<アダプタ名>をリセットする"となっている場合)。

## ドロップダウンリストとコンボ ボックス

適切なコントロールかどうかの判断基準

使用パターン

ガイドライン

全般

提示方法

ドロップダウンリスト

プレビュードロップダウンリスト

コンボ ボックス

既定値

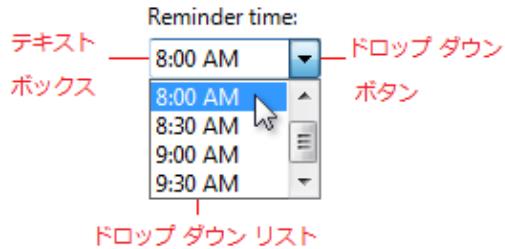
プロンプト

推奨されるサイズと間隔

ラベル

ドキュメント

"ドロップダウンリスト" または "コンボ ボックス" を使用すると、ユーザーは相互に排他的な値の一覧の中から選択できます。ユーザーが選択できるオプションは、いずれか 1 つのみです。標準的なドロップダウンリストの場合、選択肢は一覧の中にあるものに限られます。これに対してコンボ ボックスの場合は、一覧にない選択肢をユーザーが入力して選ぶことができます。



### 典型的なコンボ ボックス

この項目を理解するために重要な単語を以下に示します。

- "リスト ボックス" は、複数のアイテムの一覧を含むボックスで、複数のアイテムが見える状態になっています。
- "ドロップダウンリスト" は、選択されているアイテムが常に見えるようになっていて、選択されていない項目は必要に応じて、ドロップダウンボタンをクリックして見ることができます。
- "コンボ ボックス" は、リストボックスまたはドロップダウンリストと、編集可能なテキスト ボックスを組み合わせたものであるため、ユーザーは一覧にない値を入力できます。
  - "編集可能なドロップダウンリスト" は、ドロップダウンリストと編集可能なテキスト ボックスを組み合わせたものです。
  - "編集可能なリスト ボックス" は、リスト ボックスと編集可能なテキスト ボックスを組み合わせたものです。

注: レイアウトに関するガイドラインは、別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- 相互に排他的な値の一覧から 1 つのオプションを選択するためのコントロールであるか。該当しない場合は、別のコントロールを使用します。複数のオプションを選択する場合は、代わりに標準の複数選択リスト、チェック ボックスリスト、リストビルダー、または追加/削除リストを使用します。
- オプションがコマンドであるかどうか。該当する場合は、代わりにメニュー ボタンまたは分割ボタンを使用します。ドロップダウンリストとコンボ ボックスは、オブジェクト(名詞)や属性(形容詞)に使用し、コマンド(動詞)には使用しません。
- 一覧に表示されるものが、プログラム オプションではないデータかどうか。オプションでもデータでも、ドロップダウンリストまたはコンボ ボックスは適切な選択です。これに対して、ラジオ ボタンはプログラム オプションの数が少ない場合にのみ適しています。

### ドロップダウン リスト

- ほとんどの状況下で大多数のユーザーに推奨される既定のオプションがあるかどうか。選択されているオプションが見えることが、他のオプションが見えることよりもはるかに重要かどうかを判断します。ユーザーにオプ

ションの変更を勧めない場合は、ドロップダウンリストを使用して他のオプションを隠すことを検討します。該当しない場合は、他の選択肢がもっと目立つよう、ラジオ ボタン、単一選択リスト、または編集可能なリストボックスの使用を検討します。



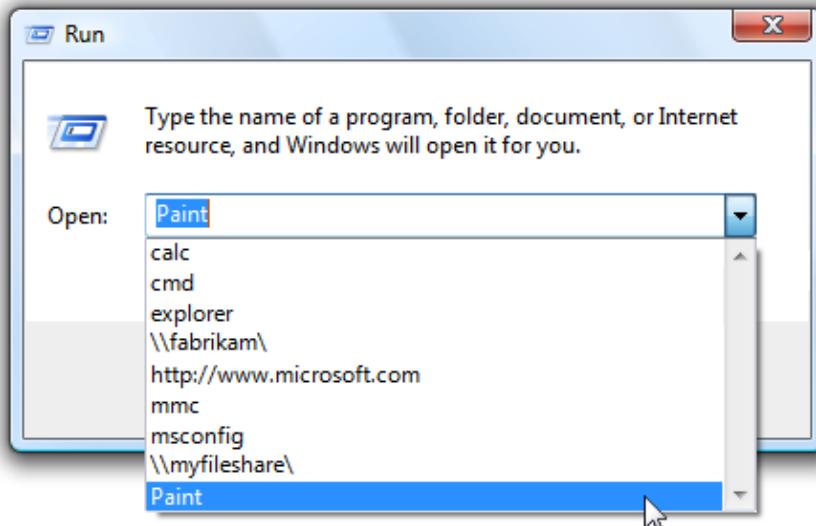
この例では、画面の色質を最高に設定することがほとんどのユーザーにとって最適な選択となるので、ドロップダウンリストが他のコントロールより適しています。

- オプションに注意を引く必要があるかどうか。該当する場合は、ラジオ ボタン、単一選択リスト、または編集可能なリストボックスの使用を検討します。これらのコントロールは占める画面領域が広いので、注意を引きやすくなります。ドロップダウンリストは小さいので、あまり目立たせる必要のないオプションに向いています。
- 画面領域が貴重か。該当する場合は、必要な画面領域が一定で選択肢の数に依存しない、ドロップダウンリストを使用します。
- 同じウィンドウ上に他のドロップダウンリストがあるか。該当する場合は、一貫性を保つためにドロップダウンリストの使用を検討します。

#### 編集可能なドロップダウンリスト

上記のドロップダウンリストに対する原則のほか、以下の事項も含まれます。

- 選択肢に制約があるかどうか。該当する場合は、代わりに通常のドロップダウンリストを使用します。コンボボックスでは入力に制約がなく、一覧にない値でも入力できます。入力の制約がないので、有効ではないテキストをユーザーが入力した場合に、エラー メッセージを使ってエラー処理を行う必要があります。
- 選択される可能性が高い選択肢をあらかじめ列挙できるかどうか。該当しない場合は、代わりにテキスト ボックスを使用します。
- ドロップダウンリストが、以前のユーザー入力を一覧表示するために使われるかどうか。ユーザーが以前の入力全部の一覧を見直す必要がなければ、代わりにオートコンプリート オプション付きのテキスト ボックスを使用します。



この例では、以前の入力をユーザーが見直すことがあるので、編集可能なドロップダウンリストが適切です。



この例では、オートコンプリート付きのテキスト ボックスが適切です。

- ユーザーが有効な値を選択するために、補助が必要かどうか。該当する場合は、参照ボタン付きのテキスト ボックスを代わりに使用します。



この例では、ユーザーは [宛先] をクリックして、有効な値を選択しやすくなっています。

- 他の選択肢を確認することや、変更を促すことが重要であるかどうか。該当する場合は、代わりに編集可能リストボックスの使用を検討します。編集可能なドロップダウンリストの場合、リストを展開するまでユーザーは他の選択肢があることがわかりません。
- ユーザーは多くのアイテムが含まれる一覧から 1 つのアイテムをすばやく見つける必要があるかどうか。(Win32 のみ) 該当する場合は、コンボボックスを使用し、アイテムの名前全体を入力してアイテムを選択できるようにします。これに対して、Win32 ドロップダウンリストでは、最後に入力された文字だけを基に項目が選択されます(たとえば、月のリストに「12」と入力すると [12 月] ではなく [2 月] が選択されます)。この場合は、選択肢に制約がある場合でもコンボボックスを使用します。

### 編集可能なリストボックス

- 選択肢に制約があるかどうか。該当する場合は、代わりに単一選択リストまたは通常のドロップダウンリストを使用します。コンボボックスでは入力に制約がなく、一覧にない値でも入力できます。入力の制約がないので、有効ではないテキストをユーザーが入力した場合に、エラーメッセージを使ってエラー処理を行う必要があります。
- 選択される可能性が高い選択肢をあらかじめ列挙できるかどうか。該当しない場合は、代わりにテキストボックスを使用します。
- 他の選択肢を確認することや、変更を促すことが重要であるかどうか。該当しない場合は、代わりに編集可能なドロップダウンリストを使用することを検討します。
- オプションに注意を引く必要があるかどうか。該当しない場合は、代わりに編集可能なドロップダウンリストを使用することを検討します。ドロップダウンリストは小さいので、あまり目立たせる必要のないオプションに向いています。
- 画面領域が貴重か。該当する場合は、必要な画面領域が一定で選択肢の数に依存しない、編集可能なドロップダウンリストを使用します。

ドロップダウンリストは、数千アイテムから 1 アイテムまで対応できるので、一覧のアイテム数はコントロールを選択する際の判断要素にはなりません。編集可能なドロップダウンリストでは、ユーザーが一覧にない値を入力できるので、数千アイテムからゼロアイテムまで対応できます。ドロップダウンリストはデータに使用できるので、アイテム数が事前にわからなかったり、確実でない可能性があります。追加の画面領域を揃えるため、編集可能なリストボックスには常に、少なくともアイテムを 3 つ含めます。

### 使用パターン

ドロップダウンリストとコンボボックスにはいくつかの使用パターンがあります。

ドロップダウンリスト 閉じられた状態では、選択されているアイテムのみが表示されます。ドロップダウンボタンをクリックすると、すべてのオプションが表示されます。値を変更するには、一覧を開いて別の値をクリックします。

値の固定セット

を持つ標準的な

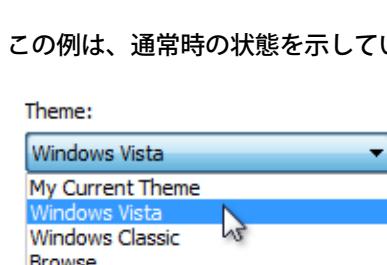
ドロップダウン

リストです。

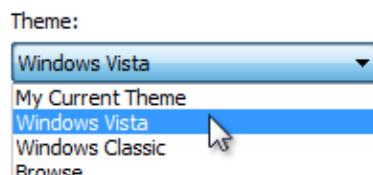
Theme:

Windows Vista

▼



この例は、通常時の状態を示しています。



この例は、展開された状態を示しています。

プレビュード

ロップダウンリ

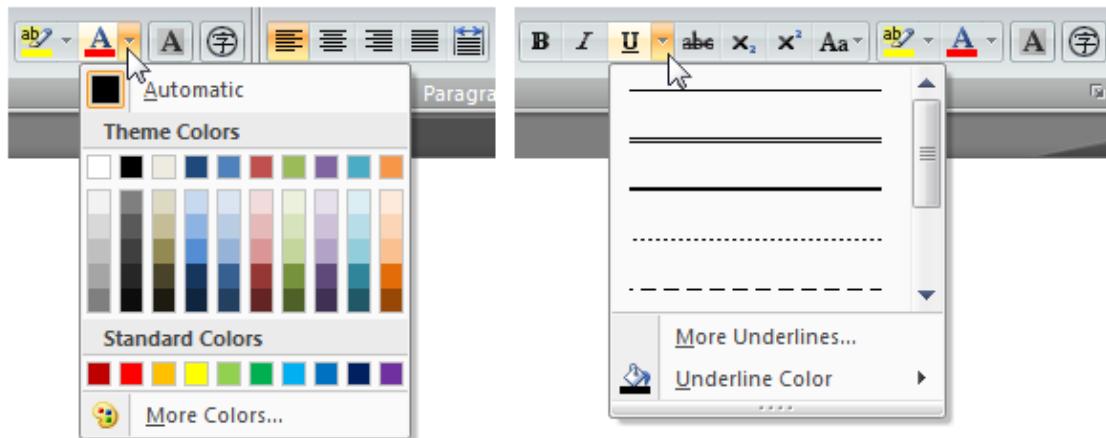
スト

選択しやすいよ

う、アイテムを

プレビュー表示

するドロップダウントリストです。



これらの例では、ドロップダウンリストにアイテムがプレビュー表示されています。

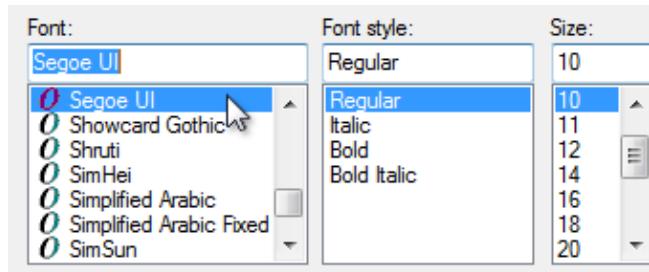
編集可能なドロップダウンリスト  
リスト  
ドロップダウンリストにない値をユーザーが入力できるドロップダウンコンボボックスです。



編集モードおよび展開モードにある編集可能なドロップダウンリストの例です。

テキストボックスの柔軟性を提供しながらも、選択される可能性が高い選択肢を便利な一覧で提供してユーザーを補助する場合に、このコントロールを使用します。

編集可能なリストボックス  
常に表示されている一覧にない値をユーザーが入力できる通常のコンボボックスです。



これらの例では、編集可能なリストボックスが常に表示されています。

他の選択肢をユーザーが確認することや、変更を促すことが重要な場合は、編集可能なドロップダウンリストよりもこのコントロールの方が適しています。

## ガイドライン

### 全般

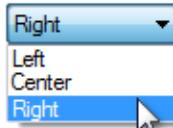
- ドロップダウンリストやコンボボックスでのアイテムの変更は、以下の動作に結び付けません。
  - コマンドの実行。
  - 追加の情報入力を行うためのダイアログボックスなど、他のウィンドウの表示。
  - 選択されているコントロールに関する、他のコントロールの動的な表示(このようなイベントは、[スクリーンリーダー](#)で検出できません)。

### 提示方法

- アイテムを論理的な順序で一覧にします。関連性が高いオプションをまとめてグループにする、最も一般的なオプションを先頭にする、またはアルファベット順にするなどの方法をとります。名前はアルファベットまたは五十音順、数字は番号順、日付は時系列に基づいて並べ替えます。12項目以上ある一覧は、項目を見つけやすいようにアルファベット順または五十音順に並べる必要があります。

### 正しい例:

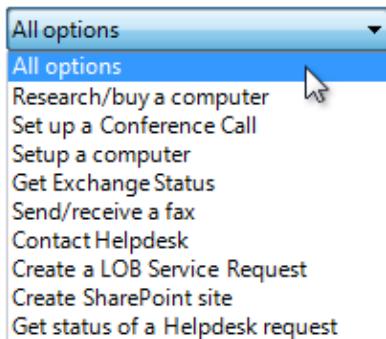
Alignment:



この例では、リストアイテムが位置関係に基づいて並べられています。

間違った例:

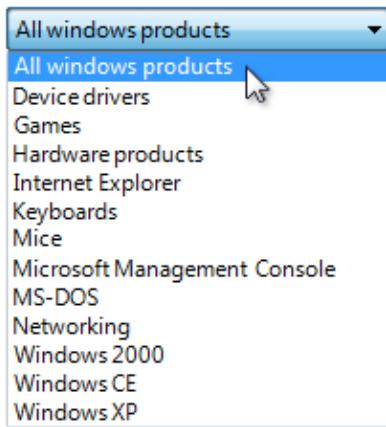
Select from list:



この例では、リストアイテム数が多すぎるので、アルファベット順に並べる必要があります。

正しい例:

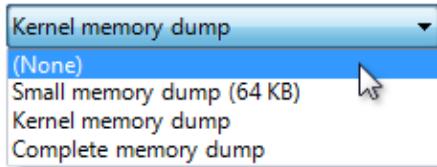
Select a product or topic:



この例では、すべてのアイテムを示すオプション以外のアイテムが、アルファベット順に並べられています。

- 他のアイテムの並び順に関係なく、一覧の先頭に "すべて" または "なし" を示すオプションを配置します。
- メタオプションはかっこで囲みます。

Write debugging information:



この例では、[(なし)]がメタオプションです。選択肢として有効な値ではなく、オプション自体が使用されていないことを示します。

- ドロップダウンリストまたはコンボ ボックスを無効にするときは、関連するラベルやコマンド ボタンも無効になります。

## ドロップダウン リスト

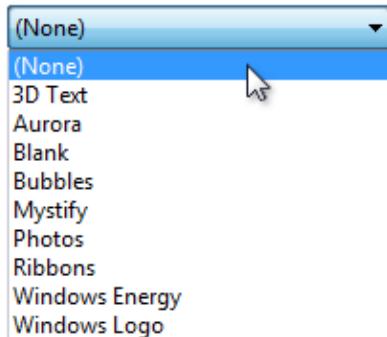
- 関連付けた別のコントロールの表示を変更するために、単一のドロップダウンリストを使用する場合は、別のコマンド ボタンをクリックしなくても、選択の時点で直ちに表示が変更されるようにします。表示の変更に長時間かかる場合に限り、別のコマンド ボタンを使用します。ただし、このような目的では、リストヘッダーおよび

ニュー ボタン コントロールの使用をお勧めします。

- 一覧には空白のアイテムは使用しないで、代わりにメタオプションを使用します。メタオプションの意味は明示的であるのに対して、空白のアイテムはどのように解釈するのかユーザーにはわかりません。

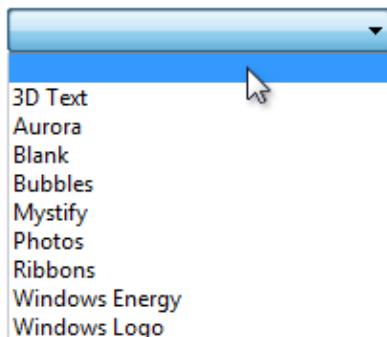
正しい例:

Screen saver:



間違った例:

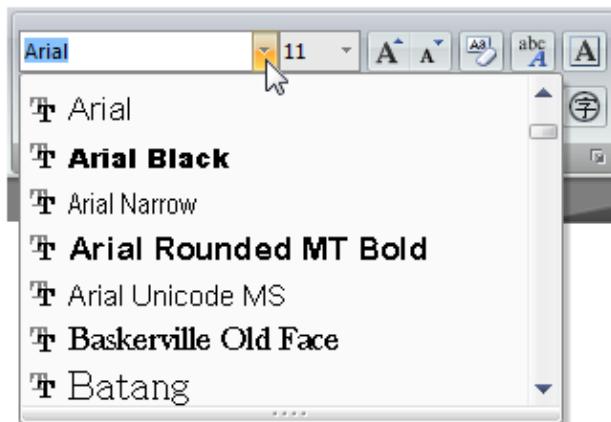
Screen saver:



この間違った例では、空白のオプションの意味が不明確です。

プレビュー ドロップダウンリスト

- テキストだけを使用して説明するよりもイメージと併せて示す方が適切な場合は、リストアイテムのプレビューを使用します。



この例では、テキストだけで示すよりもプレビューを使う方がオプションの説明としてはるかに適切です。

- 不必要的参考にならないアイコンはプレビューで使用しません。

間違った例:



この例では、プレビュー アイコンが何も情報を伝えていないので必要です。

## コンボ ボックス

- 可能な場合は、入力テキストの長さを制限します。たとえば、有効な入力値が 0 から 999 までの数値の場合は、入力を 3 文字に制限するコンボ ボックスを使用します。
- 選択可能なオプションが多数ある場合は、最も可能性が高いオプションに絞り込んで一覧に表示します。コンボ ボックスでは一覧にない値を入力できるので、すべての選択肢を一覧表示する必要はなく、可能性が高い選択肢や代表的な例を一覧表示するだけでかまいません。



この例では、フォント サイズ 15 や中間サイズのフォント 9.5 など、多くの有効な選択肢が一覧表示されていません。

## 既定値

- 最も安全(データの消失やシステム アクセスが失われることを防ぐため)で、最もセキュリティの高いオプションを既定で選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用されるオプションまたは最も便利なオプションを選択します。
  - 例外: コントロールのプロパティが混在状態である場合は、空白の既定値を表示します。混在状態は、設定が同じではない複数のオブジェクトのプロパティを表示するときに発生します。

## プロンプト

プロンプトは、編集可能なドロップダウン リスト内に既定値として配置されているラベルまたは短い指示です。静的テキストとは違い、ユーザーがコンボ ボックスに入力を開始するかテキスト ボックスに入力フォーカスが移動すると、プロンプトは画面から消去されます。



### 典型的なプロンプト

プロンプトは、次の場合に使用します。

- テキスト ボックスがツールバーのような特別な領域にあり、ラベルや説明がない方が良い場合。
- 主に一覧の目的を省スペースで示す場合。コンボ ボックスの使用中にユーザーが確認する必要のある重要な情報は、プロンプトでは示しません。

ユーザーに対して一覧からの選択やボタンのクリックを指示するだけのプロンプトは使用しません。たとえば、"オプションを選択" や "ファイル名を入力して [送信] をクリック" などのプロンプトは必要ありません。

プロンプトを使用するときは、以下のこと留意します。

- プロンプト テキストは灰色で表示し、実際に入力されたテキストは通常の黒色で表示します。プロンプト テキストと実際に入力されたテキストが混同されないようにします。
- プロンプト テキストは常に簡潔にします。文ではなく語句にすることもできます。
- センテンス スタイルの大文字化**を使用します。
- 末尾に句読点や省略記号は付けません。
- プロンプト テキストは編集可能にしません。ユーザーがテキスト ボックス内をクリックしたり、Tab キーでテキ

スト ボックスに移動したら、画面からプロンプト テキストを消去します。

- 例外: テキスト ボックスに既定で入力フォーカスがある場合は、プロンプトを表示し、ユーザーが入力を開始したときに画面から消去します。
- テキスト ボックスから入力フォーカスが移動したときに、テキスト ボックスが空白のままの場合は、プロンプト テキストを復元します。

間違った例:

Advanced search:

Any genre: ▾

Any language: ▾

Any country or region: ▾

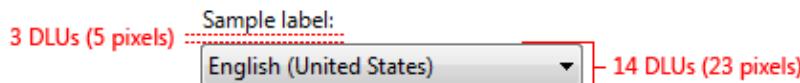
Any state (US only): ▾

Any band (AM, FM): ▾

Keyword, call sign, frequency: ▾

この例では、画面領域が重視されていません。編集可能なドロップダウンリストにいったん入力すると、何のための入力項目だったかわかりづらくなります。さらに、プロンプト テキストが編集可能になっていて、描画も実際に入力されたテキストと同じになっています。

## 推奨されるサイズと間隔



### ドロップダウンリストとコンボ ボックス推奨されるサイズと間隔

- 有効な最長データが適切に表示される幅を選択します。ドロップダウンリストは水平方向にスクロールできないので、ユーザーにはコントロールに表示されている部分しか見えません(ただし、コンボ ボックスの場合は、自動スクロール機能を効果的にできます)。
- ローカライズの対象となるすべてのテキスト(数値以外)について、30% (短いテキストの場合は最大 200%) の余白を追加します。
- 一覧の長さは、垂直方向のスクロール操作ができるだけ少なくて済むサイズにします。ドロップダウンリストは必要なときに表示するので、一覧には最大 30 アイテムまで表示することをお勧めします。ドロップダウン ボタンがない編集可能なリスト ボックスでは、3 ~ 12 個のアイテムの表示をお勧めします。

## ラベル

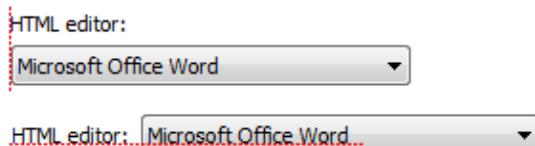
### コントロール ラベル

- ラベルは文ではなく語句にして、末尾にコロンを付けます。

次の場合は例外です。

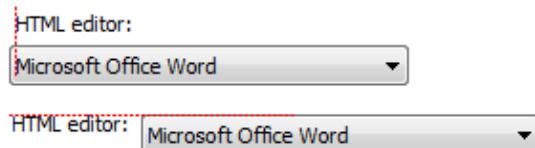
- 画面領域が貴重な場所に配置するプロンプト付きの編集可能なドロップダウンリストの場合、ラベルは付けません。
- ラジオ ボタンまたはチェック ボックスに従属し、コロンで終わるラベルの後に配置するドロップダウンリストまたはコンボ ボックスには、追加のラベルを付けません。
- 各ラベルに、一意な[アクセスキー](#)を割り当てます。ガイドラインについては、「[キーボード](#)」を参照してください。
- [センテンススタイルの大文字化](#)を使用します。
- ラベルはコントロールの左側に配置するか、または上に配置してコントロールの左端と揃えます。ラベルを左側に配置する場合は、ラベルのテキストとコントロール内のテキストが水平になるようにします。

正しい例:



この例では、ラベルとコントロール内のテキストが適切に配置されています。

間違った例:



この例では、ラベルとコントロールのテキストの配置が不適切です。

- ラベルの後ろに単位(秒、接続など)をかっこで囲んで指定することができます。
- ローカライズがないため、ドロップダウンリストまたはコンボボックスの内容(もしくはその表示単位ラベル)を文の一部にはしません。

#### オプション テキスト

- 各オプションに一意な名前を割り当てます。
- アイテムが固有名詞である場合を除き、[センテンススタイルの大文字化](#)を使用します。
- ラベルは文ではなく語句にして、末尾に句読点は付けません。
- 同じ文法構造の表現を使用し、すべてのオプションの長さがおおよそ同じになるようにします。

#### 指示テキスト

- ドロップダウンリストまたはコンボボックスについての指示テキストを追加する必要がある場合は、ラベルの上に追加します。文を使用し、末尾に句点を付けます。
- [センテンススタイルの大文字化](#)を使用します。
- 必須ではなく参考程度の追加情報は、短くまとめます。このような情報はラベルとコロンとの間に配置してかっこで囲むか、コントロールの下にかっこで囲まずに配置します。



この例では、追加情報がコントロールの下に配置されています。

#### ドキュメント

ドロップダウンリストに言及するときは、以下のことに留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、アクセスキーを示すかっこや下線付き文字、およびコロンは除外し、内容に応じて"ボックスの一覧"または"ボックス"のいずれか適切な方を付け加えます。
- 一覧内のオプションについては、大文字と小文字の区別を含め、オプションテキストを正確に引用します。
- プログラミングおよびその他の技術文書では、"ドロップダウンリスト"と記述します。それ以外のドキュメントでは、"ボックスの一覧"または"ボックス"のどちらか適切な方を使用します。
- ユーザー操作を説明する場合は、"クリック"を使用します。
- ラベルおよび一覧内のアイテムは半角の角かっこ([ ])で囲み、可能な場合は太字にします。

例: [フォント サイズ] ボックスの一覧の [大きいフォント] をクリックします。

コンボボックスに言及するときは、以下のことに留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、アクセスキーを示すかっこや下線付き文字、およびコロンは除外し、内容に応じて"ボックスの一覧"または"ボックス"のいずれか適切な方を付け加えます。

一覧内のオプションについては、大文字と小文字の区別を含め、オプションのテキストを正確に引用します。

- プログラミングおよびその他の技術文書では、"コンボ ボックス" と記述します。それ以外のドキュメントでは、"ボックスの一覧" または "ボックス" のどちらか適切な方を使用します。
- ユーザー操作を説明するには、"クリック" または "入力" を使用します。
- ラベルおよび一覧内のアイテムは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例: [フォント] ボックスに使用するフォントを入力します。

## グループ ボックス

適切なコントロールかどうかの判断基準

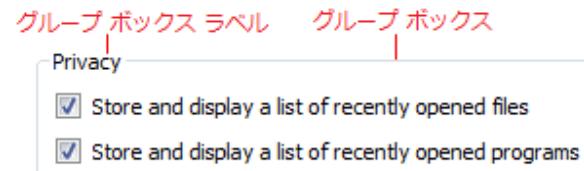
ガイドライン

ラベル

ドキュメント

"グループ ボックス" とは、関連するコントロールのセットを囲むラベル付きの四角い枠のことです。

グループ ボックスは関連性を視覚的に表示する方法であり、コントロール グループごとのアクセスキーを割り当てる能够性があります。



典型的なグループ ボックス

注: レイアウトに関するガイドラインは、別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

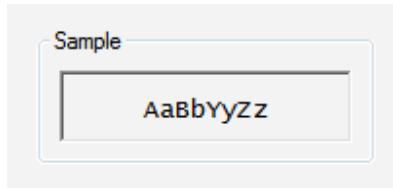
グループ ボックスは関連性を示す有効な視覚的手段ですが、使いすぎると見た目が乱雑になり、利用できるスペースが大幅に減ります。グループ ボックスは視覚的に場所を取るため、他に良い解決策がない場合に限り、控え目に使用する必要があります。

Windows® の設計は、不要な線を排除して、外観を簡素にすっきりさせる方向に向かっています。

グループ ボックスが必要かどうかは、以下のことから判断します。

- グループ内に複数のコントロールがあるかどうか。該当しない場合は、代わりに単純なテキスト ラベルを使用します。めったにない例外として、同一領域内のグループ ボックスとの一貫性を保つために、コントロールが 1 つだけ含まれるグループ ボックスを使用することがあります。

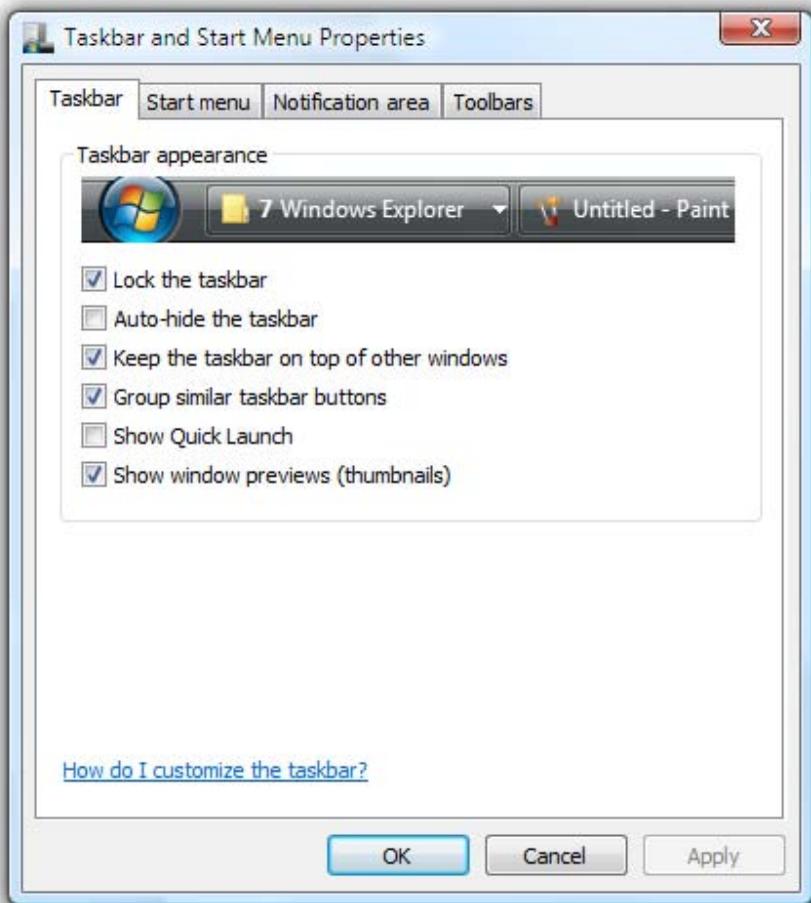
間違った例:



この例では、グループ ボックスに 1 つしかコントロールがありません。

- 各コントロールに関連があるか。また、関連性を示すことでわかりやすくなるか。該当しない場合は、各コントロールをグループ ボックス外で個別に表示します。
- すべてのコントロールがグループ内にあるか。該当する場合は、親ダイアログ ボックスやページなどの広い領域で関連性を示します。

間違った例:



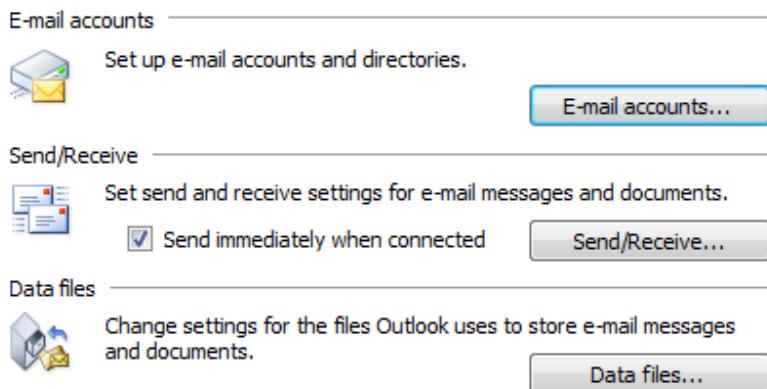
この例では、ダイアログ ボックス内のすべてのコントロール(コミット ボタンを除く)がグループ ボックス内にあります。

- レイアウトだけで効果的に関連性を伝えることができるか。該当する場合は、代わりに[レイアウト](#)を使用します。コントロールが互いに関連する場合は並べて配置し、関連しない場合は余分に間隔を空けます。インデントによって階層関係を示すこともできます。



この例では、レイアウトだけを使用してコントロールの関連性を示しています。

- 区切り記号を使用して効果的に関連性を伝えることができるか。該当する場合は、代わりに区切り記号を使用します。区切り記号とは、コントロールをまとめる横線のことです。区切り記号を使用すると、外観がすっきりします。グループ ボックスとは異なり、区切り記号は領域いっぱいの幅になると最も効果的です。
  - 開発者向け情報: 高さ 1 のくっきりした四角形を使用して区切り記号を実装できます。



この例では、ラベル付けされた区切り記号を使用してコントロールの関連性を示しています。

---

Type: File Folder  
Location: C:\  
Size: 1.36 GB (1,466,201,432 bytes)  
Size on disk: 1.38 GB (1,483,361,757 bytes)  
Contains: 8,130 Files, 567 Folders

---

Created: Sunday, July 29, 2007, 10:10:35 AM

---

Attributes:  Read-only  Hidden  Advanced

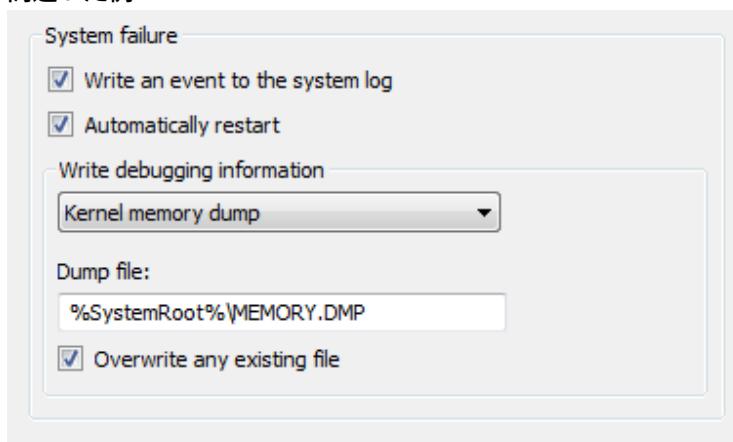
この例では、ラベル付けされていない区切り記号を使用してコントロールの関連性を示しています。

- テキストなしで効果的に関連性を伝えることができるか。該当する場合は、[背景](#)または[アグリゲーター](#)などのグラフィック要素の使用を検討します。

## ガイドライン

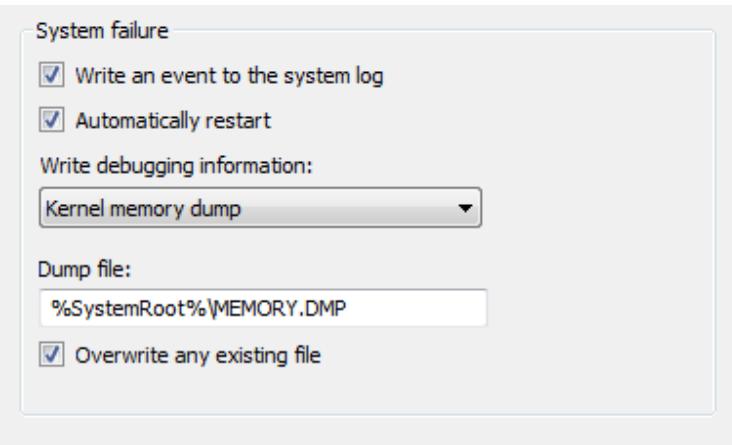
- グループボックスを入れ子にしないようにします。グループボックス内の関連性を示すには、レイアウトを使用します。

間違った例:



この例では、入れ子になったグループボックスによって見た目が煩雑になっています。

正しい例:



この例では、代わりにレイアウトを使用して同じコントロールの関連性を示しています。

- グループ ボックス ラベルにコントロールを配置しないようにします。
  - 例外: ボックス内のすべてのコントロールの有効/無効をチェック ボックスで切り替える場合は、チェック ボックスをグループ ボックス ラベルとして使用できます。

間違った例:



この例では、ドロップダウン リストがグループ ボックスに配置されています。ここでは、代わりにタグを使用する必要があります。

- グループ ボックスを無効にしないようにします。コントロール グループが現在適用されていないことを示すには、グループ ボックスそのものではなく、グループ ボックス内のすべてのコントロールを無効にします。

## ラベル

- すべてのグループ ボックスにラベルを付けます。
- ラベルにはアクセス キーを割り当てません。これは不要な操作であり、他のアクセス キーの割り当てが困難になります。代わりに、グループ ボックス内のコントロールにアクセス キーを割り当てます。
  - 例外: 画面内に多数のコントロールがある場合は、使用できるアクセス キーが不足する場合があります。該当する場合は、グループ ボックス内のコントロールではなく、グループ ボックスにアクセス キーを割り当てることで、アクセス キーの数を減らします。
- センテンス スタイルの大文字化を使用します。
- ラベルには文ではなく名詞または名詞句を使用し、コロンなどの句読点を使用しないようにします。
- 同じ画面内のグループ ボックス ラベルには同じ文法構造の表現を使用します。
- グループ ボックス ラベルは簡潔にします。指示テキストはラベルとして使用しません。ただし、グループ ボックスに指示テキストを含めることはできます。
- ボックス内のコントロール ラベルでグループ ボックス ラベルの語句を繰り返さないようにします。たとえば、グループ ボックスに "配置" というラベルが付いている場合、オプション ボタンには "左配置" や "右配置" では

なく、"左"、"右"などのラベルを付けます。

- ユーザーインターフェイスのテキストでグループボックスに言及しないようにします。

## ドキュメント

グループボックスに言及するときは、以下のこと留意します。

- プログラマー向けドキュメントやその他の技術文書でのみ、グループボックスに言及します。記述する場合は、"グループボックス"と表記します。
- それ以外の場合はすべて、グループボックスの名前を手順に含める必要はありません。ただし、ダイアログボックスに同じ名前のオプションが複数存在する場合は例外とし、"[<グループボックス名>]"と表記します。
- ラベルは半角の角かっこ([ ])で囲み、可能な場合は太字にします。

例: [効果] の [非表示] を選択します。

## リンク

適切なコントロールかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

対話操作

色

下線

アイコン付きテキストリンク

グラフィックのみのリンク

ナビゲーションリンク

タスクリンク

メニューリンク

リンクの情報ヒント

テキスト

ドキュメント

"リンク"を使用すると、別のページ、ウィンドウ、またはヘルプトピックへの移動、定義の表示、コマンドの開始、またはオプションの選択を実行できます。リンクは、クリックできることがわかるテキストまたはグラフィックです。通常、表示済みまたは未表示を示すリンクシステムカラーを使用して表示されます。従来、リンクには下線も付いていましたが、この下線は不要な場合が多く、見た目をすっきりさせるために使われなくなっています。

"表示済み"と"未表示"

典型的なリンクテキスト。

リンク上にマウスポインターを合わせると、リンクテキストに(まだ付いていなかった場合)下線が付けられて表示され、ポインターが手の形に変化します。

テキストリンクは最もシンプルなクリック可能なコントロールであり、多くの場合、デザインの外観をすっきりさせるために使用します。

注: コマンドリンクおよびレイアウトに関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- 別のページ、ウィンドウ、ヘルプトピックへの移動、定義の表示、コマンドの開始、またはオプションの選択を実行するためにリンクが使用されているかどうか。該当しない場合は、別のコントロールを使用します。
- コマンドボタンの方が適しているかどうか。次の場合には、コマンドボタンを使用します。
  - コントロールがウィンドウの表示などの即時的な操作を開始し、そのコマンドはウィンドウの主な目的に関連している。
  - 副コマンドであっても、情報の入力や選択を行うためにウィンドウが表示される。
  - ラベルが短く、4語あるいは12文字以下で構成されているので、ボタンを使用しても見苦しい外観にならない。
  - コマンドがインラインではない。
  - コントロールが他の関連するコマンドボタンのグループ内に表示されている。
  - リスクのあるアクションや取り消しできないアクションである。リンクは戻ることも可能な移動の手段と考えられているため、重要な結果をもたらすコマンドには適していません。
  - コマンドが確定を表すウィザードまたはタスクフロー。これらのウィンドウでは、コマンドボタンは確定を示し、リンクは次のステップへの移動を示します。

詳細な比較については、「[コマンドボタンとリンクの比較](#)」を参照してください。

### デザインコンセプト

識別しやすいリンクの作成

リンクにはアフォーダンスが欠如しています。アフォーダンスの欠如とは、視覚的特性では使用方法がわからず、使ってみて初めて理解されるということです。下線やリンクシステムカラーのないリンク

は、通常のテキストのように見えます。動作を確かめるには、リンクの表示やコンテキストから判断するか、リンクにポインターを合わせる必要があります。

リンクの見た目がシンプルでデザインが単純化されるのは、アフォーダンスが欠如しているからですが、驚くことに、これを理由として使用されることがよくあります。リンクには、[コマンド ボタン](#)に使用される太いフレームや、その他のコントロールに使用される境界線はありません。たとえば、主コマンドにはコマンド ボタンを使用して目立つようにし、副コマンドにはリンクを使用して目立たないようにすることができます。

ここでの課題は、ユーザーがリンクを識別できるように視覚的な手掛けりを与えることです。基本的なガイドラインは、リンクを視覚的に識別できるようにし、リンクかどうかを判断するためにオブジェクト上にカーソルを合わせたり、オブジェクトをクリックすることをユーザーに要求しないということです。

リンクに[リンク システム カラー](#)や次のような視覚的な手掛けりが1つ以上使用されていれば、ユーザーは視覚的にリンクを識別できます。

- 下線付きテキスト
- [アイコン付きテキスト](#) リンク パターンなど、グラフィックや箇条書きの記号
- ウィンドウの[コンテンツ領域](#)などの標準的なナビゲーション、オプション、またはコマンドの場所に配置したり、ナビゲーションバー、メニューバー、ツールバー、またはページフッターに配置することでリンクであることを示す場合

ユーザーは次のような視覚的な手掛けりによってリンクを識別することもできますが、これらの手掛けりだけでは十分ではありません。

- クリックを示唆するテキスト ("Show (表示)"、"Print (印刷)"、"Copy (コピー)"、"Delete (削除)"などの動詞の命令形で始まる(英語の場合)コマンドなど)
- 通常のテキスト ブロック内への配置

もちろん、カーソルを合わせる、またはクリックするという対話操作によって、ユーザーはいつでもリンクを判断できます。いずれの重要なタスクでも、リンクを見つけることが必要されない場合、リンクを目立たないようにすることができます。

Contact Us | Terms of Use | Trademarks | Privacy Statement

この例では、[お問い合わせ先]、[使用条件]、[商標]、[プライバシーに関する声明]がリンクになっています。これらは重要なタスクには必要ではないため、意図的に目立たないようにされています。これらがリンクであるという手掛けりは、マウスでポイントしたときにポインターが変化することと、ウィンドウの一番下の標準的なナビゲーション領域に配置されていることだけです。

具体的で関連性を示し、アクションを予測できるリンクの作成

リンク テキストは、リンクをクリックした結果を示している必要があります。

具体的なリンクは、通常のリンクよりもユーザーの注意を引きます。リンクをクリックした結果に関する具体的かつ説明的な情報を表示するリンク ラベルを使用します。ただし、具体的すぎるリンク テキストによって誤解を招いたり、誤って使用されたりしないように注意してください。

簡潔なリンクは、冗長なリンクに比べて読みやすくなります。不要なテキストや詳細は削除します。リンク ラベルですべてをカバーする必要はありません。

リンク テキストを評価するには、以下のことに留意します。

- リンクでサポートされているシナリオがリンク テキストに反映されるようにします。
- リンクの結果が予測可能であるようにします。ユーザーがその結果に驚くことがないようにしてください。

## 2つの重要な点

- 視覚的にリンクを見つけられるようにします。リンクを見つけるためにユーザーがプログラムを操作する必要がないようにしてください。
- 必要な長さのテキストを使用して、クリックした結果に関する具体的で説明的な情報を表示するリンクを使用します。リンク テキストとオプションの[情報ヒント](#)から、リンクの結果を正確に予測できるようにします。

## 使用パターン

リンクにはいくつかの機能パターンがあります。

ナビゲーション リンクをクリックすると、ブラウザ ウィンドウやウィザードのようにインプレースで別のページに移動するか、新しいウィンドウが表示されます。タスク リンクとは異なり、ナビゲーションではタスクは開始されず、単に別の場所へ移動したり、既に進行中のタスクを継続するだけです。ユーザーはいつでも戻ることができます。ナビゲーションは安全であるといえます。  
リンク 移動するために  
別のページまたは  
はウィンドウに  
使用するリンク  
ニュースの見出し  
です。

この例では、リンクをクリックするとニュースの見出しページに移動します。

タスク リンク リンクをクリックすると、コマンドがすぐに実行されたり、追加情報を入力するためのダイアログ  
新しいコマンド ボックスやページが表示されたりします。ナビゲーション リンクとは異なり、タスク リンクでは、既  
を開始するため 存のタスクを継続するのではなく、新しいタスクを開始します。タスクは、[戻る] コマンドで前の状態  
に使用するリンク に戻ることができないという意味において、安全ではありません。タスク リンクは、[コマンド リン](#)  
クです。 クとの混乱を避けるためにそのように呼ばれています。

### ログイン

この例では、リンクをクリックするとログイン コマンドが開始されます。

ヘルプ リンク リンクをクリックすると、別のウィンドウにヘルプの項目が表示されます。

ヘルプ トピック  
を表示するため  
を表示するため

### 強力なパスワードとは

この例では、リンクをクリックすると指定されたトピックのヘルプ ウィンドウが表示されます。

に使用するテキ  
スト リンクで  
す。

定義 リンク このパターンは、ユーザーが知らない可能性がある用語の定義を、画面を乱雑にすることなく表示でき  
ユーザーがリ  
ンクをクリックす  
るか、リンク上  
にマウス ポイン  
ターを合わせた  
ときに、情報ヒ  
ントで定義を表  
示するために使  
用するテキスト  
リンクです。

## Using Windows Defender

It's important to run antispyware software whenever you're using your computer. [Spyware](#) and other potentially unwanted software can try to install itself on your computer any time you connect to the Internet. It can also infect your computer when you install some programs using a CD, DVD, or other removable media. Potentially unwanted or malicious software can also be programmed to run at

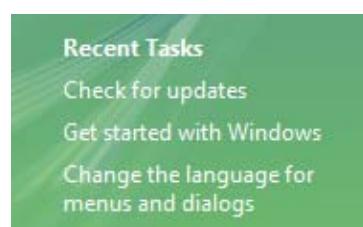
Anything used for information storage that is designed to be easily inserted into and removed from a computer or portable device. Common removable media include CD and DVD discs, as well as removable memory cards.

ally unwanted software

potentially unwanted software attempts to install itself or to run on your computer. It also alerts you when programs attempt to change important Windows settings.

この例では、情報ヒントで定義が表示されています。

メニュー リンク メニューのコンテキストはリンク セットを示しているため、通常はテキストに下線は付かず(マウスで  
メニューを作成  
するために使  
用するタスク リン  
クのセットで  
す。



この例では、メニューがリンク セットで構成されています。

オプション リン  
ク 通常のテキスト リンクとは異なり、リンクをクリックすると現在選択されたオプションの内容を反映  
するようにテキストが変更され、リンクは常に未表示リンク カラーで表示されます。

選択オプション  
またはその  
プロセスホルダー  
です。リンクを

Apply this rule after the message arrives  
from [people or distribution list](#)  
move it to the [specified folder](#)

Apply this rule after the message arrives  
from [people or distribution list](#)  
move it to the [RSS Feeds folder](#)

クリックすると、そのオプションを変更するコマンドが実行されます。

左の例では、プレースホルダー オプションを使用した Microsoft® Outlook® の自動仕分けウィザードのルールを示しています。リンクをクリックしていくつかのオプションを選択すると、右側の例でリンクテキストが更新され、結果が表示されます。

Apply this rule after the message arrives  
from [people or distribution list](#)  
move it to the [specified folder](#)

Apply this rule after the message arrives  
from [people or distribution list](#)  
and with [specific words](#) in the body  
and with [specific words](#) in the subject or body  
move it to the [RSS Feeds folder](#)

右の例では、Outlook の仕分けルールが変数形式であることが示されています。



左の例ではオプション リンクが表示されています。オプション リンクをクリックすると、右の例に示されているようなドロップダウンリストになります。

リンクにはいくつかの表示パターンがあります。

プレーンテキスト この表示は、[インライン](#)を含めてどこでも使用できるため、最も柔軟性の高いパターンです。  
トリンク  
テキストのみで構成されます。

[Post a question or search for an answer in Windows communities.](#)

アイコン付きテキストリンク 機能を示すアイコンが前に付いたテキストです。



この例では、アイコンによってリンクの識別性が向上しています。

| ▶ Play |

この例では、標準的な三角形の再生記号によって、このテキストがコマンドであることが示されています。

グラフィックのみのリンク グラフィックのみで構成されています。リンクが示すリンク カラーや下線は存在しません。このリンクは、クリックを示唆するグラフィック デザインか、ユーザーがクリックしたときの操作を示唆するグラフィック内のテキストに依存します。グラフィックのみのリンクには、リンクであることを示すために、マウス ポインターを重ねると表示が変化するものがあります。このアプローチは役立ちますが、見ただけでは判別できません。



この例では、見ただけではリンクを判別できません。

認識やローカライズに関する潜在的な問題があるので、グラフィックのみのリンクをタスクを実行する唯一の方法にすることはお勧めできません。

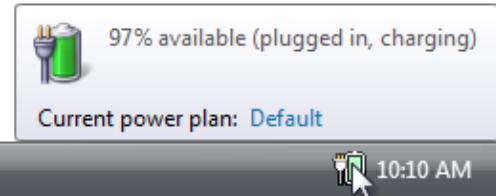
## ガイドライン 対話操作

- リンクをクリックした結果がすぐに表示されない場合は、ビジー ポインターを表示します。何もメッセージが表示されないと、ユーザーはクリックされなかったと見なして、再度クリックする可能性があります。

## 色

- 表示済みリンクと未表示リンクにテーマの色やリンクシステム カラーを設定します。これらの色の意味はすべてのプログラムで一貫しています。ユーザーがこれらの色を何らかの理由で(おそらくアクセシビリティ上の理由で)気に入らない場合は、自分で変更できます。
- ナビゲーション リンクでは、表示済みリンクと未表示リンクに異なる色を設定します。表示済みリンクの履歴はプログラム インスタンスの継続中のみ保持します。表示済みカラーは、ユーザーが既にアクセスしたリンクを示し、意図せずに同じページに何度もアクセスしないようにするために重要です。
- その他の種類のリンクでは、表示済みリンク カラーを使用しないようにします。たとえば、コマンドを"表示済み"と指定する価値はありません。
- ユーザーがリンクと見なす可能性があるので、リンクでないテキストには色を付けないようにします。色付きテキストは使わずに、太字または灰色を使います。  
例外: すべてのリンクに下線が付いているか、リンクが標準的なナビゲーションやコマンドの場所に配置されている場合、色付きテキストを使用できます。

間違った例:



この例では、リンクでないテキストに青色が使用されているので不適切です。

- リンク カラーと対照的な背景色を使用します。 ウィンドウ システム カラーは常に最適な選択です。

間違った例:

Start by checking [What's New in Windows Vista](#). These articles summarize the new Windows Vista core UI features that you should use in your Windows Vista UI designs, and how they differ from Windows XP.

この例では、背景色とリンク カラーの差があまりありません。

## 下線

- 主なタスクを実行するために必要なリンクの場合、ユーザーが見ただけでリンクを識別できるように、視覚的な手掛かりを表示します。この手掛かりには、下線、グラフィック、行頭文字や標準的なリンクの場所があります。リンクかどうかを判断するためにオブジェクト上にマウス ポインターを合わせたり、クリックすることをユーザーに要求しないようにします。リンクがコンテキストの中で目立っていない場合、下線付きテキストを使用します。
- リンクでないテキストには、ユーザーがリンクと見なす可能性があるので下線を付けないようにします。下線は使用せずに、斜体を使用します。下線はリンク専用にします。
- 下線またはリンク カラーは印刷されないようにします。印刷されたリンクには価値がなく、また混乱を招く可能性があります。

## アイコン付きテキスト リンク

- 矢印アイコンはコマンド リンクのみに使用します。Windows XP でコマンド リンクの代わりとして使用される場合を除いて、通常のリンクでは矢印アイコンは使用しません。
- アイコンはテキストの左に配置します。アイコンによってテキストに目が向くようにする必要があります。

正しい例:



間違った例:



この間違った例では、アイコンによってテキストに目が向きません。

アイコンをクリックしても、テキストをクリックしても同じ結果になるようにします。このようにしないと、予想外の動作で混乱を招きます。

## グラフィックのみのリンク

- グラフィックのみのリンクは使用しないようにします。グラフィックのみのリンクはリンクとして認識しにくく、グラフィック内のテキスト(クリックしたときの操作を示すために使用)でローカライズの問題が発生します。

## ナビゲーション リンク

- ナビゲーション リンクが確認を求めるようにします。インプレース ナビゲーションの場合は[戻る]、新しい ウィンドウを閉じる場合は[キャンセル]を使用して、いつでも最初の状態に戻れるようにします。
- 範囲の広いコンテンツではなく特定のコンテンツにリンクします。たとえば、ドキュメントの先頭にリンクするのではなく、関連セクションにリンクすることをお勧めします。
- リンク先の資料が冗長なものではなく、有益で関連性が高い情報である場合にのみリンクを使用します。ナビゲーション リンクの使用を制約します。できるからという理由だけでリンクを使用しないでください。
- リンクで外部サイトに移動する場合、情報ヒントに URL を表示します。これにより、ユーザーはリンク先を知ることができます。
- 最初に出現するリンク テキストのみをリンクします。重複したリンクは不要であり、テキストが読みにくくなることがあります。

### 正しい例:

[画像] フォルダーで画像を簡単に共有できます。画像を電子メールで送信したり、Web 上の安全なプライベートな場所で公開するには、[画像] のタスクを使用します。[画像] フォルダーから画像を直接印刷することもできます。

### 間違った例:

[画像] フォルダーで画像を簡単に共有できます。画像を電子メールで送信したり、Web 上の安全なプライベートな場所で公開するには、[画像] のタスクを使用します。[画像] フォルダーから画像を直接印刷することもできます。

正しい例では、最初に出現する関連テキストのみにリンクが付けられています。

### 次の場合は例外です。

- 指示にリンク テキストが含まれている場合、指示内にリンクを設定します。

強力なパスワードを使用することが非常に重要です。詳細については、「[強力なパスワード](#)」を参照してください。

この例では、最初に出現したリンク テキストではなく、指示内にリンクが設定されています。

- 最初に出現したリンク テキストと後のリンク テキストが離れている場合、後に出現したリンク テキストにもリンクします。たとえば、ヘルプ トピック内の異なるセクションで重複してリンクを設定できます。

## タスク リンク

- タスク リンクはリスクのないコマンド、または簡単に元に戻せるコマンドで使用します。リンクは戻ることも可能な移動の手段と考えられているため、重要な結果をもたらすコマンドには適していません。ダイアログ ボックスや確認が表示されるコマンドが適しています。

### 正しい例:

[開始](#)

[停止](#)

### 間違った例:

[ファイルの削除](#)

間違った例では、コマンドがリスクのあるものとなっています。

## メニュー リンク

- 関連するナビゲーション リンクとタスク リンクをグループ化してメニューを作成します。関連リンクのメニューを標準的なナビゲーションやコマンドの場所に配置すると、リンクを別々に配置するよりもわかりやすくなり、簡単に見つけることができます。

- 選択に依存するメニューでは、適用されないメニュー リンクは削除します。無効にはしないでください。削除することで乱雑さがなくなり、選択対象のリンクを見落とすことがなくなります。
- 選択に依存しないメニューでは、適用されないメニュー リンクは無効にします。削除はしないでください。無効にすることでメニューが固定化され、リンクを簡単に見つけることができます。



この *Windows Update* の例の場合、更新中であるため、[更新プログラムの確認] コマンドは削除されずに無効になっています。

#### リンクの情報ヒント

- リンクに説明を追加する必要がある場合、別のテキストコントロールか情報ヒントに補足説明を表示します。両方に表示する必要はありません。文を使用し、末尾に句点を付けます。テキストが同じであれば両方に表示する必要はなく、またテキストが異なると混乱を招きます。



##### Train your computer to better understand you

Read text to your computer to improve your computer's ability to understand your voice. Doing this isn't necessary, but can help improve dictation accuracy.

この例では、リンクに関する追加情報が補足説明で提供されています。

You'll find a Search box at the top of every folder. As you type in the Search box, the contents of the folder are immediately **filtered** to show only those files that match what you typed. To display files that meet a certain criteria. For example, you might filter files by a particular author so that you only see the files written by that person. Filtering does not delete files, it simply changes the view so that you only see the files that meet your criteria.

この例では、情報ヒントに追加情報を表示しています。

- リンク テキストを繰り返すだけの情報ヒントは提供しません。

間違った例:



この例では、情報ヒントの繰り返しをユーザーが不快に感じる可能性があります。

## テキスト

- **アクセスキー**は割り当てません。リンクには Tab キーでアクセスできます。
- 必要な長さのテキストを使用して、クリックした結果に関する具体的で説明的な情報を表示するリンクを使用します。リンク テキストは、リンクをクリックした結果を示している必要があります。リンク テキストとオプションの情報ヒントから、リンクの結果を正確に予測できるようにします。

間違った例:



### Security notice

この例では、重要なリンクのようですが、ラベルが一般化されすぎています。ユーザーは具体的なリンクをクリックする傾向があります。

- インラインリンクでは、以下のことに留意します。
  - テキストの大文字と小文字の設定および句読点を維持します。
  - テキストが質問である場合を除いて、リンクの末尾に句読点は付けません。
  - テキストの最も関連性の高い部分にリンクを設定し、クリックしやすい長さのリンク テキストを選択します。

正しい例:

[ニュースグループ](#)に移動します。

間違った例:

[ニュースグループ](#)に[移動](#)します。

これらの例では、"移動"は、"ニュースグループ"に対して、テキストの最も関連性の高い部分ではなく、またクリックしやすい長さのテキストでもありません。

- 2つの異なるインラインリンクが隣接しないようにします。ユーザーはこれらを1つのリンクと考える可能性があります。

間違った例:

詳細については、「[UX ガイドライン](#)」を参照してください。

この例では、"UX"と"ガイドライン"は2つの異なるリンクです。

- 独立した(オンラインではない)リンクの場合、以下のことに留意します。
  - **センテンススタイルの大文字化**を使用します。
  - リンクが質問である場合を除いて、末尾に句読点は付けません。
  - テキスト全体をリンクにします。
- 画面上の他のリンクと区別できるリンクを使用します。ユーザーが各リンク先を正確に予測でき、区別できるようにします。

間違った例:

[ウイルス対策ソフトウェアの検索](#)

[ウイルス対策ソフトウェアの取得](#)

正しい例:

[ウイルス対策ソフトウェアがインストールされているかどうかを確認する方法](#)

[ウイルス対策ソフトウェアのインストール](#)

間違った例では、2つのリンクの違いがわかりづらくなっています。

- リンク テキストに"クリック"や"ここをクリック"は追加しません。リンク自体がクリックすることを示唆しているため、こうした言葉は必要ありません。また、"ここをクリック"や"ここ"だけでは、スクリーンリーダーで読み取ったときにリンクに関する情報が伝わりません。

間違った例:

[説明を表示するには、ここをクリックしてください。](#)

[説明を表示するには、ここをクリックしてください。](#)

[説明を表示するには、ここをクリックしてください。](#)

正しい例:

[説明](#)

間違った例では、言うまでもなく "ここをクリック" はリンクに関する情報を伝えています。

## ナビゲーション リンク

- リンクの先頭を名詞にし、リンクをクリックしたときの移動先について明確に説明します。末尾に句読点は付けません。ナビゲーションリンクの先頭を動詞にすることが必要な場合もありますが、"表示"、"開く"、"移動"などの実際のリンクの動作で既に示唆されているナビゲーションを繰り返す動詞は使用しません。
- Web ページに移動するナビゲーションリンクであり、対象となるユーザーが URL を思い出してブラウザーに入力することを想定している場合、リンクを URL として表示します。可能であれば、このような URL は短く、記憶しやすいものにします。
- リンクに "www" で始まる Web サイトへの URL が含まれている場合、http:// プロトコル名を省略し、小文字のテキストを使用します。

間違った例:

<http://www.microsoft.com>

[www.microsoft.com](http://www.microsoft.com)

正しい例:

[microsoft.com](http://microsoft.com)

間違った例では、言うまでもなく "http://" や "www" は必要ありません。

## タスク リンク

- リンクは動詞の命令形で始め (英語の場合)、リンクで実行されるタスクを明確に記述します。末尾に句読点は付けません。
- 正常にタスクを完了するためにコマンドに追加情報 (確認など) が必要な場合、リンクの末尾に省略記号を付けます。タスクが正常に完了したときに別のウィンドウが表示される場合、省略記号は使用しません。タスクの実行に追加情報が必要な場合に限ります。

[印刷...](#)

この例では、[印刷...] コマンド リンクをクリックすると、情報を入力するための [印刷] ダイアログ ボックスが表示されます。

[印刷](#)

この例では、[印刷] コマンド リンクをクリックすると、ユーザーの追加操作なしでドキュメントのコピーが既定のプリンターに 1 部印刷されます。

ユーザーがタスクの実行前に追加で選択を行ったり、タスクを完全に取り消したりできることを示すことができる所以、省略記号の適切な使用は重要です。省略記号によって与えられた視覚的な手掛かりにより、ユーザーは安心してソフトウェアを使用できます。

- ナビゲーションリンクと区別するために、必要に応じてタスク リンクの先頭に "今すぐ" を付けます。

[ファイルをダウンロード](#)

[今すぐファイルをダウンロード](#)

この例では、"ファイルをダウンロード" をクリックすると、ファイルをダウンロードするページに移動し、"今すぐファイルをダウンロード" をクリックすると、実際にコマンドが実行されます。

## ヘルプ リンク

ガイドラインと使用例については、「[ヘルプ](#)」を参照してください。

## リンクの情報ヒント

- 文を使用し、末尾に句点を付けます。

その他のガイドラインと例については、「[ツールヒントと情報ヒント](#)」を参照してください。

## ドキュメント

リンクに言及するときは、以下のこと留意します。

- 大文字と小文字の区別を含め、リンク テキストを正確に引用しますが、省略記号は含めません。
- ユーザー操作を説明する場合は、"クリック" を使用します。
- リンク テキストを半角の角かっこ ([ ]) で囲み、可能な場合は、リンク テキストを太字にします。

例: スキャンを開始するには、[コンピューターのスキャン] をクリックします。

## リスト ボックス

適切なコントロールかどうかの判断基準

使用パターン

ガイドライン

提示方法

対話操作

複数選択リスト

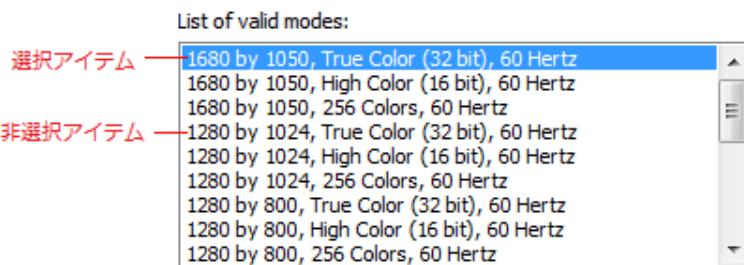
既定値

推奨されるサイズと間隔

ラベル

ドキュメント

"リスト ボックス" を使用すると、常時表示されている一覧に示された一連の値から選択できます。"単一選択リスト ボックス" では、相互に排他的な値の一覧からアイテムを 1 つ選択します。"複数選択リスト ボックス" では、値の一覧からアイテムを選択しないことも、1 個以上選択することもできます。



典型的な単一選択リスト ボックス

注: レイアウトおよびリスト ビューに関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- 一覧に表示されるものが、プログラム オプションではないデータかどうか。どちらの場合でも、リスト ボックスは最適な選択です。アイテム数は関係ありません。それに対して、ラジオ ボタンまたはチェック ボックスはプログラム オプションの数が少ない場合にのみ適しています。
- ビューの変更、グループ化、列ごとの並べ替え、列の幅と順序の変更が必要かどうか。該当する場合は、代わりにリスト ビューを使用します。
- コントロールが、ドラッグ ソースまたはドロップ ターゲットとして機能する必要があるかどうか。該当する場合は、代わりにリスト ビューを使用します。
- リスト アイテムをクリップボード経由でコピーする必要があるかどうか。該当する場合は、代わりにリスト ビューを使用します。

### 単一選択リスト

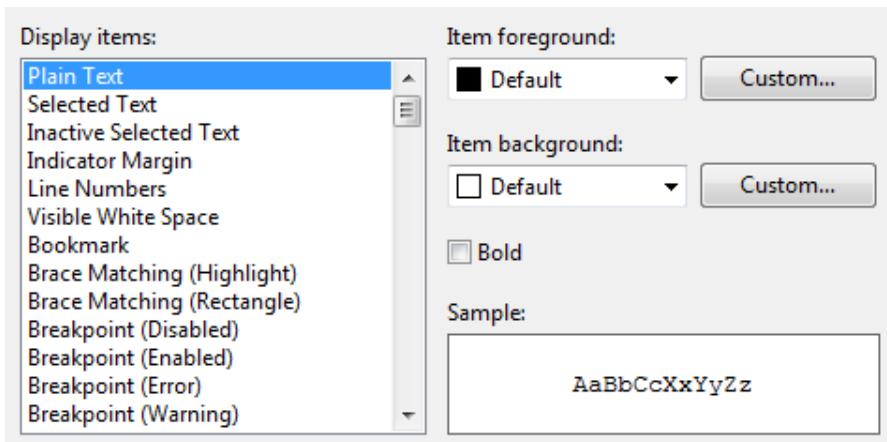
- 相互に排他的な値の一覧から 1 つのアイテムを選択するためのコントロールであるか。該当しない場合は、別のコントロールを使用します。複数のアイテムを選択する場合は、代わりに標準的な複数選択リスト、チェック ボックス リスト、リスト ビルダー、または追加/削除リストを使用します。
- ほとんどの状況下で大多数のユーザーに推奨される既定のオプションがあるかどうか。選択されているオプションが見えることが、他のオプションが見えることよりもはるかに重要かどうかを判断します。該当する場合は、ドロップ ダウン リストを使用して他のオプションを隠すことを検討します。

Colors:

Highest (32 bit) ▾

この例では、画面の色質を最高に設定することがほとんどのユーザーにとって最適な選択となるので、ドロップ ダウン リストが他のコントロールより適しています。

- リストを頻繁に操作する必要があるか。該当する場合は、対話操作を簡素化するために単一選択リストを使用します。



この例では、ユーザーは [表示アイテム] ボックスの一覧の選択アイテムを頻繁に変更して、前景色と背景色を設定します。この場合にドロップダウンリストを使用するのは非常に面倒です。

- その設定が相対量のように見えるかどうか。設定変更の効果に対するフィードバックをすぐに得られることがユーザーの役に立つかどうか。該当する場合は、代わりに[スライダー](#)を使用することを検討します。
- リストアイテムに重要な階層関係があるかどうか。該当する場合は、代わりに[ツリービュー](#)を使用します。
- 画面領域が貴重か。該当する場合は、必要な画面領域が一定でリストアイテム数に依存しない、ドロップダウンリストを使用します。

#### 標準的な複数選択リストとチェックボックスリスト

- 複数選択がタスクに不可欠、または複数選択の使用頻度が高いかどうか。該当する場合は、対象が詳しい知識のあるユーザーではないときは特に、[チェックボックスリスト](#)を使用して、複数選択であることを明確にします。多くのユーザーは[標準的な複数選択リスト](#)で複数選択がサポートされていることを知りません。チェックボックスを使用すると複数選択が強調される、または画面が煩雑になるときは、標準の複数選択リストを使用します。
- 複数選択の安定性が重要かどうか。該当する場合は、チェックボックスリスト、リストビルダー、または追加/削除リストを使用します。これらのコントロールでは1回のクリックで1個のアイテムのみが変更されます。標準の複数選択リストを使用すると、意図しない場合でもすべての選択アイテムが簡単にクリアされます。
- 値の一覧からまったく選択しない、または1個以上のアイテムを選択するためにコントロールを使用するかどうか。該当しない場合は、別のコントロールを使用します。アイテムを1つ選択する場合は、代わりに单一選択リストを使用します。

#### プレビュー リスト

- テキストだけのオプションよりイメージ付きオプションの方が選択しやすいかどうか。該当する場合は、[プレビューリスト](#)を使用します。

#### リストビルダーと追加/削除リスト

- 値の一覧からまったく選択しない、または1個以上のアイテムを選択するためにコントロールを使用するかどうか。該当しない場合は、別のコントロールを使用します。アイテムを1つ選択する場合は、代わりに单一選択リストを使用します。
- 選択したアイテムの順序が重要かどうか。該当する場合は、[リストビルダー](#)および[追加/削除リスト](#)で順序の変更がサポートされますが、その他の複数選択パターンでは順序の変更はサポートされません。
- 選択されたすべてのアイテムの概要を確認することが重要かどうか。該当する場合は、リストビルダーおよび追加/削除リストのパターンでは選択したアイテムのみが表示されますが、その他の複数選択パターンではこのような表示はできません。
- 選択肢に制約がないかどうか。該当する場合は、現時点でリストにない値を選択できるように、追加/削除リストを使用します。
- リストに値を追加する場合、オブジェクトを選択するための専用のダイアログボックスが必要かどうか。該当する場合は、追加/削除リストを使用し、[追加] をクリックするとダイアログボックスが表示されるようにします。
- 画面領域が貴重か。該当する場合は、代わりに追加/削除リストを使用します。追加/削除リストでは、オプションセットは常時表示ではないので、使用する画面スペースを節約できます。

リストボックスは、数千アイテムから1アイテム(複数選択リストでは0アイテム)まで対応できるので、一覧のアイテム数はコントロールを選択する際の判断要素にはなりません。リストボックスはデータ用に使用することもできるので、アイテム数が事前にわからないこともあります。

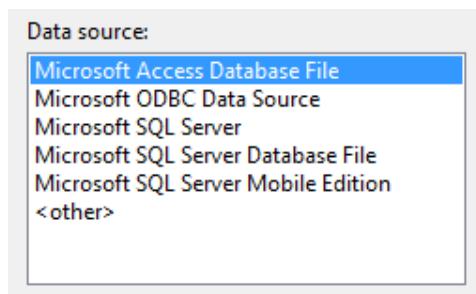
注: リスト ボックスの外観を持つコントロールを、リスト ビューを使用して実装することができます (その逆もあります)。そのような場合は、実装ではなく使用法に基づいてガイドラインを採用してください。

## 使用パターン

リスト ボックスにはいくつかの使用パターンがあります。

### 単一選択リスト

一度に 1 つのアイテムを選択できます。



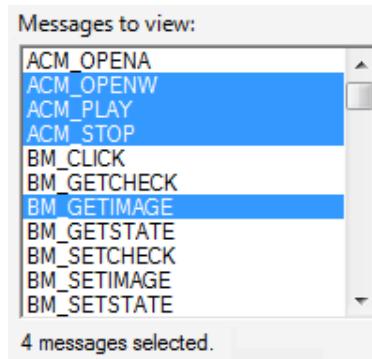
この例では、表示アイテムを 1 つだけ選択できます。

### 標準的な複数選択リスト

0 個を含む任意の個数のアイテムを選択できます。

標準的な複数選択リストの外観は、単一選択リストとまったく同じです。したがって、リスト ボックスで複数選択がサポートされているかどうかを外観から判断することはできません。複数選択が可能かどうかはユーザーが判断する必要があるため、このリスト パターンは、複数選択が必須ではなく、まれに使用されることがあるタスクに使用することをお勧めします。

複数選択モードには、[複数選択モード](#)と[拡張選択モード](#)があります。拡張選択モードは一般的なモードです。選択を拡張するには、値をドラッグするか、連続する値の場合は Shift キーを押しながらクリックし、連続しない値の場合は Ctrl キーを押しながら値をクリックします。複数選択モードでは、任意のアイテムをクリックすると、Shift キーや Ctrl キーを押さなくても選択状態が切り替わります。この一般的ではない動作のため、複数選択モードは推奨されません。代わりにチェック ボックス リストを使用することをお勧めします。



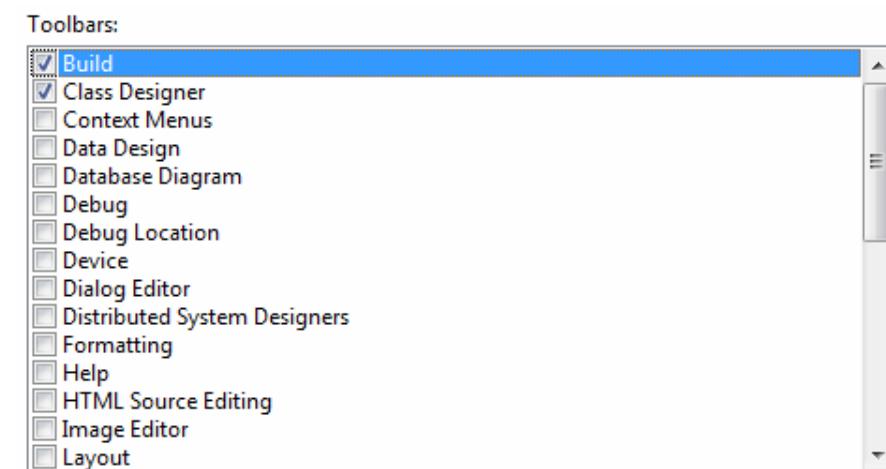
この例では、複数選択モードを使用して任意の個数のアイテムを選択できます。

### チェック ボックスリスト

標準的な複数選択リスト

とは異なり、チェック ボックスでは明らかに複数選択が可能であることがわかります。このリスト パターンは、複数選択が必須であるか、よく複数選択を行う作業で使用します。

チェック ボックスと同様に、  
チェック ボックスリストでは、0 個を含め任意の個数のアイテムを選択できます。



この例では、通常、ユーザーは複数のアイテムを選択するので、チェック ボックス リストが使用され

ています。

このように複数選択を行うことが明示できるという点で、標準的な複数選択リストよりもチェックボックスリストの方が望ましいように見えますが、実際にはほとんどの作業で複数選択は必要ないか、あまり使用しません。このような場合にチェックボックスリストを使用しても、選択することはかりに気を取られます。このような理由で、標準的な複数選択リストの方がずっと一般的になっています。

## プレビューリスト

単一選択または複数選択に使用できます。テキストのみのプレビューではなく、選択の影響がプレビューで示されます。

### Window Color and Appearance

You can change the color of windows, the Start menu, and the taskbar. Pick one of the available colors or create your own color using the color mixer.



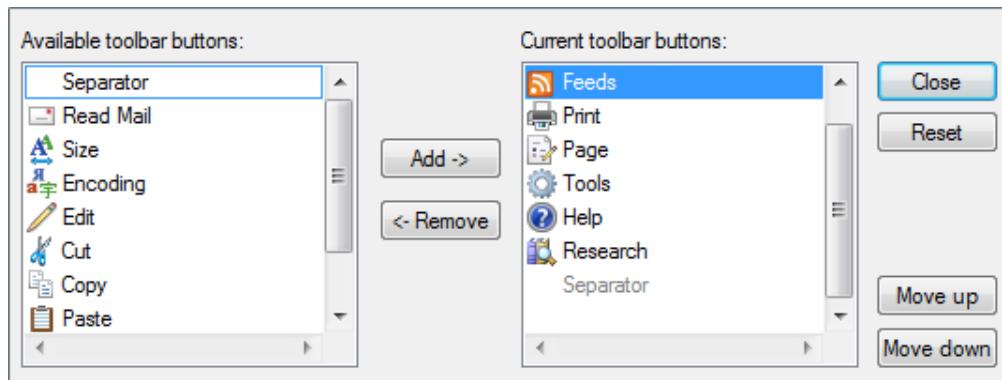
この例では、各オプションのプレビューで選択の影響が明示されています。これはテキストだけを使用するよりも効果的です。

## リストビルダー

一度に1つずつアイテムを追加し、必要に応じて一覧の順序を設定して選択肢のリストを作成できます。

リストビルダーは、2つの単一選択リストで構成されます。左側のリストは固定したオプションセットで、右側のリストは作成中の一覧です。2つのリストの間には、次の2つのコマンドボタンがあります。

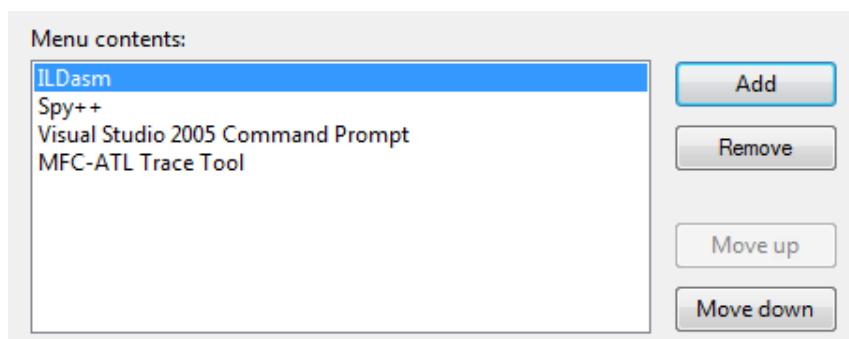
- [追加] ボタン。現在選択されているオプションを、作成中のリストの選択されたアイテムの前に挿入します(オプションアイテムをダブルクリックしても同じ結果になります)。
- [削除] ボタン。作成済みリストで選択されているアイテムを削除し、オプションリストに戻します(作成済みリストのアイテムをダブルクリックしても同じ結果になります)。作成済みリストでは、[上へ] コマンドと [下へ] コマンドを使用してリストアイテムを移動することもできます。



この例では、リストビルダーを使用して、使用可能なオプションセットからアイテムを選択し、順序を設定することでツールバーを作成しています。

追加/削除リスト  
一度に1つ以上のアイテムを追加し、必要に応じて一覧の順序を設定して、リストビルダーのように選択肢のリストを作成できます。

リストビルダーと異なる点は、[追加] をクリックすると、一覧に追加するアイテムを選択するためのダイアログボックスが表示されることです。別々のダイアログボックスを使用することで、アイテムを選択する際の柔軟性が大幅に高まります。特殊なオブジェクトピッカーや一般的なダイアログも使用できます。リストビルダーに比べて、この方法はコンパクトになりますが、アイテムを追加する手間が少し余分にかかります。



この例では、メニューにツールを追加したりメニューから削除することができ、順序を設定することも

できます。

リストビルダーと追加/削除リストのパターンは、その他の複数選択リストよりも使いにくいですが、次の2つの固有の利点があります。

- リストの作成中、作成後に一覧の順序を操作できます。
- 選択したアイテムの概要を確認でき、選択肢の数が多い場合は有益です。

これらのパターンの欠点は、広い画面領域が必要で、膨大なアイテムリストをゼロから作成する場合は使いにくいことです。したがって、小さいリストを作成するか、既に存在するリストを変更するときに使用することをお勧めします。

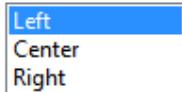
## ガイドライン

### 提示方法

- アイテムを論理的な順序で一覧にします。関連性のあるオプションをまとめてグループにする、最もよく使用されるオプションを先頭にする、またはアルファベット順にするなどの方法をとります。名前はアルファベットまたは五十音順、数字は番号順、日付は時系列に基づいて並べ替えます。12項目以上ある一覧は、項目を見つけやすいようにアルファベット順または五十音順に並べる必要があります。

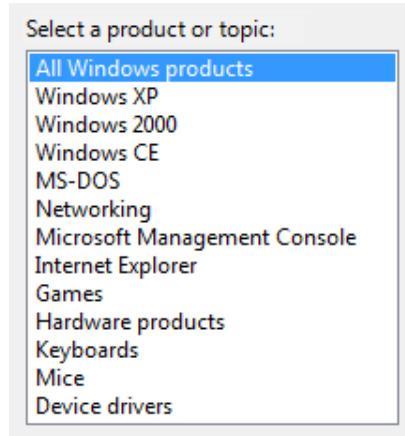
正しい例:

Alignment:



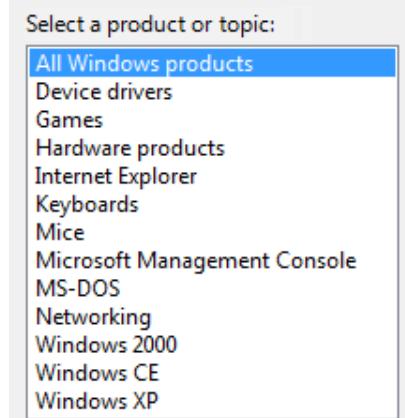
この例では、リストボックスのアイテムが位置関係に基づいて並べられています。

間違った例:



この例では、リストアイテム数が多すぎるので、アルファベット順に並べる必要があります。

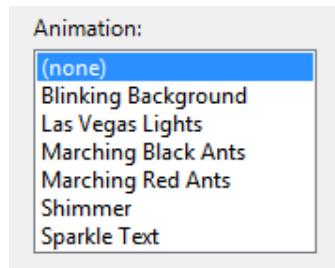
正しい例:



この例では、リストアイテムはアルファベット順に並べ替えられているので、探しやすくなっています。ただし、“すべての Windows 製品”というアイテムは並べ替え順とは関係なく、一覧の先頭に位置します。

一覧の先頭に "すべて" または "なし" を表すオプションを配置します。このオプションは、残りのアイテムの並べ替え順とは無関係です。

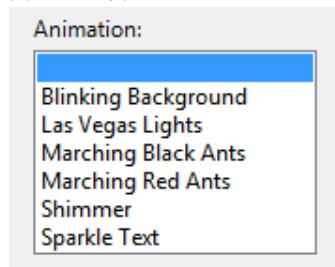
- メタオプションはかつて囲みます。



この例では、`[(なし)]` がメタオプションです。選択肢として有効な値ではなく、オプション自体が使用されていないことを示します。

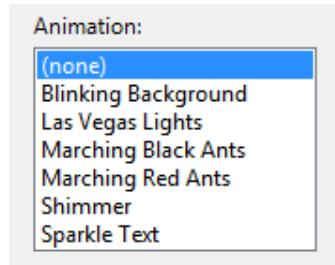
- 一覧には空白のアイテムを使用しないで、代わりにメタオプションを使用します。メタオプションの意味は明示的であるのに対して、空白のアイテムはどのように解釈するのかユーザーにはわかりません。

間違った例:



この例では、空白のオプションの意味が不明確です。

正しい例:



この例では、代わりに `"(なし)"` というメタオプションが使用されています。

## 対話操作

- ダブルクリック時の動作を提供するようにします。ダブルクリックしたときに、アイテムを選択して既定のコマンドを実行したときと同じ結果が得られるようにします。
- ダブルクリック時の動作は他の方法でも実現できる動作にします。同じ結果をもたらすコマンド ボタンやコンテキストメニューのコマンドが常に存在している必要があります。
- 選択したアイテムをユーザーが操作できない場合は、選択できないようにします。

正しい例:

You have successfully completed the Accessibility wizard.  
You made the following changes:

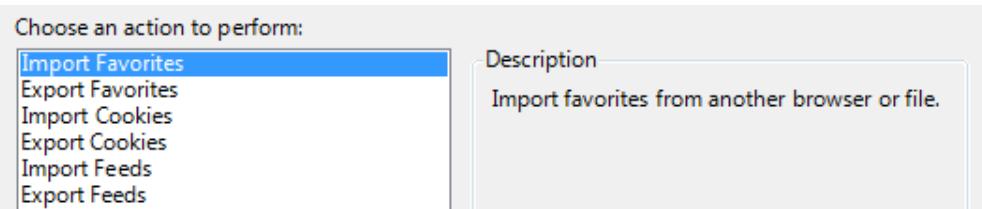


このリスト ボックスでは、変更点を示す読み取り専用の一覧が表示されています。ここでは選択の必要はありません。

- リスト ボックスを無効にする場合は、関連づけられたラベルやコマンド ボタンも無効にします。
- リスト ボックスでのアイテムの変更は、以下の動作に結び付けません。

- コマンドの実行。
- 追加の情報入力を行うためのダイアログ ボックスなど、他のウィンドウの表示。
- 選択されているコントロールに関連する、他のコントロールの動的な表示(このようなイベントは、スクリーンリーダーで検出できません)。例外: 選択したアイテムの説明に使用する静的テキストは動的に変更できます。

許容される例:



この例では、選択されたアイテムを変更すると説明が変更されます。

- 水平方向のスクロールの使用を避けます。複数列リストを使用すると、通常、垂直方向のスクロールよりも使いにくい水平方向のスクロールが必要になります。水平方向のスクロールが必要な複数列リストを使用できるのは、アルファベット順に並べ替えられたアイテムが多数存在し、幅が広いコントロール用に十分な画面スペースが用意されている場合です。

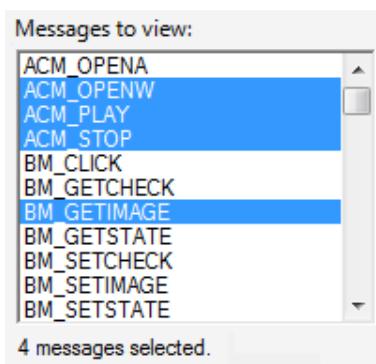
許容される例:

Name	Date modified	Type	Size
3com_dmi	1033	1054	CatRoot
1025	1037	2052	CatRoot2
1028	1041	3076	ccm
1031	1042	appmgmt	ccmsetup

この例では、アイテムが多数存在し、幅が広いコントロール用に十分な画面スペースが用意されているので、水平方向のスクロールが必要な複数列リストが使用されています。

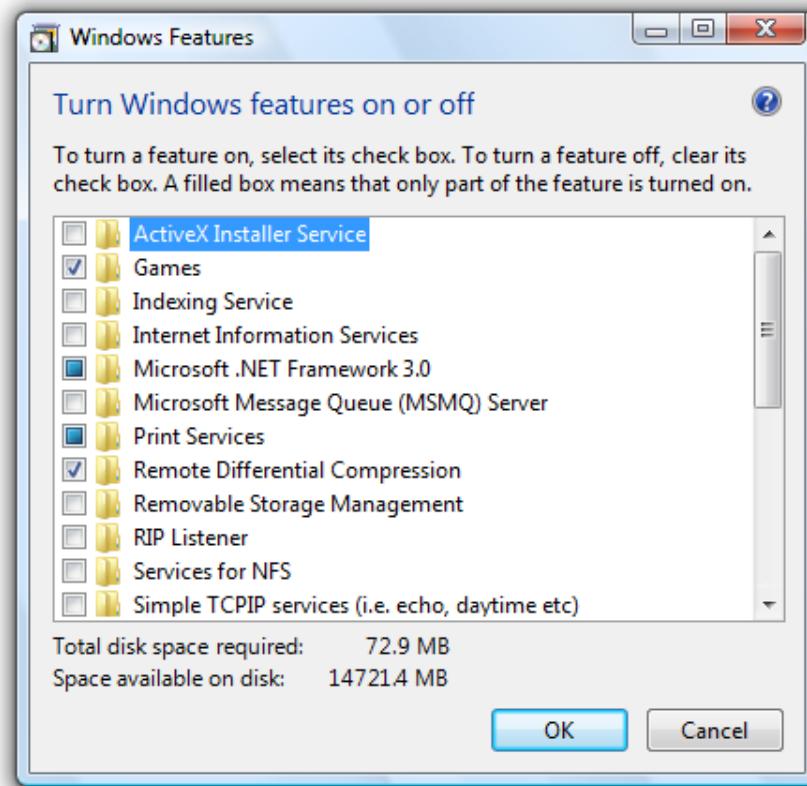
複数選択リスト

- リストの下部に、選択アイテムの数を表示することを検討します。ユーザーが複数のアイテムを選択する傾向が強い場合は特に、この情報は必要です。この情報によって有用なフィードバックが提供されるだけでなく、リスト ボックスで複数選択がサポートされることがはっきりと示されます。



この例では、リストの下に選択アイテムの数が表示されています。

- 他にも、選択に必要なリソースなど、選択アイテム数よりも意味のある、選択に関するデータを表示することができます。



この例では、選択したアイテムの個数よりも、コンポーネントのインストールに必要なディスク領域の方が参考になります。

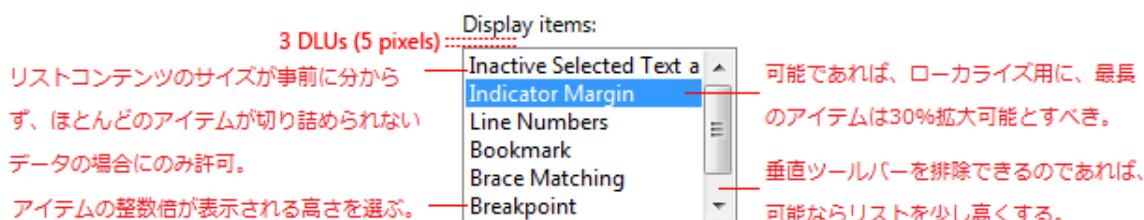
- 潜在的なアイテムの数が多く、一覧のすべてのアイテムをオンまたはオフにする操作の必要性が高い場合は、[すべて選択] および [すべてクリア] のコマンドボタンを追加します。
- 標準的な複数選択リストでは、複数選択モードは非推奨であるため使用しません。代わりに、同じ動作をするチェック ボックス リストを使用します。

#### 既定値

- 最も安全(データの消失やシステムアクセスが失われることを防ぐため)で、最もセキュリティの高いオプションを既定で選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用されるオプションまたは最も便利なオプションを選択します。

例外: コントロールのプロパティが混在状態である場合は、既定値では何も表示されないようにします。混在状態は、設定が同じではない複数のオブジェクトのプロパティを表示するときに発生します。

#### 推奨されるサイズと間隔



#### リスト ボックスに推奨されるサイズと間隔

- 有効な最長データが適切に表示されるリスト ボックス幅を選択します。標準的なリスト ボックスは水平方向にスクロールできないので、ユーザーにはコントロールに表示されている部分しか見えません。
- ローカライズの対象となるすべてのテキスト(数値以外)について、30% (短いテキストの場合は最大 200%) の余白を追加します。
- リスト ボックスの高さを、アイテムの高さの整数倍にし、垂直方向でアイテムが途切れないようにします。
- 一覧の高さは、垂直方向のスクロール操作ができるだけ少なくて済むサイズにします。リスト ボックスには 3 ~ 20 個のアイテムを表示することをお勧めします。リスト ボックスの長さをわずかに長く設定することによってスクロールバーが表示されなくなるのであれば、そうすることを検討してください。多数のアイテムが含まれる可能性のあるリスト ビューの場合は、一度に多数のアイテムを表示するため、またスクロールバーを配置しやすくするた

め、最低 5 個のアイテムを表示します。このようにすることでスクロールも容易になります。

- リスト ボックスを長くすることがユーザーのためになる場合は、リスト ボックスとその親ウィンドウのサイズを変更できるようにします。こうすると、ユーザーはリスト ボックスのサイズを必要に応じて調整できます。ただし、サイズ変更可能なリスト ボックスのアイテム数は 3 個以下にならないようにします。

## ラベル

### コントロール ラベル

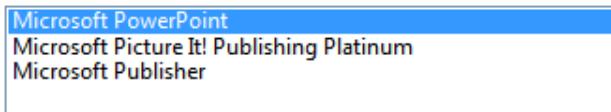
- すべてのリスト ボックスにはラベルが必要です。ラベルは文ではなく語句にして、末尾にコロンは付けません。

例外: ダイアログ ボックスの[メイン指示テキスト](#)の単なる言い直しになる場合は、ラベルを省略します。この場合、メイン指示テキストにコロン(疑問文以外の場合)とアクセスキーを付けます。

許容される例:

[Choose the application to receive the image](#)

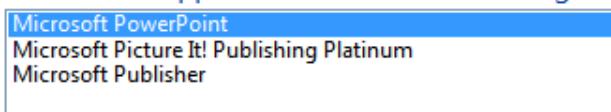
Available applications:



この例では、リスト ボックス ラベルはメイン指示テキストの言い直しです。

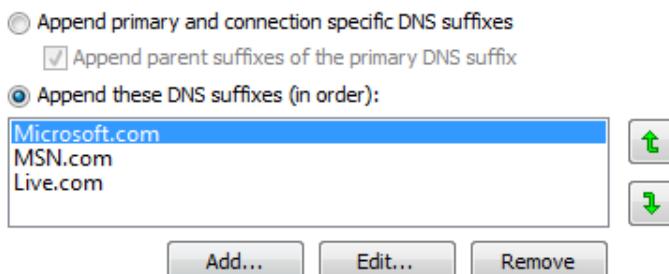
より良い例:

[Choose the application to receive the image:](#)



この例では、冗長なラベルは削除され、メイン指示テキストにコロンとアクセスキーが指定されています。

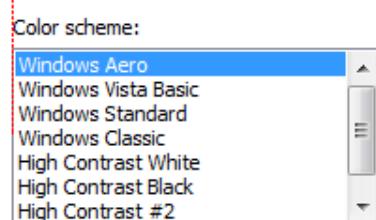
- ラジオ ボタンやチェック ボックスに従属し、コロンで終わるラベルの後に配置されるリスト ボックス コントロールには、追加のラベルを付けません。

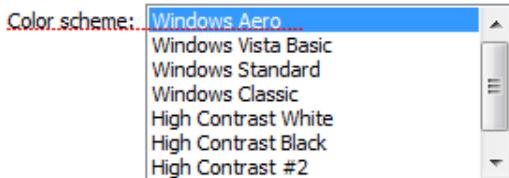


この例では、リスト ボックスはラジオ ボタンに従属していて、ラベルを共有しています。

- 一意な[アクセスキー](#)を割り当てます。ガイドラインについては、「[キーボード](#)」を参照してください。
- [センテンススタイルの大文字化](#)を使用します。
- ラベルはコントロールの左側に配置するか、または上に配置してコントロールの左端と揃えます。
  - ラベルを左側に配置する場合は、ラベルのテキストとコントロール内の一行目のテキストが水平になるようにします。

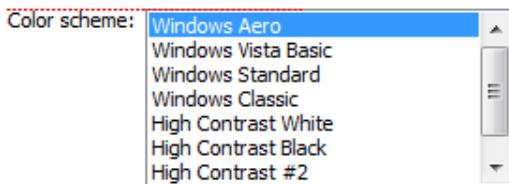
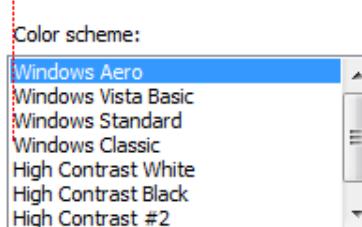
正しい例:





この例では、リストボックスの上にあるラベルはリストボックスの左端に挿えられ、リストボックスの左にあるラベルはリストボックス内のテキストと挿えられています。

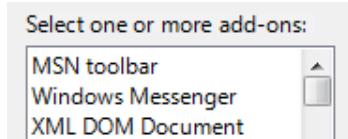
間違った例:



間違った例では、リストボックスの上にあるラベルはリストボックス内のテキストと挿えられ、リストボックスの左にあるラベルはリストボックスの上端に挿えられています。

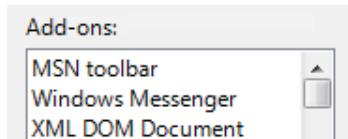
- 複数選択リストボックスには、複数選択が可能であることを明確に示すラベルを使用します。チェックボックスリストのラベルには、特に明記する必要はありません。

正しい例:



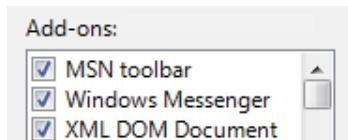
この例では、複数選択が可能なことがラベルに明記されています。

間違った例:



この例では、複数選択についての明確な情報がラベルにありません。

最も良い例:



この例では、複数選択が可能なことがチェックボックスで明示されているため、ラベルに明記する必要はありません。

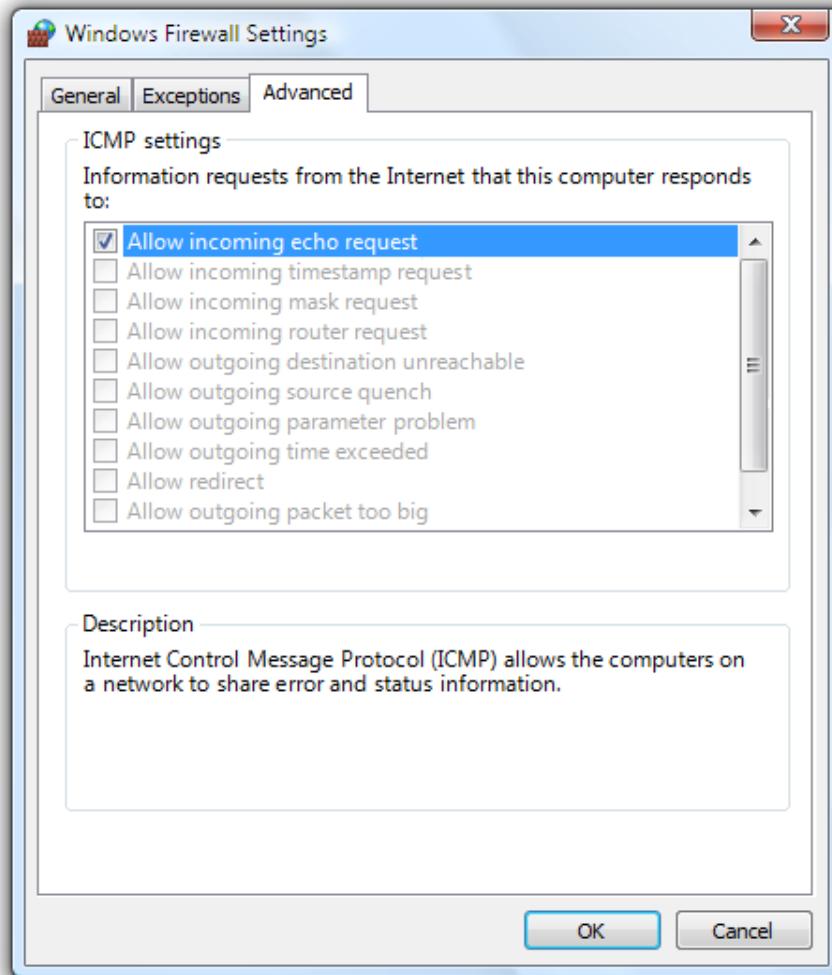
- ラベルの後に単位(秒、接続など)を括って囲んで指定することができます。

## オプション テキスト

- 各オプションに一意な名前を割り当てます。
- アイテムが有名詞である場合を除き、[センテンススタイルの大文字化](#)を使用します。
- ラベルは文ではなく語句にして、末尾に句読点は付けません。
- 同じ文法構造の表現を使用し、すべてのオプションの長さがおおよそ同じになるようにします。

## 指示テキストおよび補足テキスト

- リスト ボックスに関する指示テキストが必要な場合は、ラベルの上に追加します。文を使用し、末尾に句点を付けます。
- [センテンススタイルの大文字化](#)を使用します。
- 必須ではなく参考程度の追加情報は、短くまとめます。このようなテキストはラベルとコロンとの間に配置してかつて囲むか、コントロールの下にかつて囲まずに配置します。



この例では、リストの下に補足テキストが配置されています。

## ドキュメント

リスト ボックスに言及するときは、以下のこと留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、アクセスキーを示すかつてや下線付き文字、およびコロンは含めません。また、内容に応じて「ボックスの一覧」または「ボックス」という語を含めます。リスト ボックスを示すために、「リスト ボックス」または「フィールド」という語は使用しません。
- リスト アイテムに言及するときは、大文字と小文字の区別を含め、アイテムのテキストを正確に引用します。
- プログラマーなどの技術文書では、リスト ボックスを「リスト ボックス」とします。それ以外の場合は、内容に応じて「ボックスの一覧」または「ボックス」という語を使用します。
- ユーザー操作を説明する場合は、「クリック」を使用します。
- ラベルとリスト アイテムは半角の角かつて([ ])で囲み、可能な場合は太字にします。

例: [移動先] ボックスの一覧の [ブックマーク] をクリックします。

## リスト ビュー

適切なコントロールかどうかの判断基準

使用パターン

ガイドライン

提示方法

対話操作

複数選択リスト

ビューの変更

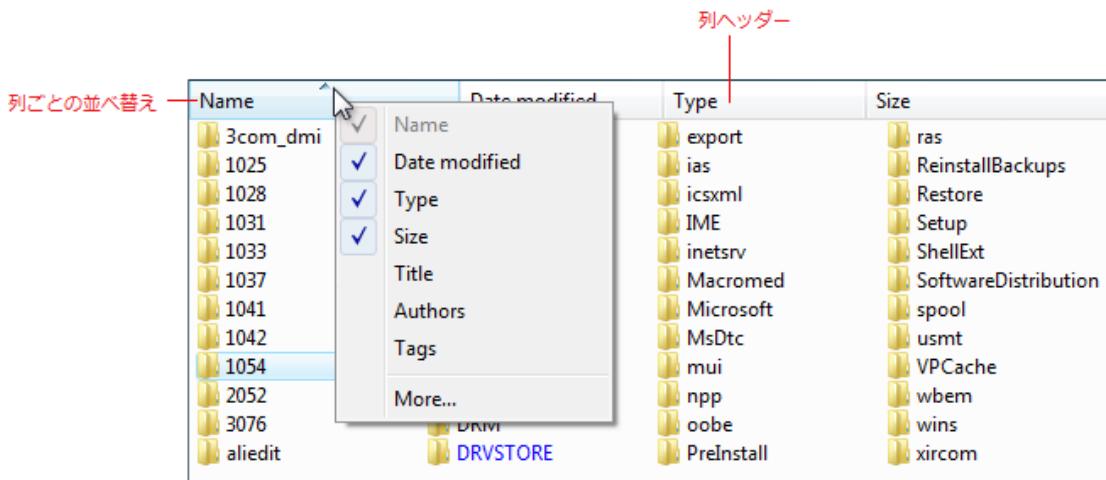
詳細ビュー

推薦されるサイズと間隔

ラベル

ドキュメント

"リスト ビュー" を使用すると、単一選択または複数選択による、データ オブジェクトのコレクションの表示と操作が可能になります。



### 典型的なリスト ビュー

リスト ビューはリスト ボックスよりも柔軟性があり、機能が豊富です。リスト ボックスとは異なり、リスト ビューではビューの変更、グループ化、見出し付きの複数列、列ごとの並べ替え、列の幅と順序の変更、ドラッグ ソースまたはドロップ ターゲットとしての機能、クリップボード経由のデータのコピーがサポートされています。

注: レイアウトおよびリスト ボックスに関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

リスト ビューはリスト ボックスの柔軟性や機能を向上しただけのものではなく、追加の機能により異なる使用方法が可能になっています。次の表は両者を比較したものです。

	リスト ボックス	リスト ビュー
データ型	データとプログラム オプションの両方。	データのみ。
コンテンツ	ラベルのみ。	ラベルおよび補助データ。複数列の場合もあります。
対話操作	選択に使用。	選択に使用。ただし、多くの場合はデータの表示と対話操作のために使用されます。ドラッグ ソースまたはドロップ ターゲットとなることができます。
提示方法	固定。	ビューの変更、グループ化、列ごとの並べ替え、列の幅と順序の変更が可能です。

このコントロールが適切かどうかは、以下のことから判断します。

- 一覧に表示されるものが、プログラム オプションではないデータかどうか。該当しない場合は、代わりにリスト ボックスを使用することを検討します。
- ビューの変更、グループ化、列ごとの並べ替え、列の幅と順序の変更が必要かどうか。該当しない場合は、代わりにリスト ボックスを使用します。
- コントロールが、ドラッグ ソースまたはドロップ ターゲットとして機能する必要があるかどうか。該当する場合は、リスト ビューを使用します。
- リスト アイテムをクリップボード経由でコピーする必要があるかどうか。該当する場合は、リスト ビューを使用します。

### チェック ボックス リスト ビュー

- データ一覧からまったく選択しない、または1個以上のアイテムを選択するためにコントロールを使用するかどうか。1個のアイテムを選択する場合は、代わりに単一選択を使用します。

- 複数選択がタスクに不可欠、または複数選択の使用頻度が高いかどうか。該当する場合は、対象が詳しい知識のあるユーザーではないときは特に、チェック ボックス リスト ビューを使用して、複数選択であることを明確にします。これに該当せず、チェック ボックスを使用すると複数選択が強調される、または画面が煩雑になるときは、標準の複数選択リスト ビューを使用します。
- 複数選択の安定性が重要かどうか。該当する場合は、[チェック ボックス リスト](#)、[リストビルダー](#)、または[追加/削除リスト](#)を使用します。これらのコントロールでは 1 回のクリックで 1 個のアイテムのみが変更されます。標準の複数選択リストを使用すると、意図しない場合でもすべての選択アイテムが簡単にクリアされます。

注: リスト ビューの外観を持つコントロールを、リスト ボックスを使用して実装することができます (その逆もあります)。そのような場合は、実装ではなく使用法に基づいてガイドラインを採用してください。

## 使用パターン

すべてのビューで単一選択と複数選択がサポートされています。単一選択では一度に 1 アイテムを選択できます。複数選択では、選択しない場合も含めて任意の数のアイテムを選択できます。リスト ビューでは[拡張選択モード](#)がサポートされています。このモードでは、ドラッグするか、Shift キーを押しながらクリック (連続したグループを選択する場合)、または Ctrl キーを押しながらクリック (隣接しない値を選択する場合) して、選択内容を拡張できます。リスト ボックスとは異なり、[複数選択モード](#)はサポートされていません。このモードでは、任意のアイテムをクリックすると、Shift キーや Ctrl キーとは関係なく選択状態が切り替わります。

### 標準リスト ビュー

リスト ビュー コントロールでは、以下の 5 つの標準ビューがサポートされています。

**タイル**  
各アイテムは中アイコンで表示され、右側にラベルとオプションの詳細情報が表示されます。

Date taken	Tags	Name	Size	Rating
	Autumn Leaves JPEG Image 269 KB		Creek JPEG Image 258 KB	
	Desert Landscape JPEG Image 223 KB		Dock JPEG Image 309 KB	
	Forest JPEG Image 648 KB		Forest Flowers JPEG Image 125 KB	
	Frangipani Flowers JPEG Image 105 KB		Garden JPEG Image 504 KB	

タイル ビューに中アイコンが表示され、右側にラベルとオプションの詳細情報が表示されています。

**大きいアイコン**  
各アイテムは特大アイコン、大アイコン、または中アイコンで表示され、下にラベルが表示されます。

Date taken	Tags	Name	Size	>>
				
		Desert Landscape		
				
				
		Frangipani Flowers		
				

大きいアイコン ビューに各アイテムが大きいアイコンで表示され、下にラベルが表示されています。

**小さいアイコン**  
各アイテムは小アイコンで表示され、右側にラベルが表示されます。

Date taken	Tags	Name	Size	»
		Creek		
		Dock		
		Forest Flowers		
		Garden		
		Tree		
		Oryx Antelope		
		Waterfall		

小さいアイコンビューに各アイテムが小さいアイコンで表示され、右側にラベルが表示されています。

**一覧**  
各アイテムは小  
さなアイコンで表示  
されます。右側にラ  
ベルが表示され  
ます。

Date taken	Tags	Name	Size	Rating	
		Garden			
		Green Sea Turtle			
		Tree			
		Humpback Whale			
		Oryx Antelope			
		Toco Toucan			
		Waterfall			

一覧モードで各アイテムが小さいアイコンで表示され、右側にラベルが表示されています。

**詳細**  
各アイテムが行  
を占め、表形式  
で表示されま  
す。左端の列に  
はアイテムのオ  
プションのアイ  
コンとラベルが  
表示され、続く  
各列にアイテム  
のプロパティな  
どの追加情報が  
表示されます。

Name	Date taken	Tags	Size	Rating
	6/24/2005 12:22 PM	Sample; Wildlife	113 KB	
	6/22/2005 8:17 PM	Sample; Ocean	310 KB	
	11/4/2005 6:12 PM	Sample; Landscape	270 KB	
	4/30/2005 11:20 AM	Sample; Landscape	259 KB	
	2/12/2004 5:30 PM	Sample; Landscape	224 KB	
	4/25/2005 12:00 AM	Sample; Landscape	649 KB	
	9/3/2005 6:40 PM	Sample; Landscape	752 KB	
	5/27/2005 8:15 AM	Sample; Landscape	281 KB	
	4/26/2005 4:50 PM	Sample; Flowers	126 KB	
	6/2/2005 3:41 PM	Sample; Flowers	106 KB	
	4/9/2004 8:17 AM	Sample; Flowers	505 KB	
	1/17/2005 7:43 AM	Sample; Flowers	207 KB	

詳細ビューに各アイテムが行を占め、表形式で表示されています。

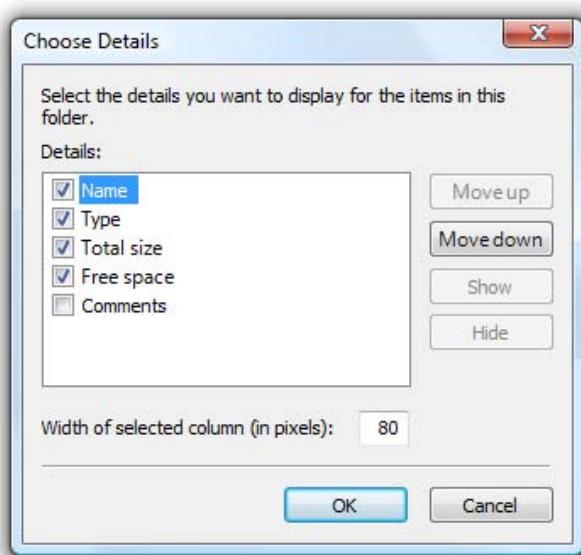
## リスト ビューのバリエーション

**列の選択**  
リスト ビューに  
多数の列が含ま  
れ、全部を表示  
するのが実用的  
ではない場合が  
あります。この  
場合、最も有用  
な列を既定でい  
くつか表示し、  
ユーザーの必要  
に応じて列を追  
加または削除で  
きるようする  
のが最適な方法

The screenshot shows the Windows Explorer context menu open over a folder containing various files and folders. The 'Select columns' option is highlighted. A secondary context menu is displayed, listing the following columns: Name, Date modified, Type, Size, Title, Authors, Tags, and More... The 'Name' and 'Date modified' checkboxes are checked, while the others are unchecked.

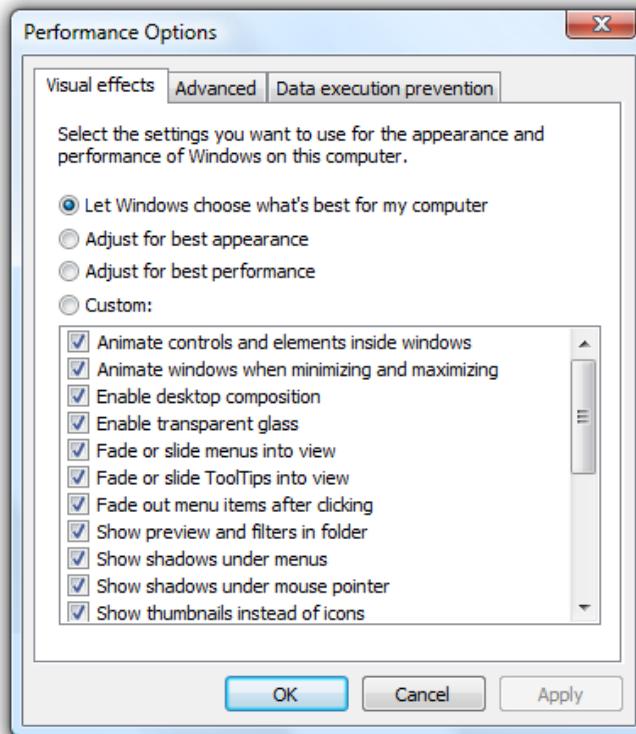
列見出しを右クリックするとコンテキストメニューが表示され、列を追加または削除できます。

です。



列ヘッダー コンテキストメニューの [その他] をクリックすると [詳細表示の設定] ダイアログ ボックスが表示され、列を追加、削除、並べ替えできます。

**チェック ボック** 複数選択リスト ビューの外観は、単一選択リスト ビューとまったく同じです。したがって、複数選択リストがサポートされているかどうかを外観から判断することはできません。チェック ボックス リスト ビューは複数選択が可能であることを明示するために使用できます。このため、このパターンは複数選択アイテムを 択が不可欠であるか、複数選択の使用頻度が高いタスクに使用する必要があります。  
複数アイテムを 選択できます。



この例では、タスクに複数選択が不可欠であるため、小さいアイコンビューでチェック ボックスを使用しています。

**グループ化され** 詳細 ビューでは多くの場合、任意の列の内容でのデータの並べ替えがサポートされますが、リストたリスト ビュー ビューを使用するとアイテムをグループに編成することもできます。以下に、グループ化の利点をいくつか紹介します。

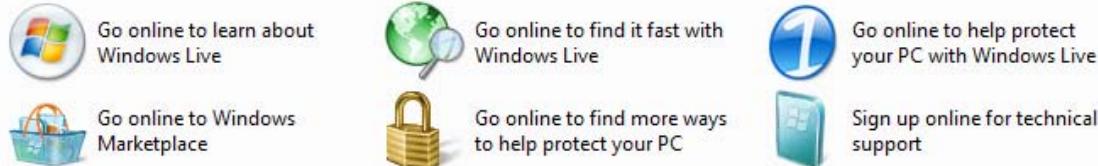
化して整理し  
ます。

- グループはすべてのビュー (一覧モード以外) で機能するため、たとえば、アルバムの特大アイコン ビューをアーティスト別にグループ化できます。
- グループは高水準コレクションにでき、多くの場合、単なるデータのグループ化以上にわかりやすくなります。たとえば、Windows® エクスプローラーでは日付が、[今日]、[昨日]、[先週]、[今年の前半]、[かなり前] にグループ化されます。

## 1. Get started with Windows (14)



## 2. Offers from Microsoft (6)

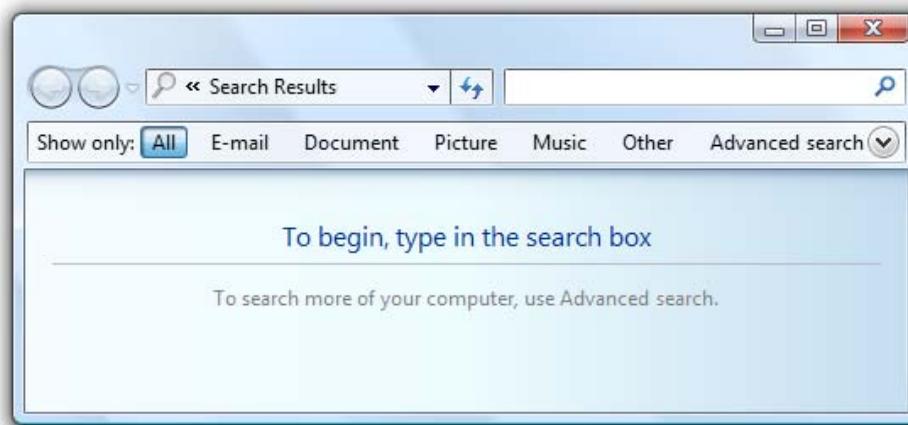


この例では、Windows ウェルカム センターのリスト ビューにアイテムがグループ化されて表示されています。

## ガイドライン

### 提示方法

- リスト アイテムを論理的な順序で並べ替えます。名前はアルファベットまたは五十音順、数字は番号順、日付は時系列に基づいて並べ替えます。
- 適切な場合は、ユーザーが並べ替えの順序を変更できるようにします。リストに多数のアイテムが含まれる場合や、既定以外の並べ替え順でアイテムを検索する方が効果的な場合に、ユーザーによる並べ替えが重要になります。
- Always Show Selection 属性を使用します。この属性を使用すると、コントロールにフォーカスがないときでも、ユーザーは選択アイテムをすぐに判断できます。
- 空のリスト ビューを表示しないようにします。ユーザーがリストを作成する場合は、指示を付けるか、または必要になる可能性のあるサンプル アイテムでリストを初期化します。



この例では、検索リスト ビューに指示が最初から表示されています。

- ユーザーがビューの変更、列の内容による並べ替え、または列の幅と順序の変更を行うことができる場合、次回リスト ビューが表示されたときにもその設定で表示します。リスト ビューの状態は、ウィンドウごと、ユーザーごとに維持されるようにします。

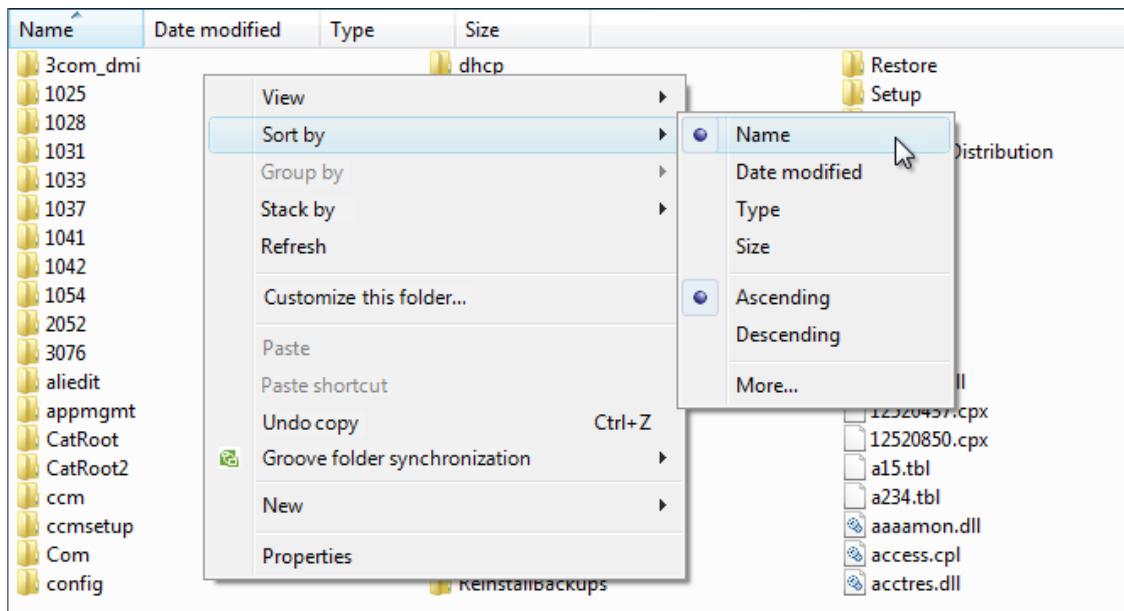
### 対話操作

- ユーザーがポイントしているアイテムはシングルクリックで選択できるようにします。例外: コマンド リンク リストのパターンの場合、シングルクリックでアイテムが選択され、ウィンドウが閉じるか、次ページに移動するようにします。
- ダブルクリック時の動作を提供するようにします。ダブルクリックしたときに、アイテムを選択して既定のコマンドを実行したときと同じ結果が得られるようにします。
- ダブルクリック時の動作は他の方法でも実現できる動作にします。同じ結果をもたらすコマンド ボタンやコンテキストメニューのコマンドが常に存在している必要があります。
- リスト アイテムにさらに説明が必要な場合は、[情報ヒント](#)で説明を提供します。文を使用し、末尾に句点を付けます。

Name	Category
Fonts	Appearance and Personalization
Indexing Options	System and Maintenance
Game Controllers	Hardware and Sound
Internet Options	Network and Internet; Security
iSCSI Initiator	System and Maintenance
Keyboard	Hardware and Sound
M	Customize your keyboard settings, such as the cursor blink rate and the character repeat rate.
M	
Network and Sharing Center	Network and Internet
Offline Files	Network and Internet
Pen and Input Devices	Hardware and Sound; Mobile PC
People Near Me	Network and Internet
Performance Information and Tools	System and Maintenance

この例では、情報ヒントで追加情報を表示しています。

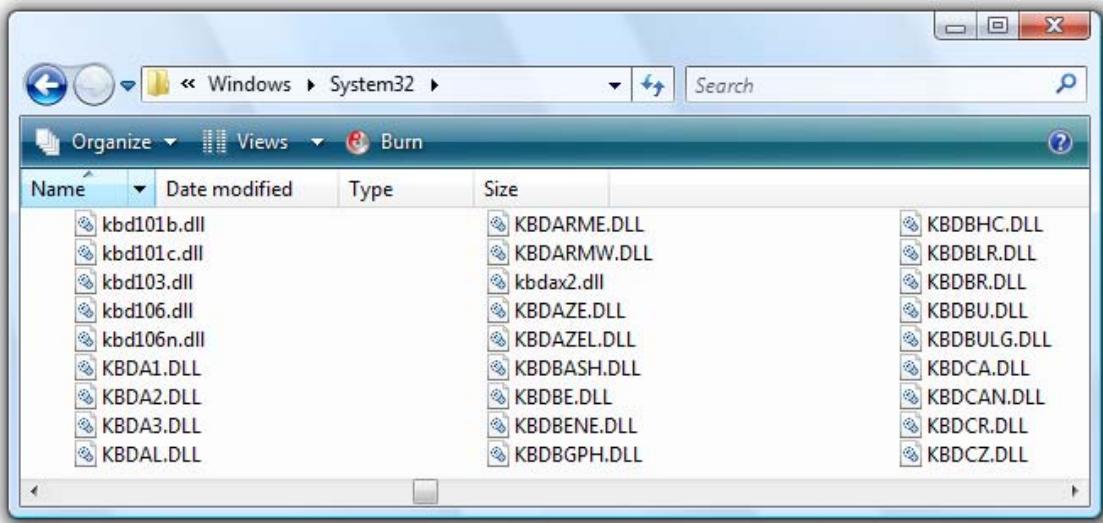
- 関連するコマンドのコンテキストメニューを提供します。関連するコマンドの例としては、[切り取り]、[コピー]、[貼り付け]、[削除]、[名前の変更]、[プロパティ]などがあります。
  - ユーザーに並べ替え順序の変更とグループ化を許可する場合、[並べ替え] および [グループ] のコンテキストメニューを用意します。列名を最初にクリックしたときに、その列の内容でリストに対して昇順の並べ替またはグループ化が行われ、2回目のクリックで降順の並べ替えまたはグループ化が行われます。2次キーとして(別の列の)以前の順序を使用します。



この例では、[並べ替え] コンテキストメニューによって並べ替え順序が変更されます。[名前] を 1 回クリックすると、名前が昇順に並べ替えられます。[名前] を再度クリックすると、名前が降順に並べ替えられます。

- リストビューの列ヘッダーにキーボードからアクセスしやすくなります。
    - 開発者向け情報: 列ヘッダーコントロールにフォーカスを設定することで実現できます。これは Windows Vista® の新機能です。
  - リストビューを無効にするときは、関連付けられたラベルやコマンド ボタンも無効になります。
  - 水平方向のスクロールの使用を避けます。一覧モードでは水平スクロールを使用します。このモードは、通常、最もコンパクトに収めることができます。しかし、一般的に水平スクロールは垂直スクロールよりも使いにくいとされています。コンパクトに収めることが重要ではない場合は、小さいアイコン ビューの使用を検討してください。ただし、アルファベット順に並べ替えられたアイテムが多数あり、幅の広いコントロールに対応できる画面スペースがある場合は、一覧モードが適切です。

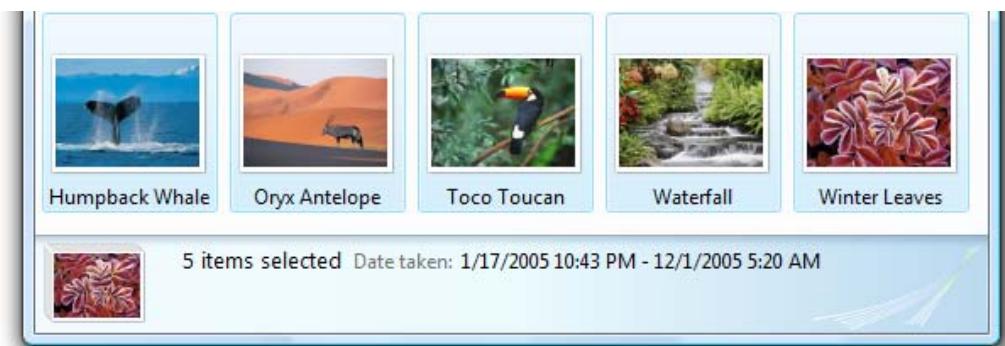
### 許容される例:



この例では、アイテム数が多く、幅の広いコントロールを表示できるスペースがあるので、一覧モードを使用しています。

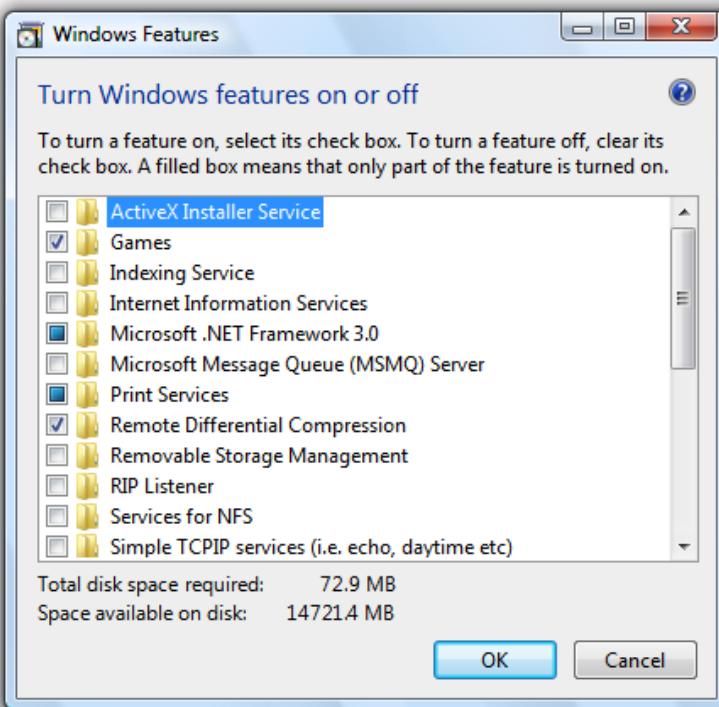
#### 複数選択リスト

- リストの下部に、選択アイテムの数を表示することを検討します。ユーザーが複数のアイテムを選択する傾向が強い場合は特に、この情報は必要です。この情報によって有用なフィードバックが提供されるだけでなく、リストビューで複数選択がサポートされることがはっきりと示されます。



この例では、リストの下に選択アイテムの数が表示されています。

- 選択アイテム数の代わりに、選択に必要なリソースなど、選択アイテム数よりも意味のある、選択に関するデータを表示することもできます。



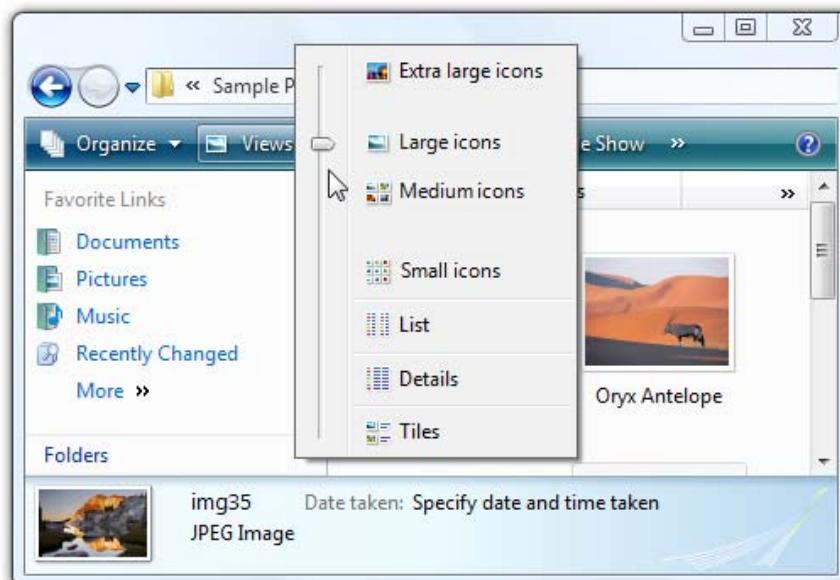
この例では、選択したコンポーネント数よりも、コンポーネントのインストールに必要なディスク空き領域の方が参考になります。

- チェック ボックス リスト ビューについては、潜在的なアイテムの数が多くかつ一覧のすべてのアイテムをオンまたはオフにする操作の必要性が高い場合に、[すべて選択] および [すべてクリア] のコマンド ボタンを追加します。
- コンテナー内のアイテムが一部選択されていることを示すには、混在状態のチェック ボックスを使用します。混在状態は、単独のアイテムの第 3 の状態としては使用しません。

#### ビューの変更

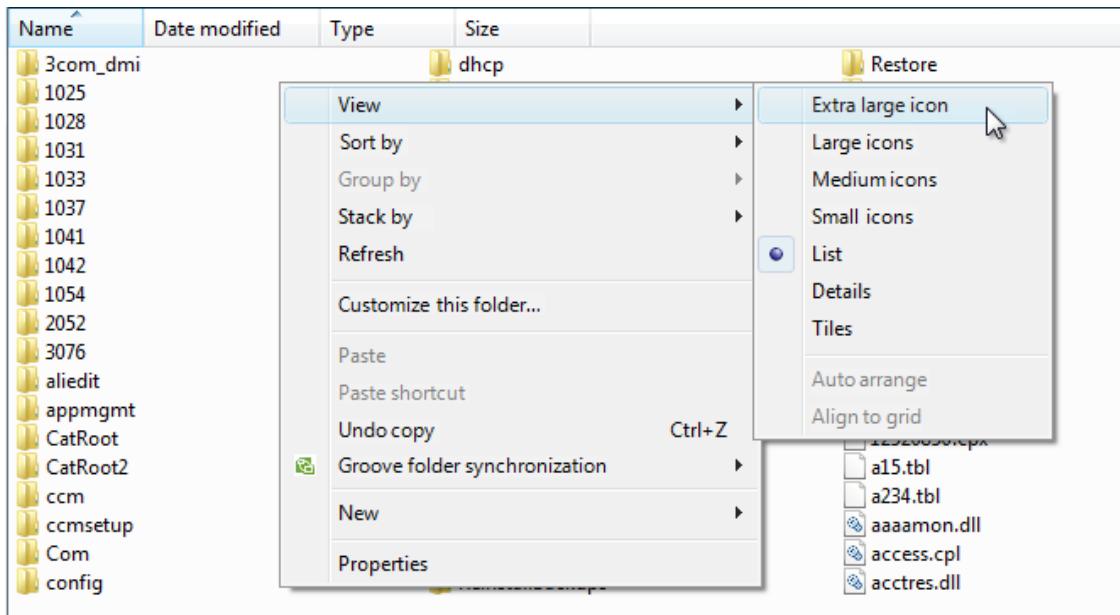
ユーザーがビューを変更できる場合は、次のようにします。

- 最も便利なビューを既定にします。ユーザーがどのような変更を加えても、その変更はリスト ビューごと、ユーザーごとに保持する必要があります。
- **分割ボタン**、**メニュー ボタン**、または**ドロップダウンリスト**を使用してビューを変更します。実用的である場合は常に、ツールバーの分割ボタンを使用するようにし、ボタンのラベルに現在のビューを反映します。



この例では、ビューの変更にツールバーの分割ボタンを使用しています。

- [表示] コンテキストメニューを提供します。



この例では、ビューの変更に [表示] コンテキストメニューが使用されています。

#### 詳細ビュー

- 読みやすさを向上するために、[並べて表示] ビューの使用を検討します。

許容される例:

Issued to	Issued by	Expiration date	Friendly name
Microsoft Corp Ent...	Microsoft Corporate R...	10/12/2008	<None>
Microsoft Corp Ent...	Microsoft Intranet CA	9/30/2007	<None>
Microsoft Corp Ent...	Microsoft Corporate R...	10/12/2008	<None>
Microsoft Corporat...	Microsoft Corporate R...	12/15/2017	<None>
Microsoft Corporat...	Microsoft Corporate R...	2/25/2008	<None>
Microsoft Corporat...	Microsoft Corporate R...	9/20/2019	<None>
Microsoft Intranet CA	Microsoft Corporate R...	5/11/2010	<None>

この例では、データ量の多さと比べて、ウィンドウ、リスト、列のスペースが小さく、リストアイテムが読みにくくなっています。

より良い例:

Issued to	Issued by	Expiration date	»
Microsoft Corp Enterprise CA 2	Microsoft Corporate Root CA	10/12/2008	
Microsoft Corp Enterprise CA 1	Microsoft Corporate Root CA	10/12/2008	
Microsoft Corporate Root Authority	Microsoft Corporate Root Authority	2/25/2008	

この例では、[並べて表示] ビューにデータが切り詰められることなく表示されています。

- 最長のデータに合わせて既定の列幅を選択します。リストビューでは長いデータが自動的に切り詰められて省略記号が表示されるため、既定でほとんど省略記号が表示されない列幅が適切であるといえます。ユーザーが列幅を変更することができますが、以下のような方法での解決をお勧めします。

- 各列の幅を、データに合うサイズに設定します。
- コントロールの幅を、列とスクロールバーを加えた長さに合うサイズに設定します。
- 必要な場合は、水平スクロールを使用します。
- データの切り詰めは普通ではないサイズのアイテムに対して行うか、最後の手段として使用します。

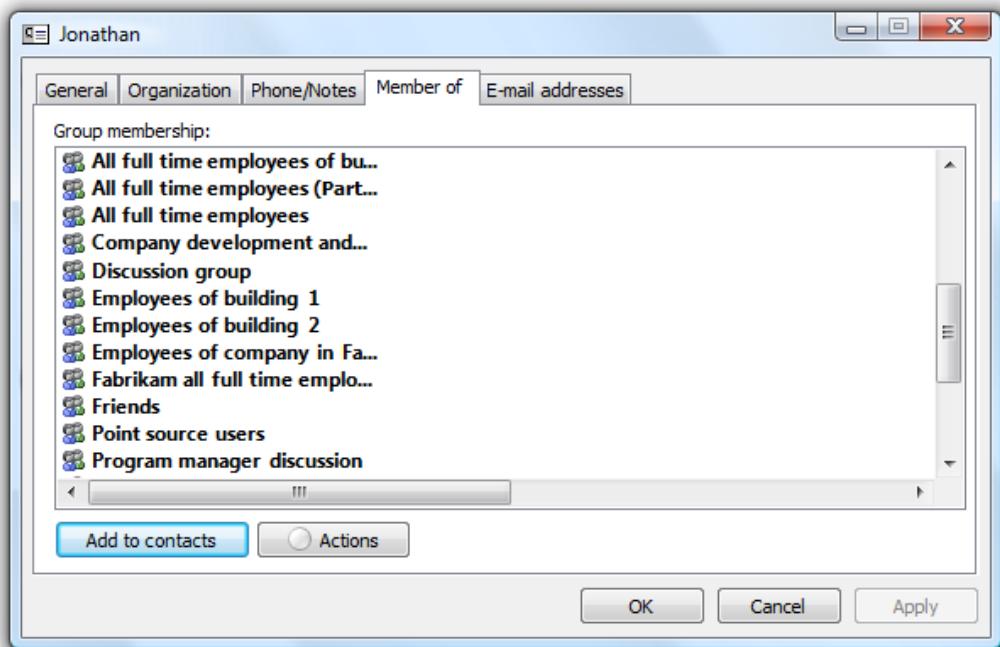
通常サイズのデータを既定で切り詰める必要がある場合は、ウィンドウとリストビューのサイズを変更可能にします。ローカライズの対象となるすべてのテキスト(数値以外)について、30% (短いテキストの場合は最大 200%) の余白を追加します。

間違った例:

Issued to	Issued by	Expiratio...	Friendly name
Microsoft corp ent...	Microsoft corporate r...	10/12/2008	<None>
Microsoft corp ent...	Microsoft intranet ca	9/30/2007	<None>
Microsoft corp ent...	Microsoft corporate r...	10/12/2008	<None>
Microsoft corporat...	Microsoft corporate r...	12/15/2017	<None>
Microsoft corporat...	Microsoft corporate r...	2/25/2008	<None>
Microsoft corporat...	Microsoft corporate r...	9/20/2019	<None>
Microsoft intranet ca	Microsoft corporate r...	5/11/2010	<None>

この例では、ほとんどのデータが切り詰められています。省略記号が多く使用され、コントロールと列の幅がデータと比べて小さいことがわかります。

間違った例:



この例では、理由もなくデータが切り詰められています。

- 適切な列順を既定にします。一般に、次の順序で列を並べます。
  - 最初に、アイテム名またはデータ ID を配置します。
  - 次に、リストアイテムの区別に役立つデータを配置します。
  - 次に、最も有用なデータを配置します(短い、または固定長のデータが望ましい)。
  - 次に、それより有用性の低いデータを配置します(短い、または固定長のデータが望ましい)。
  - 最後に、長く、可変長のデータを配置します。

長く、可変長のデータを最終列に配置し、水平スクロールを使用する頻度を下げます。これらのカテゴリ内に、関連する情報も合わせて論理的な順序で配置します。

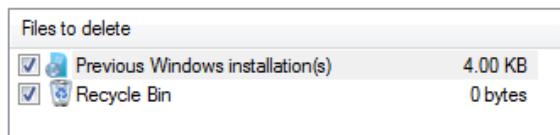
- 妥当であれば、列の追加、削除、順序変更を可能にします。最も有用な列を既定で表示します。これを実現するには、Header Drag Drop 属性を使用します。
- データに応じて配置方法を選択します。次の規則を使用します。
  - 数値、通貨、時刻は右揃えにします。
  - テキスト、ID(数字の場合も含む)、日付は左揃えにします。
- 並べ替え可能な列見出しについては、見出し上の 1 回目のクリックによってその列の内容で一覧を昇順に並べ替え、2 回目のクリックによって降順に並べ替えます。並べ替えの 2 次キーとして(別の列の)以前の並べ替え順序を使用します。

Name	Original l...	Date deleted	Size	Type
contents_files	D:\Users\...	3/29/2007 ...	4 KB	File Folder
Windows Vista User Experience Guideline...	D:\Users\...	3/29/2007 ...	0 KB	File Folder
AutoRecovery save of Document.asd	D:\Users\...	3/23/2007 ...	27 KB	ASD File
AutoRecovery save of Tree Views.asd	D:\Users\...	3/13/2007 ...	693 KB	ASD File
contents	D:\Users\...	3/29/2007 ...	2 KB	HTML Document
header	D:\Users\...	3/29/2007 ...	3 KB	HTML Document
Home	D:\Users\...	3/29/2007 ...	7 KB	HTML Document
Windows Vista User Experience Guidelines	D:\Users\...	3/29/2007 ...	1 KB	HTML Document
additional	D:\Users\...	3/16/2007 ...	74 KB	JPEG Image
Humpback Whale	D:\Users\...	3/21/2007 ...	257 KB	JPEG Image

この例では、最初に [名前] 列がクリックされ、次に [種類] 列がクリックされています。その結果、昇順の [種類] が並べ替えの主キーになり、昇順の [名前] が 2 次キーとなっています。

- 全列で選択アイテムを容易に確認できるように、Full Row Select 属性を使用します。
- データを並べ替えることができない場合は、並べ替え可能な列ヘッダーを使用しないようにします。
- 1列だけが含まれ、逆の順序に並べ替える必要がない場合は、列ヘッダーを使用しないようにします。代わりにラベルを使用してデータを識別します。

間違った例:

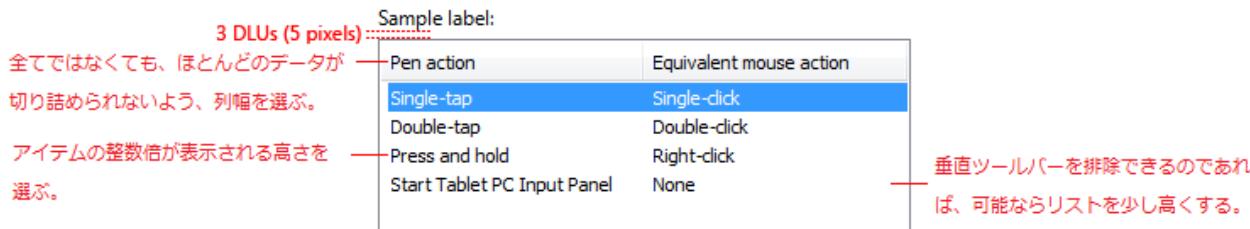


正しい例:



正しい例では、列ヘッダーの代わりにラベルが使用されています。

## 推奨されるサイズと間隔



## リスト ビューに推奨されるサイズと間隔

- リスト ビューの高さを、アイテムの高さの整数倍にし、垂直方向でアイテムが途切れないようにします。
- サポートされるすべてのビューで垂直方向と水平方向の不要なスクロールをなくすようにリスト ビューのサイズを選択します。リスト ビューには 3 ~ 20 個のアイテムを表示することをお勧めします。リスト ビューの長さをわずかに長く設定することによってスクロール バーが表示されなくなるのであれば、そうすることを検討してください。多数のアイテムが含まれる可能性のあるリスト ビューの場合は、一度に多数のアイテムを表示するため、またスクロール バーを配置しやすくするため、最低 5 個のアイテムを表示します。このようにすることでスクロールも容易になります。
- リスト ビューを大きくするとユーザーが使いやすい場合は、リスト ビューとその親 ウィンドウのサイズを変更できるようにします。こうすると、ユーザーはリスト ビューのサイズを必要に応じて調整できます。ただし、サイズ変更可能なリスト ビューのアイテム数は 3 個以下にならないようにします。

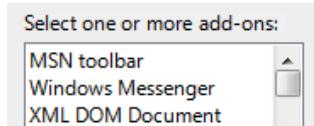
## ラベル

### コントロール ラベル

- すべてのリスト ビューにはラベルが必要です。ラベルは文ではなく語句にして、末尾にコロンを付けます。また、静的テキストにします。
- 一意な [アクセスキー](#) を割り当てます。アクセスキー割り当てのガイドラインについては、「[キーボード](#)」を参照してください。
- [センテンススタイルの大文字化](#)を使用します。
- コントロールの上部にラベルを配置し、ラベルの左端とコントロールの左端を揃えます。
- 複数選択リスト ビューには、複数選択が可能であることを明確に示すラベルを使用します。チェック ボックス リストのラベルに

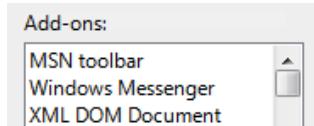
は、特に明記する必要はありません。

正しい例:



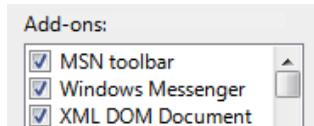
この例では、複数選択が可能なことがラベルに明記されています。

間違った例:



この例では、複数選択についての情報がラベルにありません。

許容される例:



この例では、複数選択が可能なことがチェックボックスで明示されているため、ラベルに明記する必要はありません。

- ラベルの後ろに単位(秒、接続など)をかっこで囲んで指定することができます。

#### 見出しラベル

- 見出しラベルは簡潔にします(3語あるいは9文字以下)。
- 単一の名詞または名詞句を使用し、末尾に句読点は付けません。
- [センテンススタイルの大文字化](#)を使用します。
- 見出しの揃え方はデータと同じにします。

#### グループ ラベル

- 高水準コレクションには、次のようなグループラベルを使用します。
  - [名前]: 名前または文字範囲の先頭文字を使用します。
  - [サイズ]: 未指定、0 KB、0 - 10 KB、10 - 100 KB、100 KB - 1 MB、1 - 16 MB、16 - 128 MB
  - [日付]: 今日、昨日、先週、今年の前半、かなり前
- これら以外については、グループのラベルに、大文字と小文字の区別や句読点を含め、グループ化するデータのテキストを正確に引用します。

#### データ テキスト

- [センテンススタイルの大文字化](#)を使用します。

#### 指示テキスト

- リストビューに関する指示テキストが必要な場合は、ラベルの上に追加します。文を使用し、末尾に句点を付けます。
- [センテンススタイルの大文字化](#)を使用します。
- 必須ではなく参考程度の追加情報は、短くまとめます。このような情報はラベルとコロンとの間に配置してかっこで囲むか、コントロールの下にかっこで囲まずに配置します。

#### ドキュメント

リストビューに言及するときは、以下のことに留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、アクセスキーを示すかっこや下線付き文字、およびコロンは除外し、"の一覧"という語を付け加えます。リストビューを示すために、"リストボックス"、"リストビュー"、または"フィールド"という語は使用しません。
- リストデータについては、大文字と小文字の区別を含め、データテキストを正確に引用します。
- "リストビュー"としてリストビューに言及するのは、プログラミングおよびその他の技術文書内のみとします。それ以外の場合は、"の一覧"を使用します。
- ユーザー操作を説明する場合、データに対しては"選択"を、見出しに対しては"クリック"を使用します。
- ラベルおよび一覧内のアイテムは半角の角かっこ([ ])で囲み、可能な場合は太字にします。

例: [プログラムおよびサービス]の一覧の[ファイルとプリンターの共有]を選択します。

リスト ビュー内のチェック ボックスに言及するときは、以下のことに留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用し、"チェック ボックス" という語を追加します。アクセス キーを示すかっこや下線付き文字は含めません。
- ユーザー操作を説明する場合は、"オンにする" と "オフにする" を使用します。
- ラベルは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例: [下線] チェック ボックスをオンにします。

## 進行状況バー

適切なコントロールかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

全般

確定型の進行状況バー

不確定型の進行状況バー

モードレス進行状況バー

モーダル進行状況バー

残り時間

進行状況バーの色

メーター

推奨されるサイズと間隔

ラベル

"進行状況バー" を使用すると、長い時間を必要とする処理の進行状況をユーザーが確認できます。進行状況バーには、完了した処理のおおよその割合を示すバー(確定型)と、単に処理が進行中であることを示すバー(不確定型)があります。

ユーザビリティに関する調査で、応答時間が1秒を超えると気になりだすという結果が出ています。この理由から、完了まで2秒以上かかる処理を長いものと見なし、何らかの形で進行状況をフィードバックすることを検討する必要があります。



### 典型的な進行状況バー

注: レイアウトに関するガイドラインは、別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- 5秒以内に処理が完了するかどうか。該当する場合は、進行状況バーを表示するには処理時間が短すぎるので、代わりにビジー・ポインターを使用します。ほとんどの場合は5秒以下で完了するが5秒を超えることもある処理に対しては、まずビジー・ポインターを表示し、5秒経過後に進行状況バーに切り替える方法を使用します。
- ユーザーの作業完了を待つ間に不確定型の進行状況バーを使用するかどうか。該当する場合は、ユーザー作業の待機時には、進行状況バーを使用しません。進行状況バーは、コンピューターの処理の進行状況を示すものであり、ユーザーの作業の進行状況を示すものではありません。
- 不確定型の進行状況バーをアニメーションと共に使用するかどうか。該当する場合は、両方を同時に使用せず、アニメーションだけにします。不確定型の進行状況バーは、実質的には単純なアニメーションなので、アニメーションと併用する意味はありません。
- 処理が非常に長い時間のかかる(2分を超える)バックグラウンドタスクで、進行状況よりも完了を知らせる必要があるかどうか。該当する場合は、代わりに通知を使用します。この場合、ユーザーは完了するまで他のタスクを行い、進行状況は監視しません。通知を使用することで、ユーザーは妨げられることなく他のタスクを実行できます。このような長い処理の例としては、印刷、バックアップ、システムスキャン、データの一括転送や一括変換などがあります。
- 処理が完了したときに、ユーザーが結果を再現できるかどうか。該当する場合は、代わりにスライダーを使用します。このような処理の例には、ビデオ録画、オーディオ録音、およびそれらの再生などがあります。



この例では、サウンドの再生中にスライダーで進行状況を示しています。この方法を使用すれば、ユーザーが後で結果を再現できます。

### デザインコンセプト

長い処理の間、ユーザーは行われている処理に関するおおまかな情報を必要とします。また、以下の情

報も要求します。

- ・長時間をする処理が開始された。
- ・処理が進行中であり、最終的には完了する(ロックアップしているのではない)。
- ・完了した処理のおおよその割合(および未完了の割合)。
- ・待ち続けるに値しなければ、取り消すべき処理かどうか。
- ・処理が完了するまで待つのではなく、別の作業を行うことが可能かどうか。

処理時間の長さが有限である場合は、確定型の進行状況バーを使用します。これは、時間を正確に予測できない場合であっても同じです。不確定型の進行状況バーでは、進行中であることが示されますが、その他の情報は示されません。正確に予測できない可能性があるという理由だけで、不確定型の進行状況バーを選択しないでください。

たとえば、処理に5つのステップがあり、それぞれのステップに必要な時間の長さは有限であるとします。ただし、各ステップにかかる時間は大きく変化することがあるものとします。この場合は確定型の進行状況バーを使用し、各ステップで通常かかる時間の比率に基づいて、ステップが完了するごとに進行状況を示します。確定型の進行状況バーでは処理がロックアップしたと誤解されるような場合にのみ、不確定型の進行状況バーを使用します。

#### 最も重要な点

長い処理における進行状況のフィードバックを提供し、上記の情報が明確に伝わるようにします。可能な限り確定型の進行状況バーを使用してください。

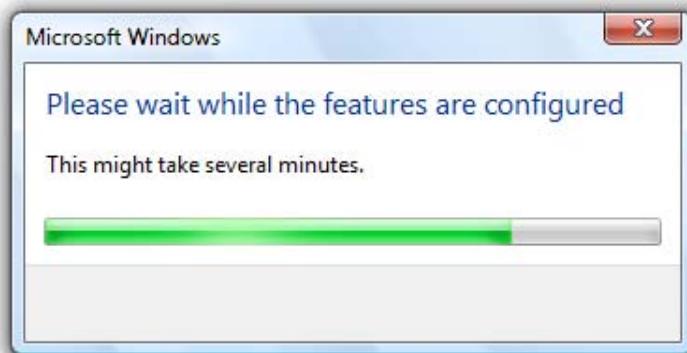
### 使用パターン

進行状況バーにはいくつかの使用パターンがあります。

#### 確定型の進行状況バー

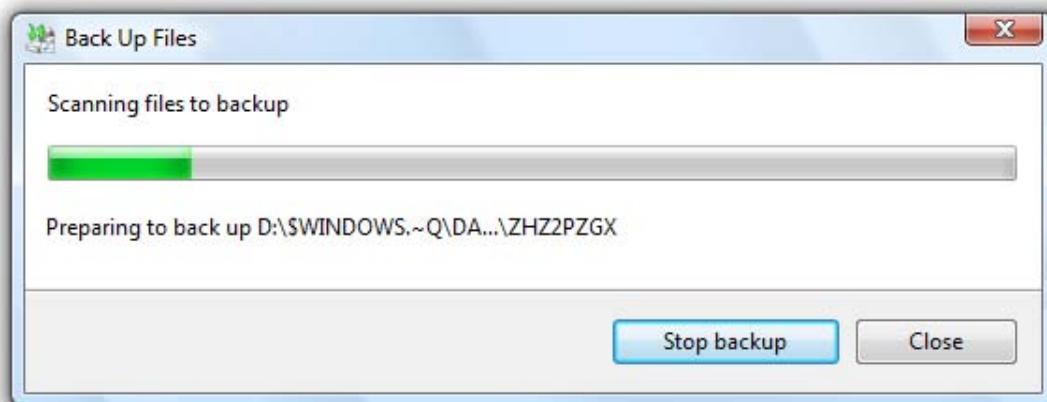
確定型のモダル進行状況バー このフィードバックはモダルであるため、処理が完了するまで、ユーザーはそのウィンドウ内(モダルダイアログボックス内に表示されている場合はその親)で別のタスクを実行できません。

左から右へバーを塗りつぶすことで処理の進行状況を示します。処理が完了するとすべて塗りつぶされます。



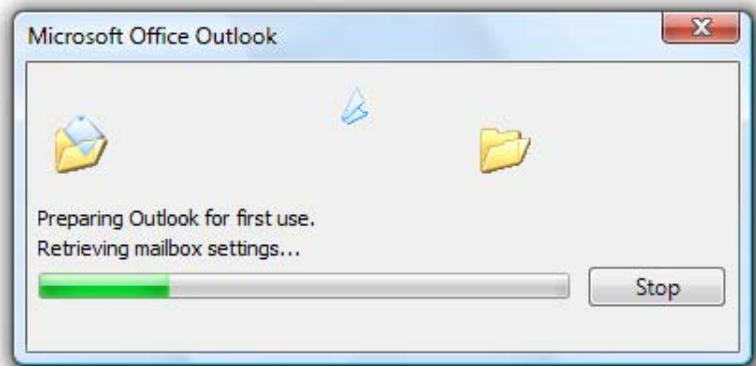
この例では、構成中のフィードバックが進行状況バーにより提供されています。

取り消しや停止のためのボタンが付いた確定型のモダル進行状況バー処理が長すぎる場合や待ち続けるに値しない場合に、ユーザーは処理を停止できます。



この例では、ユーザーは処理を停止するボタンをクリックすることで、環境を現在の状態のままにできます。

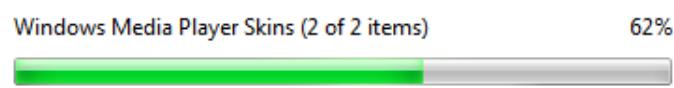
取り消しや停止のためのボタンに加えて、アニメーションが付いた確定型のモーダル進行状況バー。ユーザーが処理を停止できることに加えて、処理の影響を目で確認できます。



この例では、ユーザーは処理を停止するボタンをクリックすることで、環境を現在の状態のままにできます。

**2 本の確定型のモーダル進行状況バー** 上の進行状況バーでは追加情報がほとんど提供されず、またかなり気を散らすものになり得るので、このパターンはお勧めしません。代わりに、処理内のすべてのステップで進行状況を分割し、単一の進行状況バーを1回で完了させます。

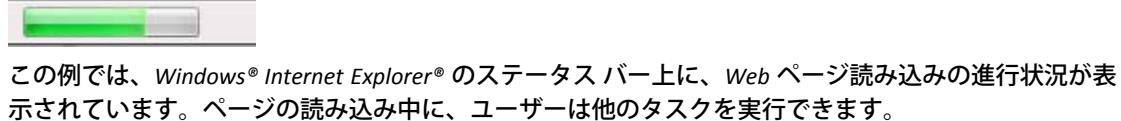
複数ステップを含む処理の進行状況を示します。上の進行状況バーで現在のステップの進行状況を示し、下の進行状況バーで全体の進行状況を示します。



この例では、上の進行状況バーで現在のステップの進行状況を示し、下の進行状況バーで全体の進行状況を示しています。

注: 通常このパターンは不要であり、使用は避けるようにしてください。

**確定型のモードレス進行状況バー** モーダル進行状況バーとは異なり、処理中でもユーザーは他のタスクを実行できます。この進行状況バーは、コンテキスト内またはステータスバー上に表示できます。



この例では、Windows® Internet Explorer® のステータスバー上に、Webページ読み込みの進行状況が表示されています。ページの読み込み中に、ユーザーは他のタスクを実行できます。

左から右へバーを塗りつぶすことで処理の進行状況を示します。処理が完了するとすべて塗りつぶされます。

## 不確定型の進行状況バー

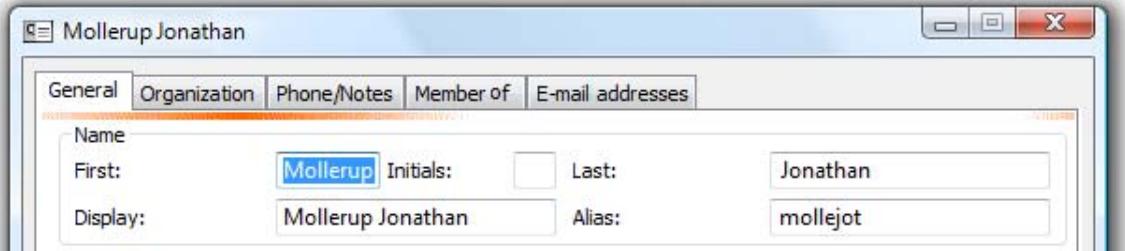
**不確定型のモードレス進行状況バー** 内を左から右へと繰り返し 移動するアニメーションで、処理が進行中であることを示します。



この例では、Windows Updateによる更新プログラムの検索中、処理が進行中であることが不確定型の進行状況バーで示されています。

**不確定型のモードレス進行状況バー** 内を左から右へと繰り返し

移動するアニメーションで、処理が進行中であることを示します。



この例では、Microsoft Outlook® で連絡先のプロパティを入力している間に、不確定型のモードレス進行状況バーが使用されています。この作業の進行中、ユーザーはこのプロパティ ウィンドウの使用を継続できます。

## メーター

### メーター

進行状況には関係しない割合を示します。

このパターンは進行状況バーではありませんが、進行状況バー コントロールを使用して実装されます。実際の進行状況バーと区別するため、メーターの外観は異なります。

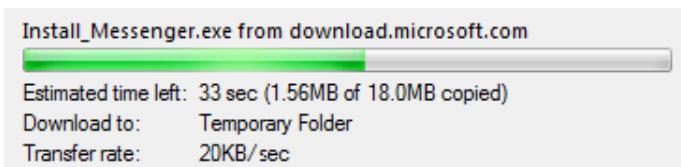


この例では、ディスク ドライブの使用領域の割合がメーターで示されています。

## ガイドライン

### 全般

- 長い処理を実行するときに進行状況をフィードバックします。進行しているかどうかをユーザーが推測する必要がないようにします。
- 実際の進行状況を明確に示します。進行したら、進行状況バーが進む必要があります。予測される所要時間の幅が大きい場合は、バーの長さと時間の尺度を変えて、長時間かかる処理の進行状況を示すことを検討します。プログラムがロックアップしていないときに、ロックアップしているとユーザーが結論付けないように配慮してください。
- 進行していない場合も、それを明確に示します。進行していないければ、進行状況バーが進まないようにする必要があります。決して完了するはずのない処理をユーザーが待ち続けることがないように配慮してください。
- 有用な進行状況の詳細を提供します。進行状況の追加情報を提供します。ただし、ユーザーがその情報に基づく何らかの作業を行うことができる場合に限ります。テキストは、ユーザーが読むのに十分な時間、表示されるようにします。

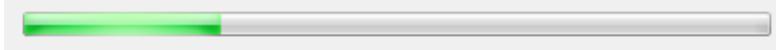


この例では、ユーザーは転送速度を確認できます。ここでは、転送速度が低いことから、より帯域幅の広いネットワーク接続の必要性がわかります。

- 不要な詳細は提供しません。一般に、ユーザーは実行中の処理の詳細を気にしません。たとえば、セットアッププログラムを使用している場合に、どのファイルをコピー中であるとか、システムコンポーネントを登録中であるとかの詳細については、期待する情報ではないのでユーザーは気にしません。通常、適切なラベルが付けられた進行状況バーさえあれば、十分な情報を提供できます。進行状況に関する追加情報は、ユーザーがそれに基づく何らかの作業を行うことができる場合にのみ提供します。ユーザーが気にしない詳細を提供することにより、ユーザーの手順は過剰に複雑でテクニカルなものになります。デバッグのために詳細情報が必要な場合でも、リリース ビルドではそれらを表示しないでください。

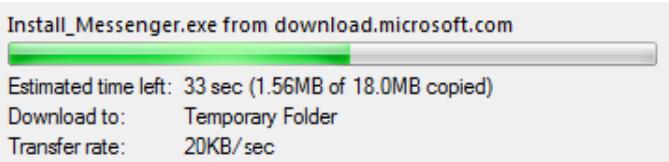
### 正しい例:

Please wait while this wizard installs Windows Live Messenger. This may take a few minutes.



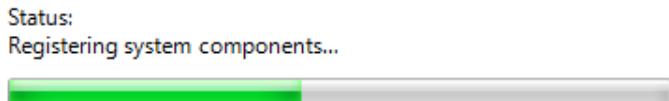
この例では、ラベル付けした進行状況バーで十分です。

### 正しい例:



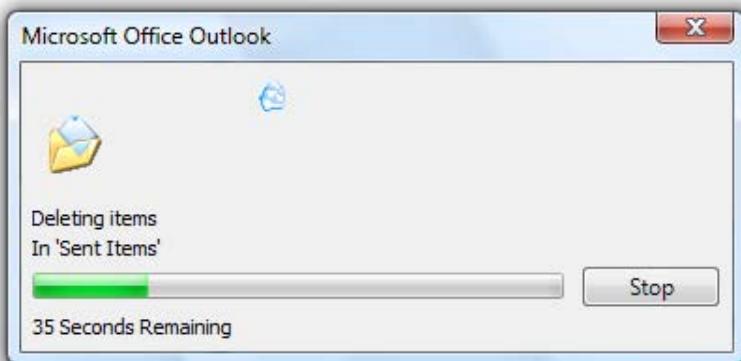
この例では、ユーザーの選択したファイルが Windows エクスプローラでコピーされているので、コピー中のファイル名を表示することは有効です。

#### 間違った例:



この例では、セットアッププログラムにより、ユーザーにとって意味のない詳細が提供されています。

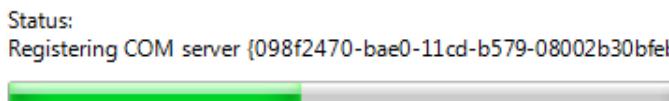
- 有用なアニメーションを提供します。適切なアニメーションを使用すると、処理を目で確認することができるので、ユーザー エクスペリエンスが向上します。優れたアニメーションは、テキストだけの表示よりインパクトがあります。たとえば、Outlook の削除コマンド用の進行状況バーには、ファイルを回復できる場合はごみ箱が表示され、回復できない場合にはごみ箱が表示されません。



この例では、ごみ箱を表示しないことで、ファイルが完全に削除されることを強調しています。この追加情報は、テキストだけではこれほど効果的に伝わりません。

- 不要なアニメーションは使用しません。アニメーションは、通常、実際のタスクとは別のスレッドで実行され、処理がロックアップしても進行中であるものと表示される場合があり、誤解を招くことがあります。また、処理が予想よりも遅い場合に、ユーザーがその一因としてアニメーションを疑うこともあります。したがって、明白な根拠がある場合にのみアニメーションを使用し、ユーザーを楽しませるためには使用しないようにします。
- アニメーションは、進行状況バーの上方中央に配置します。進行状況バーにラベルがある場合は、アニメーションをその上方に配置します。進行状況バーの右側に取り消しや停止のためのボタンがある場合は、それらのボタンも含めて中央を決めます。
- 非常に長く(2分以上)、頻度が少なく、かつ重要な処理の場合にのみ、完了時に音を鳴らします。重要な処理が行われている間に、ユーザーが別のことについている可能性がある場合、サウンド効果でユーザーの注意を引くことができます。これ以外の状況での完了時のサウンド効果は、耳障りでわずらわしいものになります。
- 進行状況の更新や処理の完了を示すために入力フォーカスを取得しないでください。ユーザーは待機中に他のプログラムに切り替えていることが多く、操作の中止は望んでいません。バックグラウンドタスクをフォアグラウンドに移動しないでください。
- テクニカルサポートへの配慮は不要です。進行状況バーで提供されるフィードバックは、必ずしも正確ではなく、またすぐに消え去ります。そのため、テクニカルサポート用に情報を提供する優れたメカニズムとはいません。したがって、セットアッププログラムなどで処理が失敗する可能性のある場合も、テクニカルサポートのみに有用な進行状況の追加情報は提供しないでください。代わりに、ログ ファイルなどの別のメカニズムを使用してテクニカルサポート情報を記録します。

#### 間違った例:



この例では、テクニカルサポート用の詳細情報が進行状況バーに示されています。

- 進行状況バー上には、完了率のパーセントやその他のどのようなテキストも配置しません。このようなテキストは判読できず、テーマを使用した際に互換性がなくなります。

間違った例:



この例では、進行状況バー上にパーセントを示すテキストが表示されていますが、これは判読できません。

- 進行状況バーとビジー ポインターを組み合わせて使用しません。どちらかのみを使用し、両方を同時に使用しないでください。
- 縦型の進行状況バーは使用しません。横型の進行状況バーの方が、配置と流れが自然です。

#### 確定型の進行状況バー

- 処理時間の長さが有限である場合は、確定型の進行状況バーを使用します。これは、時間を正確に予測できない場合であっても同じです。不確定型の進行状況バーでは、進行中であることが示されますが、その他の情報は示されません。正確に予測できない可能性があるという理由だけで、不確定型の進行状況バーを選択しないでください。
- 進行状況の段階を明確に示します。進行状況バーは、処理が初期、中間、または終期のどの段階にあるかを明示できる必要があります。たとえば、進行状況バーが最初から 99% 完了を示し、それから長い時間そのままの状態であれば、それには情報価値がなく単にわざらわしいだけです。このような場合は、最初の進行が最大でも 33% にとどまるようにし、まだ初期段階であることを示します。
- 完了を明確に示します。処理が完了したのではない限り、進行状況バーが 100% に到達しないようにします。
- 残り時間を正確に見積もることができますのであれば、それを提供します。正確に見積もられた残り時間は有用ですが、見積もりがまったくの見当違いであったり、頻繁に大きく変わるのは役に立ちません。正確な見積もりを提供するために、何らかの処理を実行する必要がある場合は、該当する場合は、処理の初期段階では、不正確な可能性のある見積もりを表示しないようにします。
- バーの進行を再開させないでください。おそらく処理の 1 つのステップが完了したという理由だと思いますが、バーの進行を最初から再開させると、進行状況バーの価値は失われます。これでは、ユーザーは処理がいつ完了するのかわかりません。代わりに、処理内のすべてのステップで進行状況を分割し、進行状況バーを 1 回で完了させます。

間違った例:



この例では、ファイルコピーのステップに処理が移行した時点で、そのステップ用に進行状況バーがリセットされています。これでは、どの程度進行したのか、残り時間はどのくらいかなど、ユーザーにはわからなくなります。

- バーの進行を後退させないでください。バーの進行が後退すると、再開したときと同じようにその価値は失われます。進行状況バーは 1 方向だけに進ませます。ただし、進行速度は変化があるので、残り時間の増減は可能です。

#### 不確定型の進行状況バー

- 進行状況の開始から終了までを確定できない処理の場合にのみ、不確定型の進行状況バーを使用します。処理時間の長さが有限でない場合や、処理でアクセスするオブジェクト数が不明の場合は、不確定型の進行状況バーを使用します。時間ベースの処理で制限時間を設けるには、タイムアウトを使用します。
- 進行状況の開始から終了までを確定できるようになったら、確定型の進行状況バーに切り替えます。たとえば、オブジェクト数を特定するのに 2 秒をはるかに超える時間がかかる場合、オブジェクト数を数えている間は不確定型の進行状況バーを使用し、後で確定型の進行状況バーに切り替えることができます。
- 不確定型の進行状況バーには、完了率のパーセントや推定残り時間を表示しません。これらの情報を提供できるのであれば、代わりに確定型の進行状況バーを使用します。
- 不確定型の進行状況バーはアニメーションと組み合わせて使用しません。不確定型の進行状況バーは、実質的には単

純なアニメーションなので、両方ではなくどちらか一方を使用します。

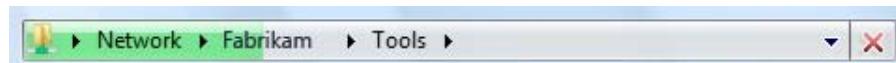
正しい例:



この例では、アニメーションだけを使用して、処理中であることを示しています。

#### モードレス進行状況バー

- 処理の進行中に、ユーザーが何か生産的なことを実行できる場合は、モードレスフィードバックを提供します。場合によっては、進行中の処理の完了が要件となる一部機能を無効化する必要があります。
- ウインドウにアドレスバーがある場合は、モードレスな進行状況をアドレスバーに表示します。



この例では、モードレスな進行状況がアドレスバーに表示されています。

- アドレスバーがなくても、ステータスバーがある場合は、モードレスな進行状況をステータスバーに表示します。ステータスバー内の進行状況の左側に、対応するテキストを配置します。



この例では、モードレスな進行状況がステータスバーに表示されています。

#### モーダル進行状況バー

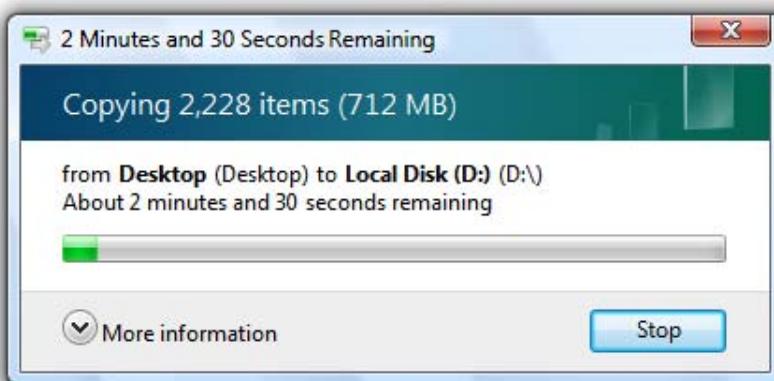
- モーダル進行状況バーは、進行状況ページまたは[進行状況ダイアログ ボックス](#)に配置します。
- 完了まで数秒以上かかる処理、または完了しない可能性のある処理の場合は、処理を停止するためのコマンドボタンを提供します。取り消すことで、何の副次的な影響もなしに環境が以前の状態に戻される場合は、ボタンに"キャンセル"というラベルを付けます。これ以外の場合は、部分的に処理が完了した状態のままになることを示すために、ボタンに"停止"というラベルを付けます。処理のある時点で環境を元の状態に戻すことが不可能になる場合は、処理の途中でボタンのラベルを"キャンセル"から"停止"に変更できます。コマンドボタンと進行状況バーの上下位置は上揃えにするのではなく、中央揃えにします。

正しい例:



この例では、ネットワーク接続の停止に副次的な影響がないので、"キャンセル"が使用されています。

正しい例:



この例では、コピーの停止でコピー後のファイルが残るので、コマンド ボタンには "停止" が使用されています。

間違った例:



この例では、検索の停止に副次的な影響はないので、コマンド ボタンのラベルは "キャンセル" にする必要があります。

残り時間

確定型の進行状況バーの場合:

- 以下の時刻形式を使用します。以下のうち最大時間単位がゼロでない形式から始めて、最大時間単位がゼロになった時点で次の形式に移行します。

進行状況バーの場合:

関連情報をコロン形式で表示する場合:

残り時間: h 時間 m 分

残り時間: m 分 s 秒

残り時間: s 秒

画面領域を優先する場合:

残り h 時間 m 分

残り m 分 s 秒

残り s 秒

その他の場合:

残り h 時間 m 分です

残り m 分 s 秒です

残り s 秒です

タイトルバーの場合:

hh:mm 残り

mm:ss 残り

0:ss 残り

このコンパクトな形式で重要な情報を最初に示すことで、タスク バーで切り捨てられないようにします。

- 見積もりは正確に行いますが、正確に見せかけた情報は提示しません。最大単位が時間の場合は、有用な場合に分を提示しますが、秒は提示しません。

間違った例:

hh 時間 mm 分 ss 秒

- 見積もりを最新の状態に保ちます。残り時間の見積もりは、少なくとも 5 秒ごとに更新します。
- 残り時間に情報を集中します。これこそユーザーが最も気にする情報です。合計経過時間は、タスクが繰り返されそうなときなどの、経過時間が役立つシナリオのみで提示します。残り時間の見積もりが進行状況バーと共に表示される場合は、完了率のパーセントは表示しません。この情報は進行状況バー自体で表現されます。
- 文法的に正しくします。数値が 1 の場合は单数形を使用します (英語の場合)。

間違った例:

1 minutes, 1 seconds

- センテンス スタイルの大文字化を使用します。

進行状況バーの色

- 赤または黄色の進行状況バーは、タスクの最終結果ではなく、進行のステータスを示すためにのみ使用します。赤または黄色の進行状況バーは、タスク完了のためにユーザーが何らかのアクションを起こすことが必要であることを示します。状況を回復できない場合は、進行状況バーを緑のままにしてエラーメッセージを表示します。
- 進行を妨げている "ユーザーが回復可能な" 状況がある場合は、進行状況バーを赤に変えます。メッセージを表示

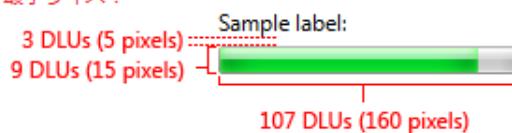
- し、問題を説明して、推奨される解決策を提示します。
- ユーザーがタスクを一時停止した場合、または進行を遅らせている(完全には止まっていない)状況が存在する場合は、進行状況バーを黄色に変えます。この状況には、ネットワーク接続の品質が低い場合などが含まれます。ユーザーが一時停止した場合は、"一時停止"ボタンのラベルを"再開"に変更します。進行が遅くなっている場合は、メッセージを表示し、問題を説明して、推奨される解決策を提示します。

## メーター

- 進行状況バーは、進行状況を示すためにのみ使用します。進行には関係しない割合を示すにはメーターを使用します。

## 推奨されるサイズと間隔

**最小サイズ:**



**最大サイズ:**



## 進行状況バーに推奨されるサイズと間隔

- 進行状況バーの高さは常に推奨値に設定します。
  - 例外: 親ウィンドウが推奨される高さをサポートしない場合は、異なる高さを使用できます。
- 進行状況バーをあまり目立たなくするには、最小幅を適用します。
- 推奨される最大幅よりも長くしないでください。利用できる領域全体を進行状況バーで埋める必要はありません。
- ウィンドウが進行状況バーの最大推奨幅よりもかなり広い場合は、進行状況バーの左右位置を中央揃えにします。

## ラベル

### 進行状況バーのラベル

- 静的テキストコントロールを使用して簡潔なラベルを付け、何の処理が行われているのかを示します。ラベルは実行中の処理を示す語句から始めて、末尾に省略記号を付けます("コピーしています..."など)。複数のステップがある処理、または複数のオブジェクトを処理する場合に、このラベルを動的に変更できます。
- このコントロールは対話型ではないので、固有の[アクセスキー](#)は割り当てません。
- センテンススタイルの大文字化**を使用します。
- ユーザーが直接開始した処理ではない場合は、割り込みが発生した背景を説明してお詫びの言葉を添えるラベルを追加できます。この追加ラベルは、"お待ちください"という句から始めます。このラベルは、処理中は変えないようにします。



Please wait while Windows connects to the "Microsoft" network.



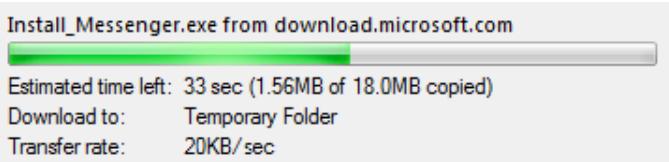
この例では、ユーザーが処理を直接開始していないので、ユーザーに待つようにお願いしています。

- このラベルは進行状況バーの上方に配置し、ラベルの左端と進行状況バーの左端を揃えます。

## 進行状況バーの詳細

- コロンで終わるラベルに続けて、静的テキストで詳細情報を提供します。詳細テキストの後に単位(秒、キロバイトなど)を指定します。

正しい例:



この例では、詳細情報に適切なラベルが付けられています。

間違った例:



この例では、詳細情報にラベルが付けられていないので、ユーザーがその意味を判断する必要があります。

- ・センテンススタイルの大文字化を使用します。
- ・この詳細情報は進行状況バーの下方に配置し、ラベルの左端と進行状況バーの左端を揃えます。
- ・処理の完了または残りの割合を示すパーセント値は表示しません。これらの情報は進行状況バー自体で表現されます。

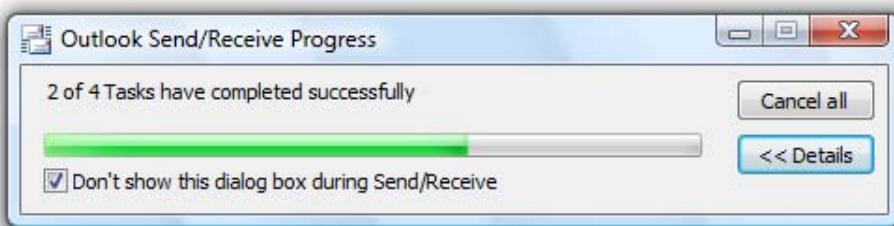
#### [キャンセル] ボタン

- ・取り消すことで、何の副次的な影響もなしに環境が以前の状態に戻される場合は、ボタンに "キャンセル" というラベルを付けます。これ以外の場合は、部分的に処理が完了した状態のままになることを示すために、ボタンに "停止" というラベルを付けます。
- ・処理のある時点で環境を元の状態に戻すことが不可能になる場合は、処理の途中でボタンのラベルを "キャンセル" から "停止" に変更できます。

#### 進行状況ダイアログ ボックスのタイトル

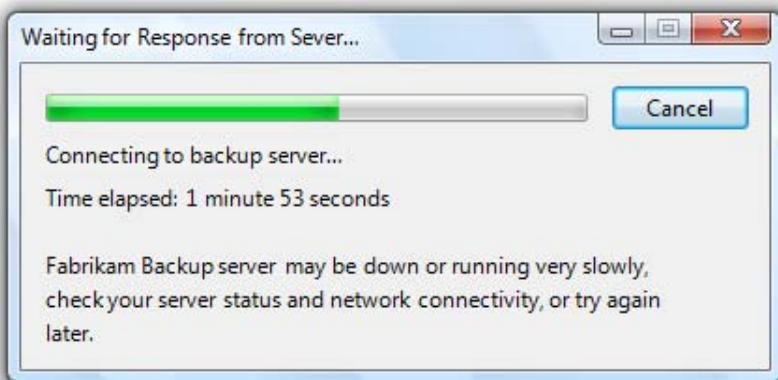
- ・モーダルダイアログ ボックスに進行状況バーを表示する場合は、ダイアログ ボックスのタイトルはプログラムの名前または処理の名前にします。進行状況バーのラベルに使用する内容は、ダイアログ ボックスのタイトルには使用しません。

正しい例:



この例では、タスク名がダイアログ ボックスのタイトルに使用されています。

間違った例:



この例では、ダイアログ ボックスのタイトルのテキストが、進行状況バーのラベルの言い換えになっています。代わりにプログラム名を使用する必要があります。

- モードレス ダイアログ ボックスに進行状況バーを表示する場合は、タスク バーでの表示を最適にするため、特徴をなす情報をタイトルの最初の方に簡潔にまとめます。例: "66% 完了。"

## 段階的表示コントロール

適切なコントロールかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

全般

対話操作

提示方法

シェブロン

矢印

推奨されるサイズと間隔

ラベル

ドキュメント

"段階的表示コントロール" を使用すると、データ、オプション、コマンドなどの追加情報の表示/非表示をユーザーが制御できるようになります。段階的表示を使用することで、本質的要素を中心に置きながら、簡単な操作で必要に応じて追加の詳細を表示できます。



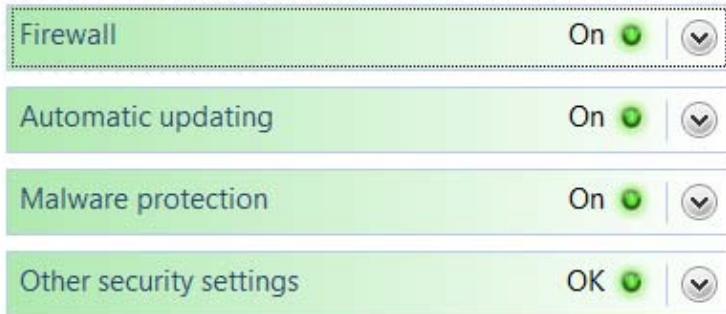
段階的表示コントロールの例。

注: レイアウト、メニュー、ツールバーに関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- その情報が常に必要とされるものではなく、状況または手順に応じて必要とされるものであるかどうか。該当する場合は、段階的表示を使用して情報を表示できるようにすると、基本的なユーザーの手順を単純化しつつ、情報へのアクセスも簡単になります。



この例では、セキュリティセンターに重要なセキュリティ状態が常時表示されていますが、段階的表示を使用して必要に応じて詳細を確認できます。

- 既定で表示される情報を、ユーザーが非表示にする選択を行う可能性があるかどうか。ユーザーがより広い領域を必要とするシナリオがあるか、ユーザーインターフェイス (UI) のカスタマイズを希望する可能性が高いかどうかを確認します。該当しない場合は、段階的表示を使用せずに情報を表示します。

間違った例:



この例では、ユーザーは情報の非表示を希望することはありません。

- 追加情報が高度な内容、情報量の多い内容、複雑な内容、または独立したサブタスクに関連する内容であるかどうか。該当する場合は、コマンドボタンやリンクを使用し、別のウィンドウで表示することを検討します。詳しい知識のあるユーザー向けの追加情報は、高度な情報と見なします。こういった情報で他の情報が読みにくく

なったり配置しにくくなると、表示が複雑になります。



この例では、ソフトウェアの名前と発行者に関する情報は、主に詳しい知識のあるユーザー向けであるため、別のウィンドウへのリンクが使用されています。

- 追加情報が、アイテムの用途や使用方法を説明する文または語句かどうか。該当する場合は、代わりにツールヒントまたは情報ヒントの使用を検討します。
- 追加情報が、現在のタスクに関連するが、現在表示されている情報からは独立したものであるかどうか。該当する場合は、代わりにタブを使用します。ただし、折りたたみ可能な一覧の方がより柔軟でスケーラブルであるため、多くの場合タブより適しています。
- 追加情報の表示/非表示の切り替えが、実質的にデータ フィルターであるかどうか。該当する場合は、代わりにドロップダウンリストまたはチェック ボックスを使用して一覧全体に対してフィルターを適用します。

## デザイン コンセプト

段階的表示の目的は次のとおりです。

- UI を簡潔にします。本質的要素を中心に置きながら、必要に応じて追加の詳細を表示できます。
- UI の外観を簡潔にします。乱雑な印象を抑えることができます。

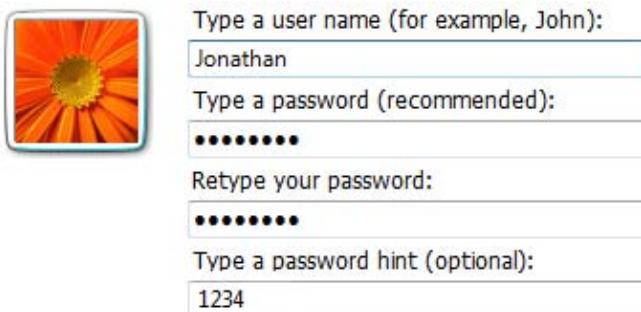
これらの目的の両方を、段階的表示コントロールで達成できます。ユーザーはこのコントロールをクリックすることで詳細を確認できます。ただし、外観を簡潔にするという 2 つ目の目的は、以下に従うことでの段階的表示コントロールを使用せずに達成できます。

- コンテキストに応じた詳細はコンテキスト内だけで表示します。たとえば、選択されたオブジェクトやモードに関連するコマンドやツールバーをコンテキストに応じて自動的に表示できます。
- セカンダリ UI に対しては、アフォーダンスの比重を下げます。アフォーダンスとは、オブジェクトの使用方法などを示唆する視覚的特性です。ユーザーがその場で対話的に操作できる UI が好まれる傾向にありますが、こういった UI すべてが "ここをクリック!" などと主張するように設計すると、視覚的にひどく乱雑になります。多くの場合、セカンダリ UI のアフォーダンスは目立たせず、マウス ポインターを重ねたときに全効果を発揮させることができます。



この例では、評価フィールドを対話的に操作できますが、マウス ポインターを重ねるまではそれがわかりません。

- 必要な操作が行われた後でのみ、続きのステップを表示します。このアプローチは、ユーザーがタスクをよく知っていて、最初のステップを迷わずに行うことができる場合に最適です。



この例では、最初にユーザー名と任意指定のパスワード ボックスのみが表示され、ユーザーがパスワードを入力した後で、確認とパスワードヒントのためのボックスが表示されます。

段階的表示は UI を簡潔にする非常に有効な方法ですが、以下のようなリスクもあります。

- ユーザーが見つけないことがあります。ユーザーは、表示されていないものは存在しない、と見なす場合があります。探しているものが目につかない場合、ユーザーはマウスポインターを重ねたり、クリックしたりしないことがあります。"その他のオプション"などをユーザーがクリックしない可能性は常に存在します。
- 安定性が低下することがあります。段階的表示は、相応の理由がある場合、または少なくとも自然に感じられる場合に使用します。思いがけない状況で表示/非表示が切り替わると、UIに不安定な印象を与えます。

## 段階的表示コントロール

段階的表示コントロールは、通常、動作説明のラベルなしで表示されます。そのため、ユーザーはコントロールの外観のみに基づいて、以下を判断できる必要があります。

- コントロールで段階的表示が提供されていること。
- 現在、展開されている状態か、または折りたたまれている状態か。
- 追加情報、オプション、またはコマンドがタスクの実行に必要なものか。
- 必要な場合に元の状態に戻すための方法。

上記の判断にユーザーによる試行錯誤が必要になるようであれば、試さなくても判断できるように工夫してください。

段階的表示コントロールのアフォーダンスは非常に微弱です。言い換えると、微弱ながらも視覚的特性によって使用方法が示されているということです。次の表は、一般的な段階的表示コントロールの外観を比較したものです。

コントロール	目的	外観	グリフの意味
シェブロン » « ▼ ▲	すべて表示: 全部または部分的に非表示になっているコンテキスト内の残りのアイテムを表示したり、非表示にしたりします。アイテムはインプレースで表示されるか(一重山かっこ)、またはポップアップメニュー内に表示されます(二重山かっこ)。	シェブロンの向きはアクションが発生する方向を示します。	操作後の状態
矢印 ▼ ▶ ▼ ▶	オプションの表示: ポップアップコマンドメニューを表示します。	矢印の向きはアクションが発生する方向を示します。	操作後の状態
プラス/マイナス コントロール + -	コンテナーの展開: 階層構造を参照するときに、コンテナーの内容をインプレースで展開または折りたたみます。	プラスとマイナスの記号は方向を示しませんが、アクションは常に右側に発生します。	操作後の状態
回転する三角形 ◀ ▶	詳細の表示: 個別アイテムに対する追加情報の表示/非表示をインプレースで切り替えます。これらは、コンテナーの展開にも使用されます。	回転する三角形は、回転するレバーに似ています。これらは、アクションが発生した方向を指します。	現在の状態

### 最も重要な点

ユーザーが段階的表示コントロールを見ただけで、その動作を予測できるようにします。これを達成するには、適切な使用パターンを適用し、一貫性を保つように外観、場所、および動作を設定します。

## 使用パターン

段階的表示コントロールにはいくつかの使用パターンがあります。コモンコントロールに組み込まれているものもあります。

### シェvron

シェvronは、全部または部分的に非表示になっているコンテキスト内の残りのアイテムを表示したり、非表示にしたりします。アイテムは、通常、インプレースで表示されますが、ポップアップメニュー内に表示することも可能です。インプレースで表示する場合は、ユーザーが折りたたむまでアイテムは展開されたままになります。

シェvronは次のように使われます。

インプレース UI  
関連付けられて  
いるオブジェク  
トが入力フォー  
カスを受け取  
り、Space キー  
で一重シェブロ  
ンがアクティブ  
になります。



外部ラベル付き  
コマンドボタン  
コマンドボタン  
が入力フォーカス  
を受け取り、  
Space キー  
で一重シェブロ  
ンがアクティブ  
になります。



内部ラベル付き  
コマンドボタン  
コマンドボタン  
が入力フォーカス  
を受け取り、  
Space キー  
でアクティブに  
なります。



この例では、標準のコマンドボタンにシェvronを配置して、意味を示しています。

### 矢印

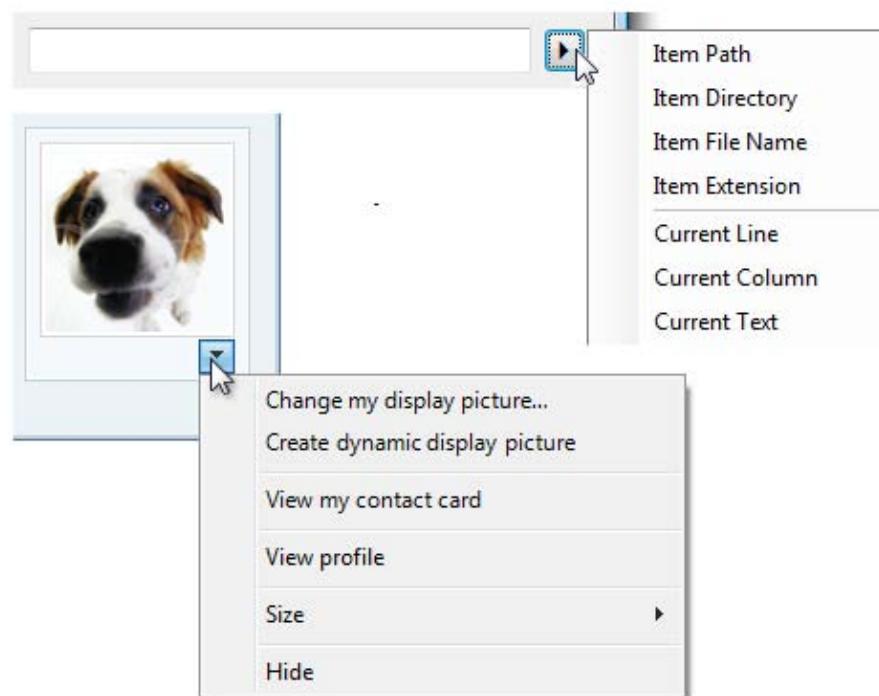
矢印は、ポップアップコマンドメニューを表示します。ユーザーが選択するか他の場所をクリックするまで、アイテムは展開されたままになります。

矢印ボタンは独立したコントロールなので、それ自体が入力フォーカスを受け取り、Space キーでアクティブ化されます。矢印ボタンに親コントロールがある場合は、親コントロールが入力フォーカスを受け取り、矢印はドロップダウンリストコントロールのように Alt + ↓ キーおよび Alt + ↑ キーでアク

ティブ化されます。

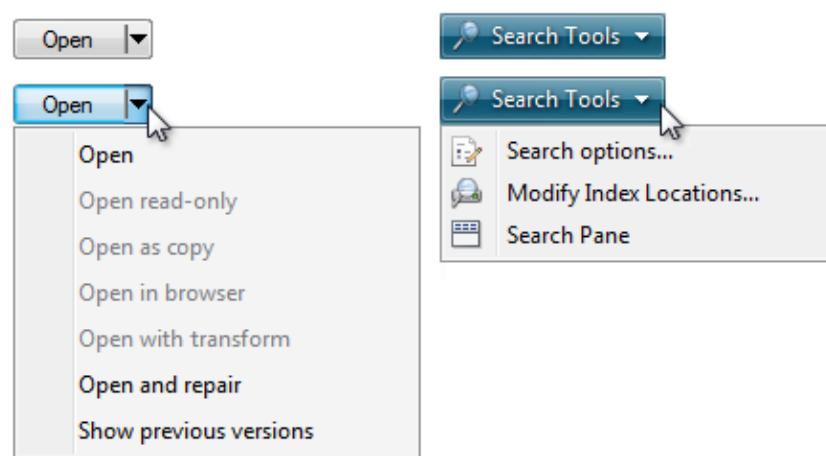
矢印は次のように使用されます。

独立したボタン  
この矢印は独立  
したボタンコン  
トロールです。



これらの例では、右側に配置されている独立した矢印ボタンでコマンドメニューが表示されます。

コマンドボタン  
この矢印はコマ  
ンドボタンの一  
部です。



これらの例では、メニュー ボタンおよび分割ボタンのテキストの右側に矢印ボタンが配置されています。

#### プラス/マイナス コントロール

階層構造を参照するときに、プラス/マイナス コントロールを使ってコンテナーの内容をインプレースで展開または折りたたみます。ユーザーが折りたたむまでアイテムは展開されたままになります。これらのコントロールはボタンのように見えますが、動作はインプレースで行われます。

関連付けられているオブジェクトが入力フォーカスを受け取ります。プラスは → キーでアクティブ化され、マイナスは ← キーでアクティブ化されます。

プラス/マイナス コントロールは次のように使用されます。

折りたたみ可  
能なツリー  
コンテナーの内  
容を複数レベル  
の階層で表示し  
ます。



この例では、関連付けられているコンテナーの左側にプラス/マイナス コントロールが配置されています。

折りたたみ可能なリスト  
コンテナーの内容を 2 レベルの階層で表示します。

■ Accessibility	
■ Appearance	
■ Behavior	
AllowDrop	False
AutoValidate	EnablePreventFocusChange
ContextMenuStrip	(none)
DoubleBuffered	False
Enabled	True
ImeMode	NoControl

この例では、関連付けられているリスト ヘッダーの左側にプラス/マイナス コントロールが配置されています。

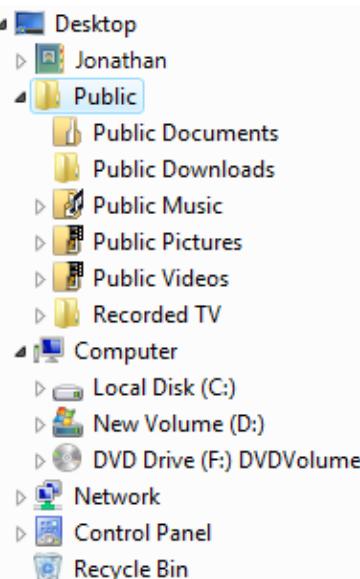
### 回転する三角形

回転する三角形で、個別アイテムに対する追加情報の表示/非表示をインプレースで切り替えます。これらは、コンテナーの展開にも使用されます。ユーザーが折りたたむまでアイテムは展開されたままになります。

関連付けられているオブジェクトが入力フォーカスを受け取ります。折りたたまれた(右向きの)三角形は→キーでアクティブ化され、展開された(下向きの)三角形は←キーでアクティブ化されます。

回転する三角形は次のように使用されます。

折りたたみ可能なツリー  
コンテナーの内容を複数レベルの階層で表示します。



この例では、関連付けられているコンテナーの左側に回転する三角形が配置されています。

折りたたみ可能なリスト  
追加情報を 2 レベルの階層でインプレースに表示します。

<input checked="" type="checkbox"/>	▽ DHCP Server
<b>Description:</b> Select this role to provide automatic IP addressing for clients on your network.	
	<b>Required services:</b> DHCP Server, COM+ Event System, System Event Notification
<input checked="" type="checkbox"/>	▷ DNS Server
<input checked="" type="checkbox"/>	▷ Domain Controller

この例では、関連付けられているリストアイテムの左側に回転する三角形が配置されています。

## プレビュー矢印

シェブロンと同様、インプレースで追加情報の表示/非表示を切り替えます。ユーザーが折りたたむまでアイテムは展開されたままになります。シェブロンと異なる点は、グリフにアクションが図式化されていることです。通常は、発生するアクションが矢印で示されます。



これらの例は Windows Media® プレーヤーのものですが、発生するアクションがグリフに矢印で示されています。

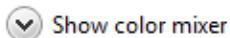
プレビュー矢印は、表示が複雑になる場合や 2 種類以上の表示がある場合など、標準的なシェブロンではコントロールの動作がユーザーに伝わらない状況での、予備的な方法とすることをお勧めします。

## ガイドライン

### 全般

- 用途に応じて段階的表示のパターンを選択します。各使用パターンの説明については、上の表を参照してください。
- 段階的表示コントロールにはリンクを使用しません。「使用パターン」セクションに掲載されている段階的表示コントロールのみを使用します。ただし、[ヘルプトピック](#)へのナビゲーションには "必ず" リンクを使用します。

正しい例:



間違った例:

[Show color mixer](#)

間違った例では、インプレースで追加オプションを表示するためにリンクが使用されています。リンクにより別のページまたはダイアログ ボックスに移動する場合、またはヘルプトピックが表示される場合、これは正しい使用法です。

### 対話操作

- 直接ラベルが付けられないシェブロンと矢印の用途を説明するには、[ツールヒント](#)を使用します。

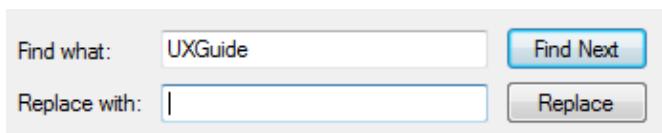


この例では、ラベルのないシェブロン コントロールに対してツールヒントを使用し、その動作を示しています。

- ユーザーがアイテムを展開または折りたたんだ場合は、その状態を維持し、次に同じウィンドウを表示したときに再現されるようにします。ただし、ユーザーが開始時に既定の表示を望む可能性が高い場合は、この限りではありません。状態は、ウィンドウごと、ユーザーごとに維持されるようにします。
- 展開した内容は必ず折りたためるように、また折りたたまれている内容は必ず展開できるようにします。逆の操作がひとめでわかるようにしてください。こうすれば、探索の作業を促進し、ストレスを減らすことができます。逆の操作がひとめでわかるようにする最適な方法は、コントロールを同じ場所に固定することです。コントロールを移動する必要がある場合は、明確に見分けることができる領域内で、相対的に同じ位置を保ちます。

間違った例:





この例では、シェブロンが付いた [置換] ボタンをクリックすると、[置換後の文字列] ボックスが表示されます。クリックすると、展開表示を行う [置換 >>] ボタンが [置換] コマンド ボタンに変更されるので、元の状態に戻す方法がなくなります。

- 段階的表示パターンに適したアクセスキーのみを使用します。これらは、「使用パターン」セクションに掲載されています。段階的表示をアクティブにするために Enter キーは使用しないでください。

#### 提示方法

- 三角形の矢印は、段階的表示以外の目的には使用しません。

間違った例:

▶ Control Panel Home

Classic View

この例では、段階的表示 パターンではないのに矢印が使用され、コマンドがポップアップ ウィンドウに表示されるように思われています。

正しい例:

• Control Panel Home

Classic View

この例では、代わりにドット記号が正しく使用されています。

- 現在のコンテキストに適用されない段階的表示コントロールは、無効にするのではなく削除します。

間違った例:

▼ More options

この例では、適用されない段階的表示が無効化されているだけなので、正しくありません。

#### シェブロン

- インプレースでの表示/非表示には一重のシェブロンを使用します。ポップアップメニューを使用しての表示/非表示には二重のシェブロンを使用します。ただし、内部ラベルを持つコマンド ボタンに対しては、常に二重のシェブロンを使用してください。

正しい例:

▼ More options

間違った例:

▽ More options

間違った例では、インプレースの段階的表示に対してシェブロンが使用されています。

正しい例:

More >>

この例では、インプレースの段階的表示に対してシェブロンが使用されていますが、内部ラベルを持つコマンド ボタンなので、正しい使い方です。

- シェブロンとそれが関連付けられているコントロールとの関係を視覚的に示します。インプレースのシェブロンは、関連付けられている UI の右側に右揃えで配置されるので、シェブロンとそれが関連付けられているコントロールとの間の距離が、かなり離れことがあります。

### 正しい例:



この例では、インプレースのシェブロンとそれが関連付けられている UI との関係が明白です。

### 間違った例:



この例では、インプレースのシェブロンとそれが関連付けられている UI との関係が視覚的に明白ではなく、関係なく存在しているように見えます。

### 矢印

- ・ "戻る"、"進む"、"移動"、"再生"などのコントロールと紛らわしい矢印のグラフィックは使用しません。単純な三角形の矢印(線なしの矢印)をニュートラルな背景で使用します。

### 正しい例:



これらの矢印は、わかりやすい段階的表示コントロールです。

### 間違った例(段階的表示として):



これらの矢印は、段階的表示コントロールには見えません。

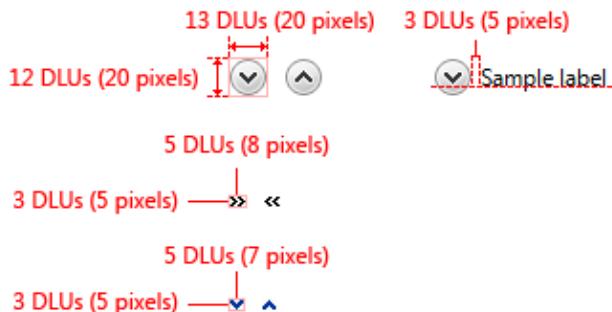
### 間違った例("戻る"、"進む"として):



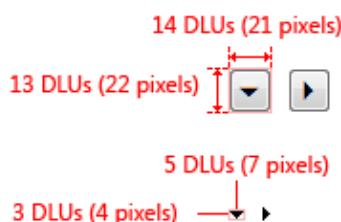
これらの矢印は、段階的表示コントロールのように見えますが、実際は違います。

### 推奨されるサイズと間隔

### シェブロン:



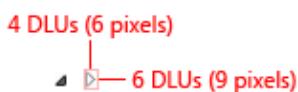
### 矢印:



### プラス・マイナス:



### 回転:



段階的表示コントロールに推奨されるサイズと間隔

## ラベル

- 外部ラベル付きのコマンド ボタン上のシェブロンの場合:
  - 一意な[アクセスキー](#)を割り当てます。割り当てのガイドラインについては、「[キーボード](#)」を参照してください。
  - [センテンススタイルの大文字化](#)を使用します。
  - ラベルには語句を使用し、末尾に句読点はつけません。
  - ボタンをクリックしたときの動作を説明するラベルを付け、状態の変化に合わせてラベルを変更します。
  - 何らかのオプション、コマンド、または詳細が常に表示される場合は、以下のラベルの組み合わせを使用します。
    - オプションの詳細表示/オプションの簡易表示。オプションに、またはオプション、コマンド、詳細が混在する場合に使用します。
    - コマンドの詳細表示/コマンドの簡易表示。コマンドにのみ使用します。
    - 詳細表示/簡易表示。情報にのみ使用します。
    - <オブジェクト名>の詳細表示/<オブジェクト名>の簡易表示。フォルダーなど、上記以外のオブジェクトの種類に使用します。
  - または:
    - オプションの表示/オプションの非表示。オプションに、またはオプション、コマンド、詳細が混在する場合に使用します。
    - コマンドの表示/コマンドの非表示。コマンドにのみ使用します。
    - 詳細の表示/詳細の非表示。情報にのみ使用します。
    - <オブジェクト名>の表示/<オブジェクト名>の非表示。フォルダーなど、上記以外のオブジェクトの種類に使用します。
- 外部ラベル付きのコマンド ボタン上のシェブロンの場合は、標準の[コマンド ボタンのラベル](#)のガイドラインに従います。

## ドキュメント

段階的表示コントロールに言及するときは、以下のことに留意します。

- コントロールに固定されたラベルがある場合は、コントロールをラベルのみで言及します。コントロールの説明は行いません。ラベルのテキストは正確に記述しますが、アクセスキーを示すかっこや下線付き文字は含めません。
- コントロールにラベルがない場合、またはラベルが変更される場合は、コントロールを種類で言及します ("シェブロン"、"矢印"、"三角形"、または "プラス/マイナス ボタン")。必要に応じて、コントロールの場所も説明します。[ページ領域コントロール](#)などのように、コントロールが動的に表示される場合は、コントロールを表示する方法の説明から始めます。

例: フォルダー内のファイルを表示するには、ポインターをフォルダーノードの先頭に移動してから、フォルダーの横の三角形をクリックします。

- ボタンではない他の段階的表示コントロールとの対照を示すのでない限り、コントロールを "ボタン" とは呼びません。
- ユーザー操作を説明する場合は、"クリック" を使用します。明確にすることが必要な場合は、"...をクリックして展開または折りたたみます" を使用します。
- ラベルは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例:

- シェvronの場合: ファイルのサイズを確認するには、[詳細] をクリックします。
- 矢印の場合: すべてのオプションを表示するには、[検索] ボックスの横の矢印をクリックします。
- プラス/マイナスの場合: 画像を表示するには、[画像] をクリックします。

## ラジオ ボタン

適切なコントロールかどうかの判断基準

ガイドライン

全般

従属コントロール

既定値

推奨されるサイズと間隔

ラベル

ドキュメント

"ラジオ ボタン" を使用すると、ユーザーは相互に排他的な一連の関連オプションから選択できます。ユーザーが選択できるオプションは、いずれか 1 つのみです。ラジオ ボタンという名前は、ラジオのチャンネルプリセットのように機能することに由来します。

- Display as a link
- Display as a menu
- Don't display this item

### 典型的なラジオ ボタンのグループ

ラジオ ボタンのグループは、単一のコントロールと同じ機能を果たします。**Tab** キーを使ってアクセスできるのは選択されている項目だけですが、ユーザーは方向キーを使用してグループ内を順に移動できます。

注: レイアウトおよびキーボード ナビゲーションに関するガイドラインは、別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- 相互に排他的な一連の選択肢から 1 つのオプションを選択するために使用するかどうか。該当しない場合は、別のコントロールを使用します。オプションを複数選択できる場合は、代わりに [チェック ボックス](#)、[複数選択リスト](#)、または [チェック ボックスリスト](#) を使用します。
- オプションの数が 2 個以上 7 個以下か。使用する画面領域はオプションの数に比例するので、グループ内のオプション数は 2 個以上 7 個になるようにします。8 個以上のオプションがある場合は、[ドロップダウンリスト](#) または [单一選択リスト](#) を使用します。
- チェック ボックスの方が適していないか。オプションが 2 個のみの場合は、単一の [チェック ボックス](#) を使用することもできますが、チェック ボックスが適しているのは、單一オプションのオン/オフを切り替える場合のみです。オプションがまったく別の内容である場合は、ラジオ ボタンを使用できます。チェック ボックスとラジオ ボタンの両方が考えられる場合は、以下に従います。
  - チェック ボックスがオフになっていることによって示される意味がはっきりしない場合は、ラジオ ボタンを使用します。

間違った例:



正しい例:

- Landscape
- Portrait

この正しい例では、逆の意味ではない選択肢に対して、ラジオ ボタンが適切に使用されています。

- 通常はチェック ボックスを使ってもかまわないような場合でも、可能な選択肢をはっきりと示すために、ウィザード ページではラジオ ボタンを使います。
- 十分な画面領域を確保でき、画面領域を多く使用するだけの重要性を持つオプションであれば、ラジオ ボタンを使用します。それ以外の場合は、チェック ボックスまたはドロップダウン リストを使用します。

### 間違った例:

- Show this message again
- Don't show this message again

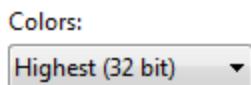
この例のオプションは、ラジオ ボタンを使うほど重要ではありません。

### 正しい例:

- Don't show this message again

この例のオプションは、それほど重要ではないので、チェック ボックスを使うのが効率の良い画面領域の使い方です。

- ページ上に他にもチェック ボックスがある場合は、チェック ボックスを使用します。
- ドロップダウン リストの方が適していないか。ほとんどの状況下で大多数のユーザーに対して既定のオプションが推奨される場合、ラジオ ボタンを使用すると、各オプションが必要以上に注意を引いてしまうことがあります。
  - 各オプションが注意を引かないようにする場合、またはユーザーに変更を勧めない場合は、ドロップダウン リストの使用を検討します。ドロップダウン リストでは現在選択されているオプションに注意が向けられるのに対し、ラジオ ボタンではすべてのオプションが等しく強調されます。



この例では、ドロップダウン リストで現在の選択に注意を集め、ユーザーが変更を行わないように促しています。

- ページ上に他にもドロップダウン リストがある場合は、ドロップダウン リストの使用を検討します。
- コマンド ボタン、コマンド リンク、または分割ボタンの方が適していないか。コマンドの実行方法を制御する目的だけでラジオ ボタンが使用されているのであれば、多くの場合、代わりにコマンドのバリエーションを提供する方が良い結果になります。この方法により、ユーザーは単一の操作で正しいコマンドを選択できるようになります。
- 選択対象がデータではなく、プログラム オプションかどうか。オプションの値は、コンテキストや他のデータに基づかない値である必要があります。データの場合は、ドロップダウン リストまたは単一選択リストを使用します。
- コントロールをウィザード ページまたはコントロール パネルで使用する場合、そのコントロールがメイン指示テキストへの回答の形式を取り、ユーザーが後に変更できるものであるかどうか。該当する場合は、ラジオ ボタンの代わりにコマンド リンクを使用して対話操作を効率的にすることを検討します。
- 値が数値以外かどうか。数値データの場合は、テキスト ボックス、ドロップダウン リスト、またはスライダーを使用します。

## ガイドライン

### 全般

- オプションを論理的な順序で一覧にします。選択される可能性が高いものから低いもの、処理が単純なものから複雑なもの、またはリスクが最も低いものから高いものなどの順で並べます。アルファベット順は言語に依存し、ローカライズ時に順序を再現できないのでお勧めしません。
- いずれの選択肢も妥当ではないことがある場合、いずれも適用しないことを選択する別のオプションを追加します。これには、"なし"、"適用しない"などを使用します。
- ラジオ ボタンは横にではなく縦に並べるようにします。横方向に並べると読みにくくなり、またローカライズもしにくくなります。

### 正しい例:

- Left
- Center
- Right

この例では、ラジオ ボタンが縦に並んでいます。

間違った例:

- Left
- Center
- Right

この例では、横方向に並んでいるため読みにくくなっています。

- グループ ボックスを使ってラジオ ボタンを整理することが妥当か再検討します。この方法では、不必要に散漫な画面に見える場合があります。
- ラジオ ボタンのラベルは、グループ ボックスのラベルとしては使用しません。
- ラジオ ボタンによる選択は、以下の目的には使用しません。
  - コマンドの実行。
  - 追加の情報入力を行うためのダイアログ ボックスなど、他のウィンドウの表示。
  - 選択されているコントロールに関連する、他のコントロールの動的な表示/非表示(このようなイベントは、スクリーン リーダーで検出できません)。ただし、選択に基づいてテキストを動的に変更することは可能です。

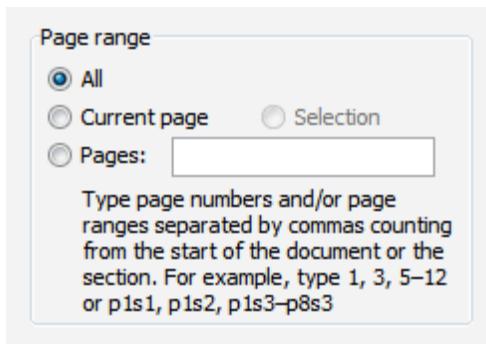
#### 従属コントロール

- 従属コントロールは、ラジオ ボタンおよびそのラベルの右側または下方に配置します。下方に配置する際は、インデントしてラジオ ボタンのラベルに揃えます。ラジオ ボタンのラベルの最後にコロンを使用します。

- Never
- After period of time (in seconds):  

この例では、ラジオ ボタンおよびその従属コントロールは、ラジオ ボタンのラベルおよびそのアクセスキーを共有しています。この場合、ラジオ ボタンから従属テキスト ボックスにフォーカスを移動するには、方向キーを使用します。

- 従属コントロールの編集可能なテキスト ボックスやドロップダウン リストがラジオ ボタンのラベルを共有している場合は、ボックスやリストを無効にしないでください。ユーザーがボックスに入力したり何かを貼り付けたりしたときに、対応するオプションが自動的に選択されるようにし、対話操作を単純化します。



この例では、ページ番号を入力すると自動的に [ページ] が選択されます。

- ラジオ ボタンを、他のラジオ ボタンまたはチェック ボックスと入れ子にすることは避けます。可能であれば、すべてのオプションを同じレベルにします。

正しい例:

- Let Internet Explore decide how pop-ups should open
- Always open pop-ups in a new window
- Always open pop-ups in a new tab

この例では、オプションが同じレベルにあります。

間違った例:

- Let Internet Explore decide how pop-ups should open
- Always open pop-ups:
  - In a new window
  - In a new tab

この例では、オプションを入れ子にすることで不必要に複雑化されています。

- ラジオ ボタンを、他のラジオ ボタンまたはチェック ボックスとどうしても入れ子にする必要がある場合は、上位レベルのオプションが選択されるまで、従属コントロールを無効にしておきます。こうすることで、従属コントロールの意図に関する混乱を回避できます。

## 既定値

- ラジオ ボタンのグループは相互に排他的な一連の選択肢を表すので、常にいずれかのラジオ ボタンが既定で選択されているようにします。この際、最も安全(データの消失やシステム アクセスが失われるなどを防ぐため)で、最もセキュリティの高い非公開のオプションを選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用されるオプションまたは最も便利なオプションを選択します。

例外: 以下の場合は既定の選択を行いません。

- 安全性、セキュリティ、または法的な理由で、許容できる既定のオプションがない場合は、ユーザーが明示的に選択する必要があります。ユーザーが選択しない場合は、エラー メッセージを表示し、いずれかを選択することを強制します。
- ユーザー インターフェイス (UI) に現在の状態を反映する必要がある場合、ユーザーの選択前にオプションが設定されていると混乱が生じます。既定値が設定されていると、選択する必要がないという意味にとられかねません。
- 先入観を持たずにデータを収集することが目的の場合は既定値を設定しません。既定値がデータ収集中バイアスを生じる可能性があります。
- ラジオ ボタンのグループが混在状態のプロパティを表す場合は既定値を設定しません。この状態は、設定が同じではない複数のオブジェクトのプロパティを表示するときに発生します。この場合、各オブジェクトは有効な状態にあるので、エラー メッセージは表示しません。
- 最初のオプションを既定のオプションにします。ユーザーは、このような状態を期待しています。ただし、こうすることで論理的な順番が崩れる場合を除きます。最初のオプションを既定のオプションにする際に、オプションのラベルの変更が必要になることがあります。

間違った例:

- Apply policy:
- Now
  - Later

この例では、既定のオプションが最初のオプションではありません。

正しい例:

- Policy:
- Apply later
  - Apply now

この例では、オプションのラベルの言い回しを変えて、最初のオプションが既定のオプションになるようになっています。

## 推奨されるサイズと間隔

Display Sidebar on this side of screen:  
3 DLUs (5 pixels)  Right  
4 DLUs (7 pixels)  Left

Radio1  10 DLUs (17 pixels)

ラジオ ボタンに推奨されるサイズと間隔

## ラベル

ラジオ ボタンのラベル

- ラジオ ボタンにはすべてラベルを付けます。
- 各ラベルに、一意な[アクセス キー](#)を割り当てます。割り当てのガイドラインについては、「[キーボード](#)」を参照してください。
- [センテンス スタイルの大文字化](#)を使用します。
- ラベルには文ではなく語句を使用し、末尾に句読点は付けません。
  - 例外: ラジオ ボタンのラベルが後続の従属コントロールのラベルでもある場合は、ラベルの末尾にコロンを使用します。
- 同じ文法構造の表現を使用し、すべてのラベルの長さがおおよそ同じになるようにします。
- オプション間の違いがわかるラベル テキストにします。すべてのオプションが同じテキストで始まっている場合は、そのテキストをグループ ラベルに移動します。
- 肯定文を使用します。たとえば、"しない"ではなく"する"、"印刷しない"ではなく"印刷する"を使用します。
- ラベルではオプションのみを説明します。ラベルを簡潔にすることで、メッセージおよびドキュメントで言及することが容易になります。オプションに詳細な説明が必要な場合は、句点を付けた完全な文として[静的 テキスト](#) コントロール内で説明します。

On (recommended)

This setting blocks all outside sources from connecting to this computer, except for those unblocked on the Exceptions tab.

Off

Avoid using this setting. Turning off Windows Firewall will make this computer more vulnerable to hackers or malicious software.

この例では、独立した静的テキスト コントロールを使用してオプションが説明されています。

注: 1つのラジオ ボタンに説明を追加したからといって、すべてのラジオ ボタンに説明を追加する必要はありません。関連情報は可能な限りラベルで提供し、必要がある場合にのみ説明を追加します。ラベルの言い回しを変えただけの説明は、一貫性を保つためであっても追加しないでください。

- 強く推奨されるオプションに対しては、ラベルに"(推奨)"を追加します。補足的なメモに対してではなく、コントロールのラベルに追加してください。
- 詳しい知識のあるユーザー向けのオプションに対しては、ラベルに"(詳細設定)"を追加します。補足的なメモに対してではなく、コントロールのラベルに追加してください。
- 複数行にわたるラベルを使用する必要がある場合は、ラベルの先頭をラジオ ボタンと揃えます。
- 従属コントロール、それに含まれる値、または単位のラベルを使用して、文や句を作成しないでください。文の構造は言語ごとに異なるので、このように設計するとローカライズできなくなります。

ラジオ ボタン グループのラベル

- すべてのラジオ ボタン グループにラベルが必要です。静的テキストまたはグループ ボックスを使ってラベルを付け、文ではなく単語または句を使用し、末尾にコロンを付けます。

例外: ダイアログ ボックスの[メイン指示テキスト](#)の单なる言い直しになる場合は、ラベルを省略します。メイン指示テキストに(疑問文でない限りは)コロンを付けます。存在する場合はアクセス キーも

メイン指示テキストに付けます。

許容される例:

Select an alignment

Alignment:

- Align left:
- Center
- Align right

この例では、ラジオ ボタン グループのラベルは、メイン指示テキストを言い直しているだけです。

より良い例:

Select an alignment:

- Align left:
- Center
- Align right

この例では、余分なラベルは削除され、メイン指示テキストにコロンが付けられています。

- ラベルにはアクセス キーを割り当てません。ラベルにアクセス キーは不要であり、割り当てる他のアクセス キーの割り当てが困難になります。
  - 例外: すべてのコントロールに一意なアクセス キーを割り当てることができない場合、個々のコントロールの代わりに、ラベルにアクセス キーを割り当てるすることができます。詳細については、「[キー ボード](#)」を参照してください。

## ドキュメント

ラジオ ボタンに言及するときは、以下のことに留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、アクセス キーを示すかっこや下線付き文字、およびコロンは含めません。
- プログラミングおよびその他の技術文書では、"ラジオ ボタン"と記述します。それ以外のドキュメント(特にユーザー向けマニュアル)では、"オプション ボタン"を使用します。
- ユーザー操作を説明する場合は、"クリック"を使用します。
- ラベルは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例: [現在のページ] をクリックしてから [OK] をクリックします。

## 検索ボックス

適切なコントロールかどうかの判断基準

デザインコンセプト

ガイドライン

対話操作

場所

外観

機能

推奨されるサイズと間隔

テキスト

ドキュメント

"検索ボックス" を使うと、検索結果のフィルター処理または強調表示を利用して、大量のデータの中から特定のオブジェクトまたはテキストをすばやく見つけることができます。標準の検索コントロールというものはありませんが、作成するプログラムの検索機能を Windows® の検索機能と一貫性があるものにすることを強くお勧めします。

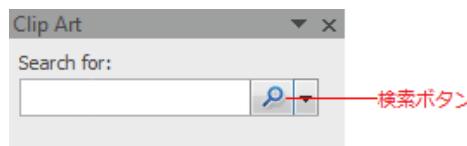
検索には 2 つの種類があります。

- クイック検索: 入力中にすぐに検索結果が表示されます。ボタンをクリックする必要がないので、検索シンボルの虫眼鏡はボタンではなくグラフィックとして表示されます。



クイック検索を使用する典型的な検索ボックス。文字を入力するたびに検索が自動的に実行されます。

- 通常検索: 検索ボタンをクリックしたときに検索が実行されます。検索シンボルの虫眼鏡はボタン上に表示されます。



通常検索を使用する典型的な検索ボックス。検索を実行するには、ボタンをクリックします。

これらの検索オプションのどちらかまたは両方をユーザーに提供できます。

## 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- 特定のオブジェクトを見つけにくいかどうか。この状況は以下の場合に考えられます。
  - 多数のオブジェクトがある。
  - オブジェクトが単一の場所にまとまっていない。検索は、ツリー構造内のオブジェクトを探す際に特に有効です。
  - 見つけにくい検索データ(メタデータなど)である。
- ユーザーがドキュメント内の特定のテキストを見つけ出す必要があるかどうか。
- 実装する機能により、5秒以内に適切な検索結果が返されるかどうか。該当しない場合は、検索ダイアログ ボックスなど、時間がかかる検索に配慮して視覚的フィードバックを提供できる設計の検索機能を提供します。

## デザイン コンセプト

多くのシナリオで、検索は非常に重要な最初のステップです。ユーザーがオブジェクトを使用するには、まずそれらを見つける必要があります。ハードディスクの容量は増大し続けており、そこに保存されるオブジェクトはますます多くなっていますが、オブジェクトを目視で探す方法はそれに伴って効率が良くなるわけではありません。検索は、ユーザー エクスペリエンスの一部として、単純で一貫性と信頼性の高いものである必要があります。

Windows の検索ボックスには次のような特長があります。

- [エクスプローラー] ウィンドウの一部であり、所在と機能の判別が容易です。
- 外観と動作に一貫性があります。
- 効率的で高速です。クイック検索モードでは即座に結果が表示されます。

Windows では、以下の部分で検索ボックスが使用されています。

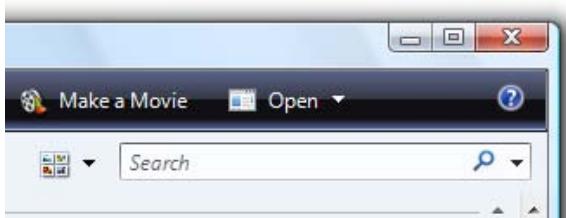
- エクスプローラー
- エクスペリエンス関連 (Microsoft® Windows Media® Player、Windows® フォト ギャラリー、Windows Internet Explorer®)
- スタートメニュー (プログラムや最近使ったファイルの検索)
- コントロールパネル ホーム (コントロールパネルアイテムとタスクの検索)
- ヘルプ (関連ヘルプ トピックの検索)

## 外観

クリック検索がサポートされ、Windows の検索機能は飛躍的に向上しました。即座に検索結果が得られることで、Windows の機能がより強力かつ直接的なものになっています。

Windows エクスプローラーおよびアプリケーションのウィンドウでは、検索が補助エントリ ポイントである場合に右上隅に配置されます。探しているものがこれらのウィンドウ内に見つからない場合に、ユーザーは検索メカニズムを探します。ただし、検索がプライマリ エントリ ポイントである場合は、ウィンドウの上部中央に配置されます。

[検索] ボタンは検索ボックスに隣接しています。領域を最小にするために、検索ボックスではラベルの代わりにオプションの [プロンプト](#) が使用されます。検索キーワードを入力などの指示や、"画像を検索"などの検索範囲を示すプロンプトを使用できます。



ラベルも独立したボタンも使用していない Windows のクリック検索の外観は、軽快を感じさせます。

検索が正しく実行されると、検索結果を表示するための仮想ページが作成され、バック スタックとアドレスバーに追加されます。元のページを復元し検索ボックスをクリアするには、[戻る] をクリックする、アドレスバー内の元のページをクリックする、Esc キーを押す、検索ボックス内のテキストを消すなどの方法があります。

前の検索結果ページを復元することなく、検索ボックスをクリアすることもできます。クリック検索モードでは、ユーザーが入力し始めると、検索シンボルの虫眼鏡がクリア ボタン "x" で置き換えられます。"x" にポインターを重ねると、ボタンの外観になり、クリックできるようになります。



ユーザーは、コントロールの右端にある "x" をクリックして検索ボックスをクリアできます。

通常検索モードでは、クリア ボタンはオプションです。検索の完了まで時間がかかる場合などに、このボタンがあると、"x" をクリックして進行中の検索を停止できるので便利です。既に検索が完了している場合は、"x" をクリックして検索ボックスをクリアできます。

## ガイドライン

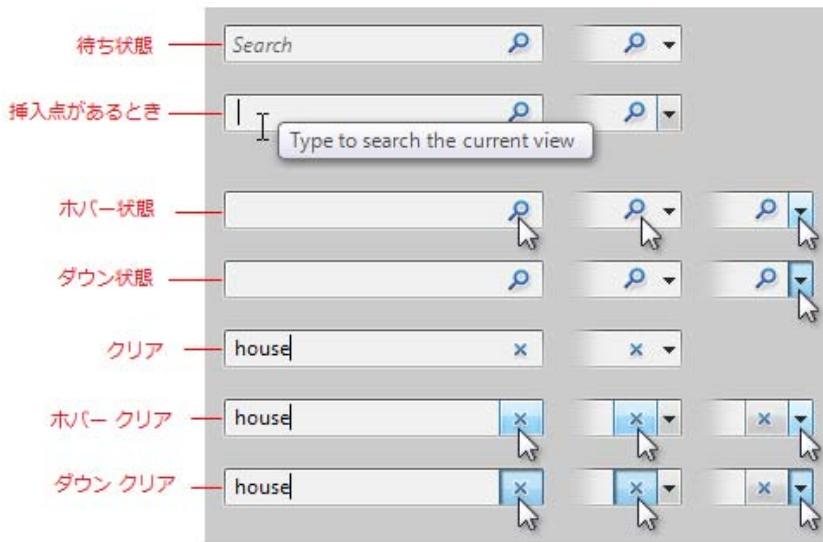
### 場所

- アプリケーションのウィンドウでは、検索は右上隅に配置します。
- ポップアップ ウィンドウでは、最も目立つ使いやすい場所に配置します。
- 例外: ウィンドウ内でユーザーが行う最初の操作が検索の場合、つまりプライマリ エントリ ポイントである場合、ウィンドウの上部中央に検索を配置します。

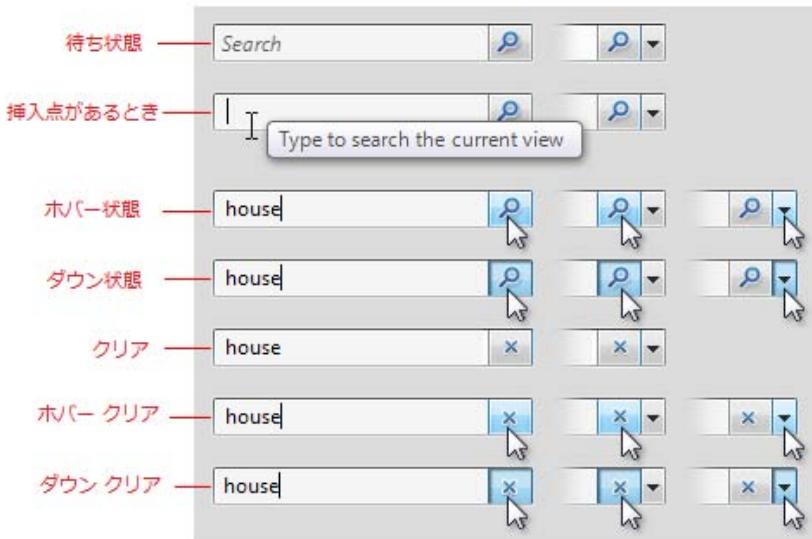
### 外観

- 標準の検索ボタン グラフィックを使用します。以下の 3 つのパターンがあります。
  - 検索シンボルの虫眼鏡のみ (ポインターを重ねてもボタン化しません)。クリック検索に使用します。
  - ボタン化した検索シンボルの虫眼鏡。検索を開始するためにボタンのクリックが必要な場合に使用します。
  - ボタン化した検索シンボルの虫眼鏡とドロップダウン矢印。ユーザーが検索範囲を変更可能、または他の設定を選択可能な場合に、ドロップダウン矢印を追加します。

- ・ クイック検索では、ドロップダウン矢印のみを使用し、ポインターを重ねたときにボタン化するようにします。
- ・ 通常検索では、ドロップダウン矢印をボタンとして表示します。



クイック検索の視覚的な仕様



通常検索の視覚的な仕様

- ・ ラベルを使用せずに、オプションのプロンプトを使用します。一般的なファイル検索と見間違えやすい場合は、プロンプトを使用して範囲を示します。そうでない場合は、"検索キーワードを入力"などの簡潔な語句を表示します。



これらの例では、ユーザーが検索を行う際に役立つ簡潔なテキストプロンプトが使用されています。

#### 対話操作

- ・ 入力フォーカスを取得すると、前に入力されたテキスト全体が自動的に選択されます。このようにすることで、ユーザーは入力するだけで新しい検索を実行できます。前の検索キーワードを変更する場合は、方向キーを使用してキャレットを移動します。



Live Search

book

Search

Options ▾

この例では、入力フォーカスを取得したときに、前に入力したテキストが選択されています。

- 検索ボックスにキーボードショートカットの **Ctrl + E** を割り当ててください。キーボードショートカットの割り当ての詳細については、「[Windows のキーボードショートカットキー](#)」を参照してください。

## 機能

- 可能であれば常にクイック検索を利用可能にします。待ち時間が長くなつても通常検索を使用するメリットがあるシナリオの場合は、通常検索とクイック検索の両方を提供します。
- 通常検索では関連性の高い検索結果を5秒以内に返す必要があります。クイック検索では検索結果を2秒以内に返す必要があります。その後、プログラムが応答し、ユーザーが他のタスクを実行できる限りは、関連性の低い検索結果を徐々に追加していくことができます。この応答性を確保するには、検索データのインデックス化が必要なことがあります。
- 通常検索とクイック検索の両方を提供する場合は、クイック検索の結果が通常検索の結果のサブセットである必要があります。
- すべての検索は前方一致型であり、部分一致や後方一致はありません。末尾のワイルドカード文字列は必須ではなく、結果に影響しません。複数の単語が入力される場合には、OR検索を使用します。
- 検索が正しく実行されると、検索結果を表示する仮想ページがバック スタックとアドレスバーに追加されます。複数回検索しても仮想ページは1つなので、[戻る]をクリックすると常に元のページに戻ります。
- ランク付けが必要な場合は、検索結果を関連性で分けます。
- 何も入力せずに検索すると、元のページが返されます。

## 推奨されるサイズと間隔

実際のコントロールサイズ：



125 DLUs (187 pixels)

15 DLUs (24 pixels)



15 DLUs (24 pixels)

表示されるサイズ：



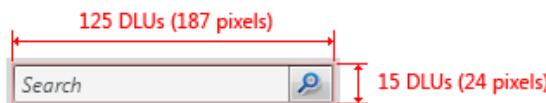
123 DLUs (185 pixels)

13 DLUs (22 pixels)

コントロールの外周に透明な1ピクセルのボーダーがあるため、表示されるサイズはコントロールのサイズより小さい

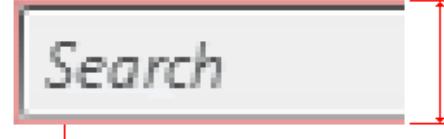
## クイック検索に推奨されるサイズと間隔

実際のコントロール サイズ：



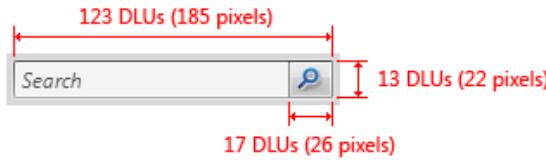
125 DLUs (187 pixels)

15 DLUs (24 pixels)



15 DLUs (24 pixels)

表示されるサイズ：



123 DLUs (185 pixels)

13 DLUs (22 pixels)

17 DLUs (26 pixels)

コントロールの外周に透明な1ピクセルのボーダーがあるため、表示されるサイズはコントロールのサイズより小さい

## 通常検索に推奨されるサイズと間隔

### テキスト

- 検索ボックスのプロンプトでは、"検索キーワードを入力"などの指示や、"画像を検索"などの検索範囲を示すテキストを使用します。
- プロンプトのテキストは簡潔にする必要があります。単語または短い語句で十分です。
- [センテンススタイルの大文字化](#)を使用します。
- 末尾に句読点や省略記号は付けません。

## ドキュメント

- 言及する際は "検索ボックス" とします。英文ドキュメントでは、"Search box" のように最初の単語の最初の文字は大文字にし、"box" はすべて小文字にします。
- 2種類の検索をそれぞれ、"クイック検索"、"通常検索" と表記します。英語ドキュメントでは "Instant search" (クイック検索) の最初の文字は大文字にしますが、"regular search" (通常検索) はすべて小文字にします。

# スライダー

適切なコントロールかどうかの判断基準

ガイドライン

推薦されるサイズと間隔

ラベル

ドキュメント

"スライダー" を使用すると、連続的な値範囲から選択できます。スライダーには、範囲を示すバーと、現在の値を示すインジケーターがあります。オプションの目盛りは値を示します。



典型的なスライダー

注: レイアウトに関するガイドラインは、別の項目として記載しています。

## 適切なコントロールかどうかの判断基準

定義済みの連続的な値(音量、明るさなど)または不連続な値(画面解像度の設定など)をユーザーが設定できるようにする場合に、スライダーを使用します。

ユーザーが値を数値ではなく相対量として考えることがわかっている場合、スライダーは良い選択です。たとえばオーディオ音量の場合、ユーザーは値を 2 または 5 に設定しようと考えるのではなく、低または中に設定しようと考えます。

以下の点に基づいて判断します。

- その設定が相対量のように見えるかどうか。該当しない場合は、ラジオボタン、ドロップダウンリスト、または単一選択リストを使用します。
- その設定が既知の正確な数値かどうか。該当する場合は、数値テキストボックスを使用します。
- 設定変更の効果に対するフィードバックをすぐに得られることがユーザーの役に立つかどうか。該当する場合は、スライダーを使用します。たとえば、色合い、彩度、輝度などの値を変更した効果をすぐに確認できることで、色の選択がより簡単になります。
- 設定範囲内に 4 個以上の値があるかどうか。該当しない場合は、ラジオボタンを使用します。
- ユーザーが値を変更できるかどうか。スライダーはユーザーとの対話操作のためにあります。ユーザーが決して値を変更できない場合は、読み取り専用のテキストボックスを使用します。

スライダーも数値テキストボックスも考えられるときは、次の場合に数値テキストボックスを使用します。

- 画面領域があまりない場合
- ユーザーがキーボードの使用を好む可能性が高い場合

スライダーは次の場合に使用します。

- ユーザーにとって、すぐにフィードバックが得られると便利な場合

## ガイドライン

- 自然な向きにします。たとえば、通常は縦に表示される現実世界の値(温度など)をスライダーで表す場合は、縦向きにします。
- ユーザーの文化を反映する向きにします。たとえば、西洋文化では文字を左から右へ読むので、横型のスライダー場合、バーの左側を低い値、右側を高い値にします。文字を右から左へ読む文化では、この逆にします。

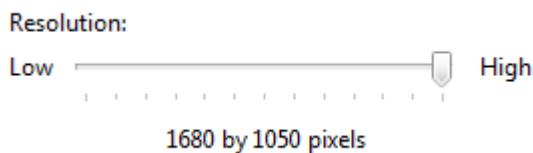
- ユーザーが容易に目的の値を設定できるように、コントロールのサイズを決めます。不連続な値を設定する場合、ユーザーが任意の値をマウスで簡単に選択できるようにします。
- 値の範囲が広く、ユーザーの選択値が範囲の一方に偏ることが多い場合は、非線形的なスケールの使用を検討します。たとえば、時間の場合は値を 1 分、1 時間、1 日、1 か月のように設定します。
- 実用的である場合は常に、ユーザーが選択している間または直後にフィードバックを提供するようにします。たとえば、Microsoft® Windows® の音量コントロールでは、変更後の音量を示すためにビープ音が鳴ります。
- ラベルを使用して値の範囲を示します。

例外: スライダーが縦向きで、一番上のラベルが "最大"、"高"、"大" などの場合、意味は明らかなので一番下のラベルは省略できます。



この例では、縦向きのスライダーを使用することにより、範囲ラベルは必要なくなります。

- 設定のおおよその値をユーザーが知る必要がある場合は、目盛りを使用します。
- 選択した設定値をユーザーが正確に知る必要がある場合は、目盛りと値ラベルを使用します。設定の意味を理解するためにユーザーが単位を知る必要がある場合は、値ラベルを常に使用します。



この例では、選択した値を明確に示すラベルが使用されています。

- 横向きのスライダーでは、スライダーの下に目盛りを配置します。縦向きのスライダーの目盛りは、西洋文化では右側に、文字を右から左に読む文化では左側に配置します。
- 関連性が明確になるように、値ラベルは全体をスライダーコントロールの下に配置します。

間違った例:



All accelerations are enabled. Use this setting if your computer has no problems. (Recommended)

この例では、値ラベルがスライダーの下にありません。

- スライダーを無効にする場合は、関連付けられているラベルも無効にします。
  - 同じ設定に対して、スライダーと数値テキストボックスの両方を使用しないでください。より適切なコントロールだけを使用します。
- 例外: すぐにフィードバックが必要で、かつ明確な数値を設定可能にする必要がある場合は、両方のコントロールを使用します。
- スライダーは、進行状況のインジケーターとして使用しません。
  - スライダー インジケーターは、既定のサイズから変更しません。

間違った例:



この例では、既定より小さなサイズが使用されています。

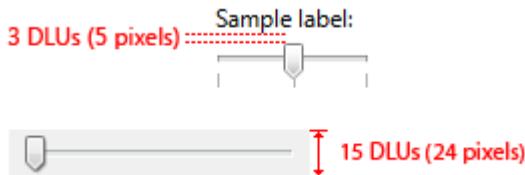
正しい例:



この例では、既定のサイズが使用されています。

- 目盛りすべてにラベルを付けたりしないでください。

## 推奨されるサイズと間隔



スライダーに推奨されるサイズと間隔

## ラベル

### スライダーのラベル

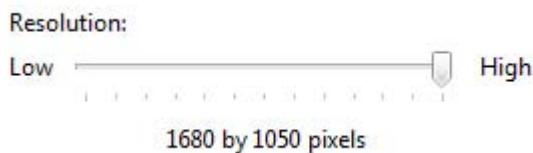
- 静的テキストを使用する場合は末尾にコロンを付け、グループ ボックス ラベルを使用する場合は末尾に句読点を付けません。
- 各ラベルに、一意なアクセスキーを割り当てます。割り当てのガイドラインについては、「キーボード」を参照してください。
- センテンススタイルの大文字化を使用します。
- 静的テキスト ラベルの場合は、スライダーの左に配置するか、または上にラベルを配置してスライダー(存在する場合は左側の範囲ラベル)の左端と揃えます。

### 範囲ラベル

- スライダーの範囲の両端にラベルを付けます。ただし、縦向きでラベルが不要な場合を除きます。
- 可能であれば、各ラベルには単語のみを使用します。
- 末尾に句読点は付けません。
- これらのラベルには、同じ文法構造の説明的な表現を使用します。例: 最大/最小、多/少、高/低、大/小
- センテンススタイルの大文字化を使用します。
- アクセスキーは割り当てません。

### 値ラベル

- 値ラベルが必要な場合は、スライダーの下に表示します。
- コントロールの中央に揃えて配置し、単位(ピクセルなど)を含めます。



この例では、値ラベルがスライダーの下に配置され、単位も含まれています。

## ドキュメント

スライダーに言及するときは、以下のことに留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用し、"スライダー"という語を追加します。アクセスキーを示すかっこや下線付き文字、およびコロンは含めません。

- ユーザー操作を説明する場合は、"つまみを移動する" を使用します。
- ラベルは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例: 画面の解像度を上げるには、[画面の解像度] スライダーのつまみを右に移動します。

# スピン コントロール

適切なコントロールかどうかの判断基準

ガイドライン

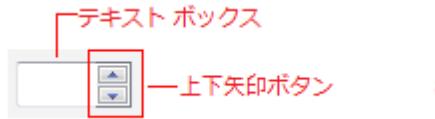
全般

値

ラベル

ドキュメント

"スピン コントロール" では、矢印ボタンをクリックすることにより、コントロールに関連付けられている数値テキスト ボックス内の値を増減できます。"スピン ボックス" という用語は、テキスト ボックスとそれに関連付けられているスピン コントロールの組み合わせを指します。



## 典型的なスピン ボックス

スピン コントロールは、マウスから手を離さずに値を変更できるという点で多くのユーザーに好まれます。スピン コントロールがテキスト ボックスと対になっている場合、ユーザーはテキスト ボックスに値を直接入力するか貼り付けることができるので、スピン コントロールの使用は任意になります。

スピン コントロールは数値の入力に使用するものですが、入力は純粋な整数だけにする必要はなく、小数を入力することも、マイナス記号、コロンやハイフンなどの区切り記号、および単位修飾子を追加することもできます。

注: テキスト ボックスおよびレイアウトに関するガイドラインは、それぞれ別の項目として記載しています。

## 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- コントロールを数値の入力に使用するかどうか。該当しない場合は、ドロップダウンリスト、スライダーなど、固定値のセットから選択を行う別のコントロールを使用します。スクロールにはスクロールバーを使用します。
- ユーザーが、値を数値ではなく相対量として考えるかどうか。該当する場合は、代わりにスライダーを使用します。スピン ボックスは、正確な既知の数値にのみ使用します。たとえばオーディオ音量の場合、ユーザーは値を 2 または 5 に設定しようと考えるのではなく、低または中に設定しようと考えます。
- コントロールをテキスト ボックスと対にするかどうか。該当しない場合は、スピン コントロールは使用しません。スピン コントロールは、単独で使用したり、テキスト ボックス以外のコントロールと共に使用するものではありません。

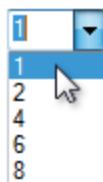
間違った例:



この例では、スピン コントロールが動的グラフィックの制御に使用されています。

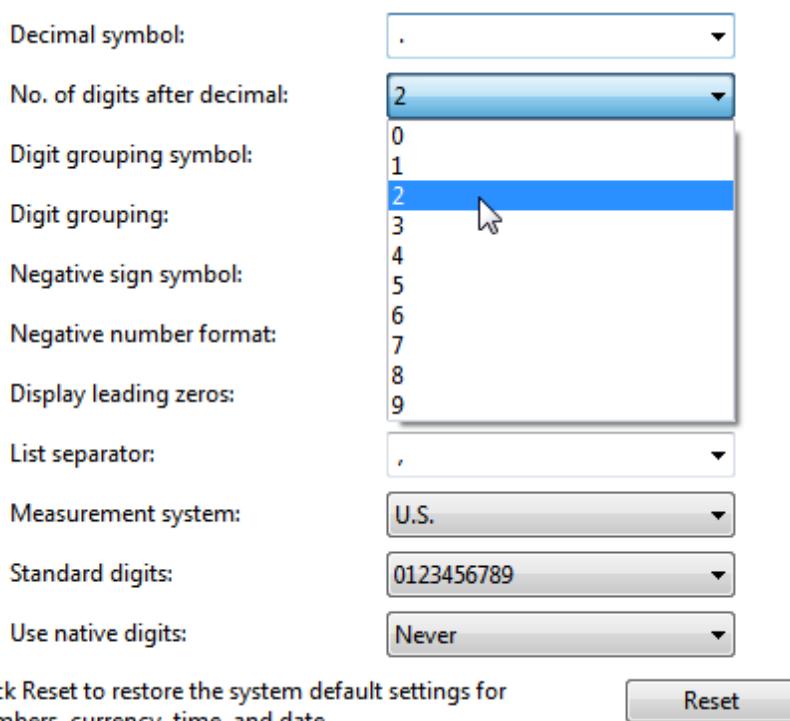
- 連続した値で構成される範囲が有効かどうか。該当しない場合は、代わりに有効な値を列挙したドロップダウンリストを使用します。

Number of disk drives:



この例では、ディスク ドライブの数がすべての値で有効とは限らないため、ドロップダウンリストの方が適しています。

- ・スピン コントロールの使用が現実的かどうか。スピン コントロールの使用が現実的なのは次のような場合です。
  - ・小さな数値を入力する場合(通常 100 未満)
  - ・既存の値や既定値を少し変更する場合
- スピン コントロールはどのような数値の入力にも使用できますが、ここに挙げた状況以外では非効率的になります。
- ・スピン コントロールが有用かどうか。コントロールを使用しようとしているコンテキストで、ユーザーがマウスを使用する可能性が高いかどうかを考えたとき、該当しない場合は、スピン コントロールは必須ではありません。
- ・兄弟コントロールとして[ドロップダウンリスト](#)を使用しているか。他にドロップダウンリストを使用している場合は、一貫性を保つためにドロップダウンリストを使用するようにします。



この例では、スピン ボックスを使用できないこともないですが、一貫性を保つためにドロップダウンリストを使用しています。

- ・タッチ パネルやペンを使用するユーザーが主な対象かどうか。該当する場合は、代わりにドロップダウンリストを使用するようにします。タッチ パネルやペンを使用する場合、スピン コントロール内の矢印ボタンは小さすぎて効率的ではありません。

スライダーもスピン ボックスも考えられるときは、次の場合にスピン ボックスを使用します。

- ・画面領域があまりない場合
- ・ユーザーがキーボードの使用を好む可能性が高い場合

スライダーは次の場合に使用します。

- ・ユーザーにとって、すぐにフィードバックが得られると便利な場合

## ガイドライン

### 全般

- ・スピン コントロールは、現実的で有用な場合は常に使用します。[適切なコントロールかどうかの判断基準](#)を参照してください。
  - ・例外: 同じユーザー インターフェイス(UI)上の他のテキスト ボックスと一貫性を保つために、必ずし

も現実的とは言えなくともスピン コントロールを使用することができます。

正しい例:

Month:  Day:  Year:

A screenshot showing three separate spinners for Month, Day, and Year. Each spinner has a small input field with a value and up/down arrow buttons on either side.

この例では、必ずしも現実的とは言えなくとも、一貫性を保つためにスピン コントロールを年の設定にも使用しています。

間違った例:

Use the following IP address:  
IP address:

A screenshot showing an IP address input field with a checked radio button next to it. The radio button is blue with a white circle. The IP address field contains "255 . 255 . 255 . 0".

この例では、スピン コントロールは使用できません。

- スピン コントロールは、常にどのテキスト ボックスと関連づけられているかがわかるようにします。こうすると、スピン コントロールの位置はテキスト ボックスの内部になります。

正しい例:

Calendar  
When a two-digit year is entered, interpret it as a year between:  
1930 and 2029

A screenshot showing a calendar configuration dialog. It includes a text input field containing "When a two-digit year is entered, interpret it as a year between:" followed by two spinners labeled "1930" and "2029".

間違った例:

Calendar  
When a two-digit year is entered, interpret it as a year between:  
1930 and 2029

A screenshot showing a calendar configuration dialog. It includes a text input field containing "When a two-digit year is entered, interpret it as a year between:" followed by two spinners labeled "1930" and "2029". The spinners are positioned outside the main text input field.

正しい使用例では、スピン コントロールの位置は、関連付けられているテキスト ボックスの内部になっています。

- 関連付けられているテキスト ボックスが無効のときは、スピン コントロールも無効にします。スピン コントロールは補助的な入力方法であり、単独で入力できるようにはしません。

値

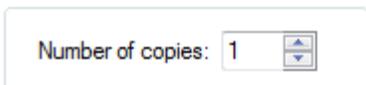
- 上のボタンでは値を 1 単位増加させ、下のボタンでは値を 1 単位減少させるように定義します。単位は 1 が通常ですが、値の変化に共通する最小の単位にする必要があります。スピン コントロールの理想は、すべての有効値に対応することと、テキストを入力するよりも便利であることです。

Margins  
Top:   Bottom:    
Left:   Right:    
Gutter:   Gutter position:

A screenshot showing margin settings. It includes four spinners for Top, Bottom, Left, and Right, each with a text input field and up/down arrow buttons. It also includes a spinner for "Gutter" with a text input field and a dropdown menu set to "Left".

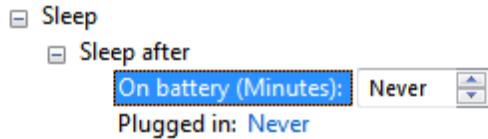
この例では、スピン コントロールをクリックすると値が .1 变化します。これは、値の変化に共通する最小の単位です。これより小さな単位を使用する場合は、有効値の範囲により細かく対応することができますが、スピン コントロールは使用できなくなります。

- スピン コントロールを使用して、入力を有効値に制限します。スピン コントロールを使用すると、無効な値が入力されることはありません。
- 有効値の範囲の最後になったら、範囲の先頭に戻ります。スピン コントロールのイメージは、ユーザーが値を輪(ホイール)のように回転させることです。そのため、スピン コントロールはホイールのような動作になります。
  - 例外: 範囲の先頭に戻ると明らかに正しくない値になる場合は、先頭には戻りません。



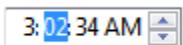
この例では、範囲の先頭の値 (最大値) は明らかに正しくないため、下の矢印ボタンをクリックしても範囲の先頭には戻りません。

- 特別な数値の場合は代わりにテキストを使用します。ユーザーが特別な値をスピンできるようにすると、ユーザーは値を把握してから入力するという手間を省くことができます。



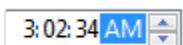
この例では、[なし] という特別な値をスpinできるようにしています。

- 値に区切りがある場合、関連付けられているテキスト ボックスには、複数の入力フォーカス ポイントを用意します。こうすると、それぞれの数値部分を個別に操作できます。



この例では、時、分、秒、および午前/午後のうちフォーカスがある部分において、スピン コントロールを使用して値を変更できます。

- 値に単位が含まれる場合、単位の変更にも同様にスpin コントロールを使用できます。



この例では、単位の変更にスpin コントロールを使用できます。

## ラベル

- テキスト ボックスのラベルに関するガイドラインに従って、関連付けられているテキスト ボックスのラベルを指定します。スpin コントロールに直接ラベルは付きません。

## ドキュメント

スpin コントロールに言及するときは、以下のことに留意します。

- ユーザー向けドキュメントでは、スpin コントロールではなく、関連付けられているテキスト ボックスのラベルを指し示します。
- スpin コントロールおよびスpin ボックスに言及するのは、プログラミングおよびその他の技術文書内のみとします。

例: [日付] ボックスで、日付を入力するか、変更する日付部分を選択して変更します。

## ステータスバー

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

全般

提示方法

アイコン

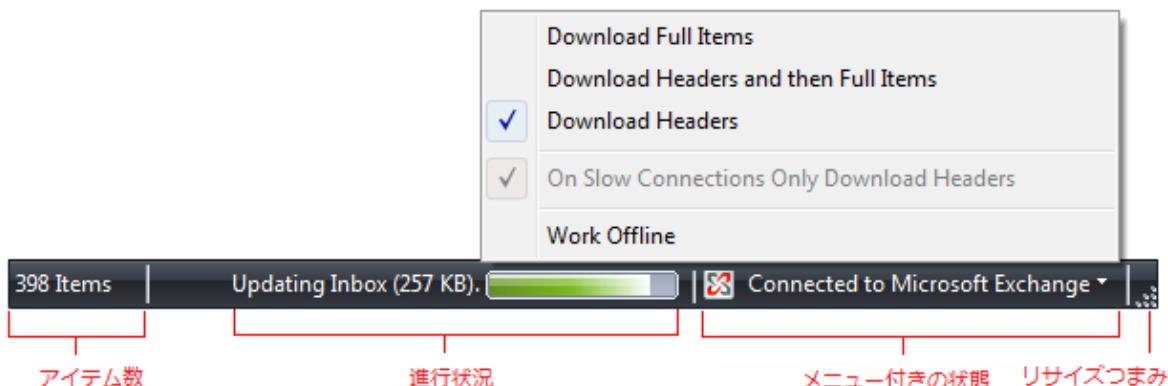
対話操作

テキスト

ドキュメント

"ステータスバー"は、メインウィンドウの下部にある領域です。ステータスバーには、現在のウィンドウの状態(現在表示されている内容、表示方法など)、バックグラウンドタスク(印刷、スキャン、書式設定など)、またはその他のコンテキスト情報(選択内容、キーボードの状態など)が表示されます。

通常、ステータスバーでは状態を示すためにテキストやアイコンが使用されますが、進行状況のインジケーター、状態に関するコマンドやオプションのメニューが使用されることもあります。



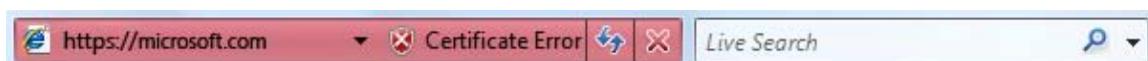
### 典型的なステータスバー

注: [通知領域](#)に関するガイドラインは、別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- ユーザーが他のプログラムをアクティブに使用しているときにも、同じ状態が当てはまるかどうか。該当する場合は、[通知領域アイコン](#)を使用します。
- その状態項目について、通知を表示する必要があるかどうか。該当する場合は、通知領域アイコンを使用する必要があります。
- ウィンドウがメインウィンドウかどうか。該当しない場合は、ステータスバーは使用しません。ダイアログボックス、ウィザード、コントロールパネル、プロパティシートにステータスバーは使用しません。
- 主に状態に関する情報かどうか。該当しない場合は、ステータスバーは使用しません。ステータスバーを、2次的な[メニューバー](#)または[ツールバー](#)として使用することはできません。
- 選択されたコントロールの使用方法を説明する情報かどうか。該当する場合は、補足説明や説明ラベルを使用して、関連コントロールの横に情報を表示します。
- 有用で関連性のある状態かどうか。つまり、ユーザーがその情報を見て対応を変える可能性が高いかどうか。該当しない場合は、状態を表示しないで済ませるか、ログファイルに書き込みます。
- 重大ですぐに対応が必要な状態かどうか。該当する場合は、[ダイアログボックス](#)を使用するか、メインウィンドウ自体の中で、注意を喚起し目立つように情報を表示します。



Windows® Internet Explorer® の赤色のアドレスバー

- 主に初級ユーザーを想定したプログラムかどうか。一般的に経験が浅いユーザーは、ステータスバーに気付かないことが多いため、ステータスバーの使用は見合わせます。

## デザイン コンセプト

ステータスバーは、ユーザーの邪魔になったり作業の流れを止めたりすることなく状態情報を通知できる有効な手段です。ただし見落とされがちでもあり、実際ステータスバーにまったく気が付かないユーザーも数多くいます。

この問題に対処するには、目立つアイコン、アニメーション、点滅などによってユーザーの注意を引こうとするのではなく、ステータスバーの限界を知ったうえで設計することです。具体的には次のことに留意します。

- 確実に有用で関連性の高い状態情報をだけに使用します。該当しない場合は、ステータスバーは使用しません。
- 重大な情報にはステータスバーを使用しません。ステータスバーに表示する情報は、ユーザーが認識しなくても問題ない情報です。ユーザーに本当に知らせる必要がある情報は、ステータスバー以外の方法で表示します。

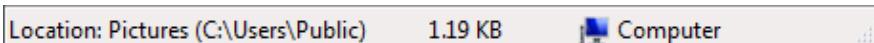
### 最も重要な点

ステータスバーには、重大ではないものの、確実に有用で関連性の高い情報を表示します。

## 使用パターン

ステータスバーにはいくつかの使用パターンがあります。

### 現在のウィンドウの状態



この例では、ステータスバーにドキュメントのパスが表示されています。

### 表示内容のソースと、表示モードが設定されている場合は表示モードを表示します。

### 進行状況

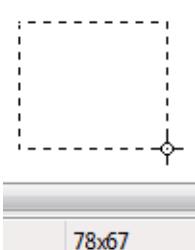
バックグラウンドタスクの進行状況を、確定型の進行状況バーまたはアニメーションで表示します。



この例では、ステータスバーに、Windows Internet Explorer ウィンドウへの Web ページの読み込みを示す進行状況バーが表示されています。

### コンテキスト情報

ユーザーが行っている操作についてのコンテキスト情報を表示します。



この例では、Microsoft Paint での選択部分のサイズがピクセルで表示されています。

## ガイドライン

### 全般

- ステータスバーの情報を必要とするユーザーがいる場合に限り、[表示] メニューの [ステータスバー] を提供するようにします。ステータスバーを必要とするユーザーがほとんどいないと思われる場合、ステータスバーは既定で非表示にします。
- メニューバーの項目の説明に、ステータスバーは使用しません。この方法で説明を表示しても見つけにくいため、助けにはなりません。

### 提示方法

- 該当しないモーダルステータスは無効にします。モーダルステータスにはキーボードおよびドキュメントの状態も含まれます。

- 該当しないモードレス ステータスは削除します。
- 状態情報は、現在のウィンドウの状態、進行状況、コンテキスト情報の順序で表示します。

## アイコン

- 状態アイコンのデザインはわかりやすいものにします。正方形や長方形のアイコンよりも、独自の形をしたアイコンを使用するようにします。
- 赤、黄、緑の純色が大部分を占めるアイコンは、状態情報を知らせる目的のみに使用します。それ以外の目的でこのようなアイコンを使用すると、混乱を招きます。

正しい例:



間違った例:



間違った例の赤色のアイコンは、エラーを意図していなくてもエラーと受け取られやすいため、混乱の元になります。

- 状態や状態の変化を示すには、アイコンのバリエーションまたはオーバーレイを使用します。量や強度などの変化を示すには、アイコンのバリエーションを使用します。その他の種類の状態を示すには、以下の標準的なオーバーレイを使用します。

オーバーレイ	状態
⚠	警告
✖	エラー
✗	無効または切断
🚫	ブロックまたはオフライン

- 状態の変更は最小限にとどめます。ステータスバー アイコンは、目立つたり、頻繁に変化したり、注意を引くものにならないようにします。視界の隅で変更があっても目は敏感に反応するため、状態の変更はわずかなものにとどめる必要があります。
- 重要な状態情報を知らせるアイコンには、できるだけインプレース ラベルを使用します。
- ラベルのないステータスバー アイコンには、ツールヒントが必要です。

詳細については、「[アイコン](#)」を参照してください。

## 対話操作

- ステータスバー領域は対話可能にして、ユーザーが関連コマンドやオプションに直接アクセスできるようにします。
  - メニュー ボタンや分割ボタンのような外観と機能を備えたコントロールを使用します。このようなステータスバー領域には、クリックできることを示すドロップダウン矢印が必要です。
  - マウスの左ボタンがクリックされマウスが押し下げられたときにメニューを表示します（マウスのボタンが元に戻ったときではないことに注意）。
  - 右クリックまたはダブルクリックはサポートしません。ユーザーはステータスバーで右クリックやダブルクリックを使った対話操作ができると考えないため、通常はこの2つを試しません。
- ポイントされたときにヒントを表示します。

## テキスト

- 一般的に、簡潔なラベルを使用し、省略できるテキストは省略します。
- 語句をできるだけ使用し、末尾に句読点は付けません。短い語句にならない場合に限り、文を使用し、末尾の句点を付けます。
- 進行状況ラベルを付ける場合は、"コピーしています..." または "コピー中..." のように処理内容を省略記号と共に示します。複数のステップがある処理、または複数のオブジェクトを処理する場合に、このラベルを動的に変更できます。
- ステータスバーのテキストを強調するために、色付き文字、太字、斜体は使用しません。
- ツールヒントの語句のガイドラインについては、「[ツールヒントと情報ヒント](#)」を参照してください。

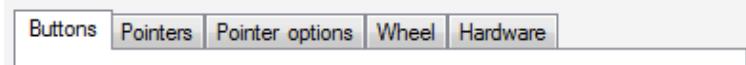
## ドキュメント

ステータスバーについて言及するときは、"ステータス行" やその他の類似表現ではなく、"ステータスバー" という語を使用します。たとえば、"現在のページ番号は、ステータスバーに表示されています。" と記述します。

## タブ

適切なコントロールかどうかの判断基準  
使用パターン  
ガイドライン  
全般  
対話操作  
アイコン  
動的なウィンドウ領域のパターン  
複数ビューおよび複数ドキュメントのパターン  
排他的なオプションのパターン  
ラベル  
ドキュメント

"タブ" を使用すると、関連する情報をラベル付きの個別ページで表示できます。



### 典型的なタブ セット

タブは、通常プロパティ ウィンドウとの間で関連付けられますが、どのような種類のウィンドウでも使用できます。

タブコントロールは、ファイルキャビネット内の情報整理に使用される、米国式のタブ付き紙フォルダーをイメージしたものです。

**注:** レイアウト、タブメニュー、ダイアログボックス、プロパティ ウィンドウに関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- すべてのコントロールが、適切なサイズの単一ページに無理なく収まるかどうか。該当する場合は、単一ページを使用します。
- タブが1つだけかどうか。該当する場合は、単一ページを使用します。
- タブどうしに明白な関連があるかどうか。該当しない場合は、情報を別々の関連情報ウィンドウに分割します。
- 設定用のタブの場合、各ページの設定が完全に独立しているか。あるページの設定を変更すると他のページの設定にも影響が生じるなど、設定が独立していない場合は、タスクページまたは`ウィザード`を使用します。
- タブはほぼ並列関係にあるか、それとも階層的な関係にあるか。階層的な関係にある場合は、`段階的表示`または`子ダイアログボックス`を使用して関連情報を表示します。
- 1つのタスクに含まれる手順を表示するものかどうか。外観から手順であることがわかるということと、次の手順に進むわかりやすい代替手段 ([次へ] ボタンなど) があるということを条件として、1つのタスクに含まれる手順を"タブ"で表示できます。それ以外の場合、手順を示すには、`ページフロー`または`ウィザード`のページを使用します。省略可能な手順は、モーダル`ダイアログボックス`を使用して表示します。
- 同じデータを別のビューで表示するかどうか。該当する場合は、`分割ボタン`または`ドロップダウンリスト`を使用してビューを変更します。タブを使用するとビューを効率的に変更できますが、分割ボタンやドロップダウンリストを使用する方が動作が軽くなります。

### 使用パターン

タブにはいくつかの使用パターンがあります。

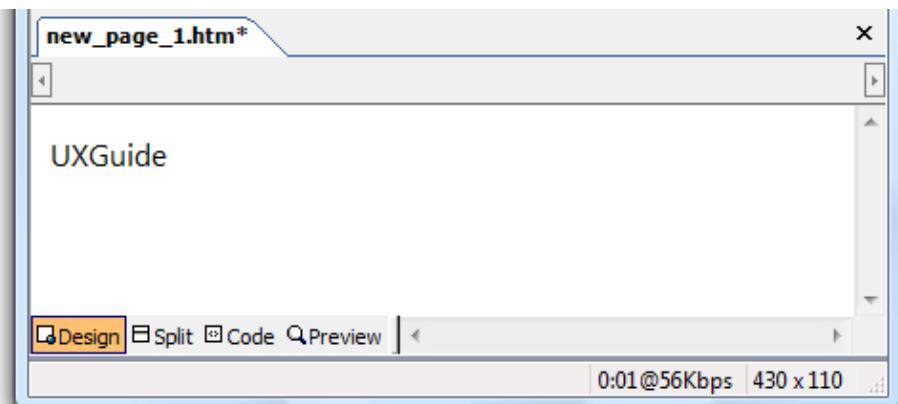
**動的なウィンドウ領域** このパターンでは、单一のスクロール可能な領域に一列に並んだタブを、ラベルを見出しにして使用するようなイメージになります。

スクロールバーのように、関連情報を表示する



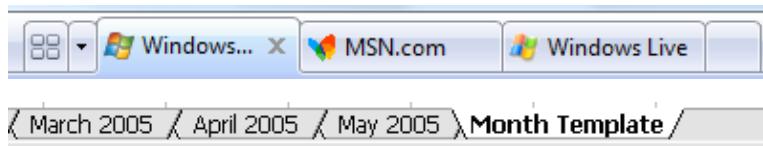
**ウィンドウ領域を拡大します。** この例では、タブによりウィンドウ領域が効率的に拡大されます。

複数ビュー  
分割ボタンやドロップダウンリストのように、同一または関連する情報を異なるビューで表示します。



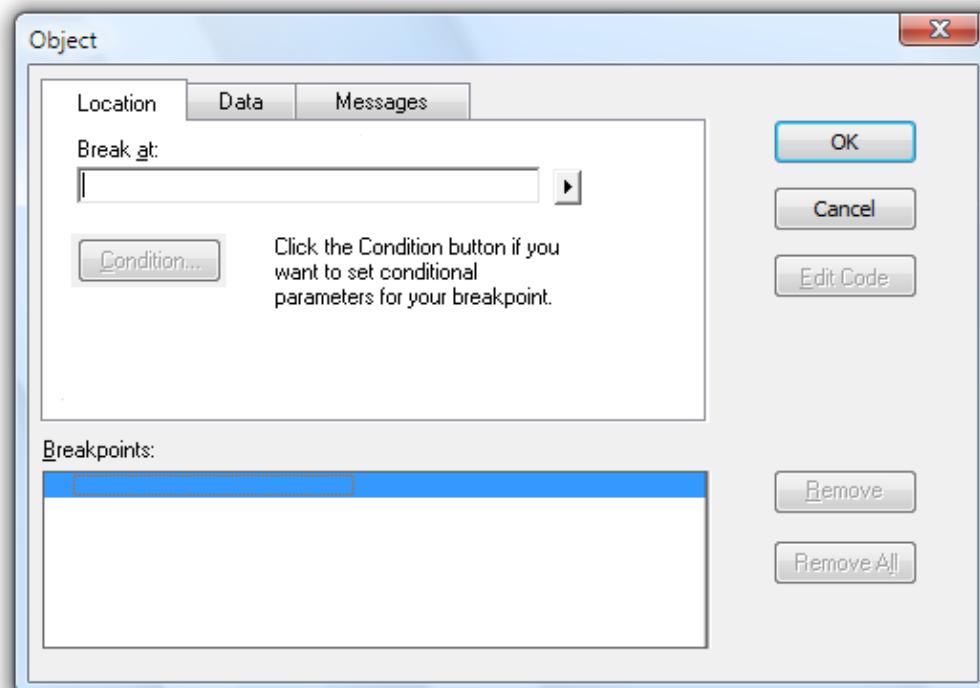
この例では、タブにより 1 つのドキュメント内でビューが切り替えられます。

複数ドキュメント  
複数のウィンドウのように、異なるドキュメントを单一のウィンドウ内で表示する。



この例では、タブにより、異なるドキュメントが単一のアプリケーション ウィンドウ内で表示されます。

排他的なオプション  
ラジオ ボタンのように、複数の排他の選択肢を表示します。このパターンでは、選択されたタブだけが適用され、他のすべてのタブは無視されます。



この例では、タブがラジオ ボタンの代わりに使用されています(これは正しい使用法ではありません)。

このパターンは推奨されません。この場合、非標準的な動作になります。このタブは、ウィンドウ内を移動する純粋な手段としてではなく、設定として動作しています。

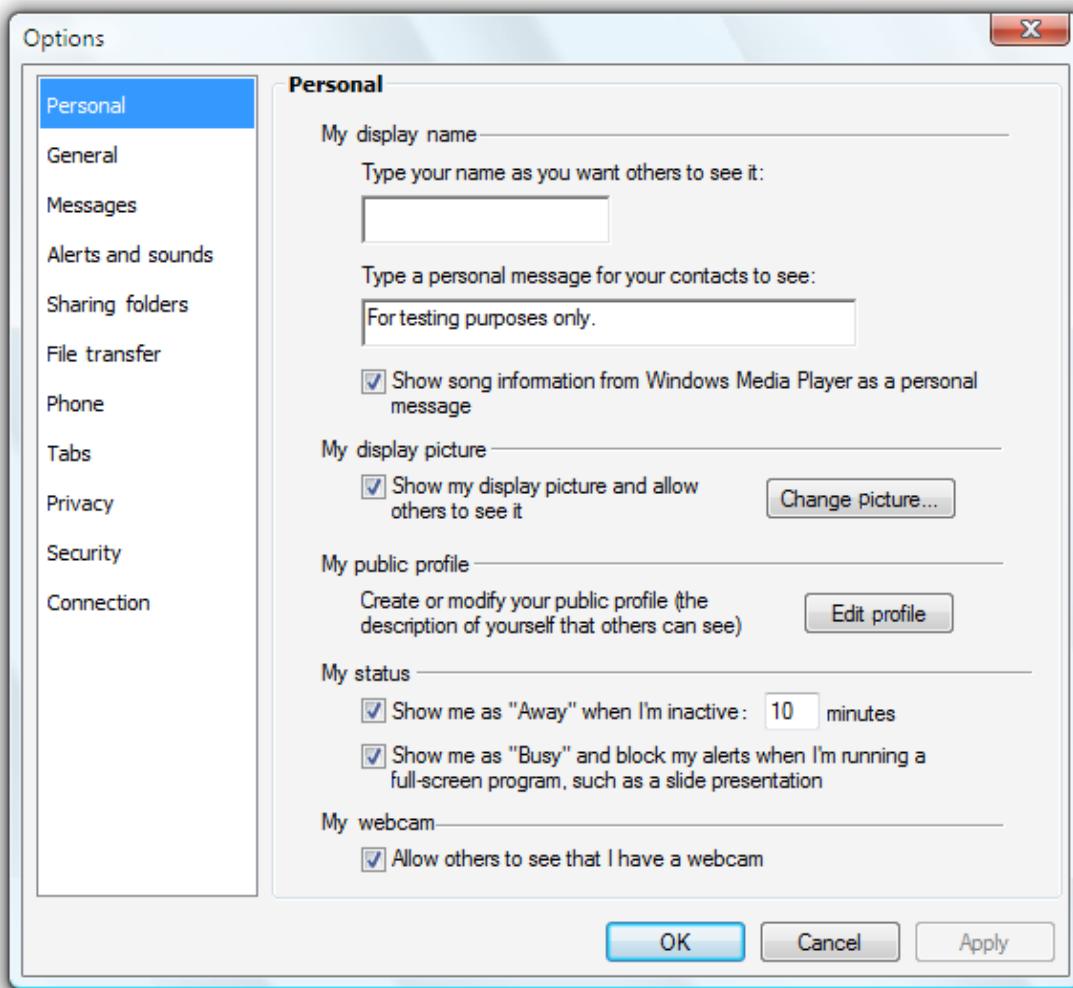
最も重要な点  
各タブの情報が関連し、ページごとの設定が独立していることを確認します。最後に選択されたタブが、特別な意味を持たないようにします。

## ガイドライン

### 全般

- 次のような場合は、水平タブを使用します。
  - ウィンドウのタブが 7 個以下の場合。
  - ユーザー インターフェイス (UI) がローカライズされても、すべてのタブが 1 行に収まる場合。

- 次のような場合は、垂直タブを使用します。
  - プロパティ ウィンドウのタブが 8 個以上の場合。
  - 水平タブを使用すると 2 行以上になる場合。



この例では、垂直タブによって 8 個以上のタブを使用できます。

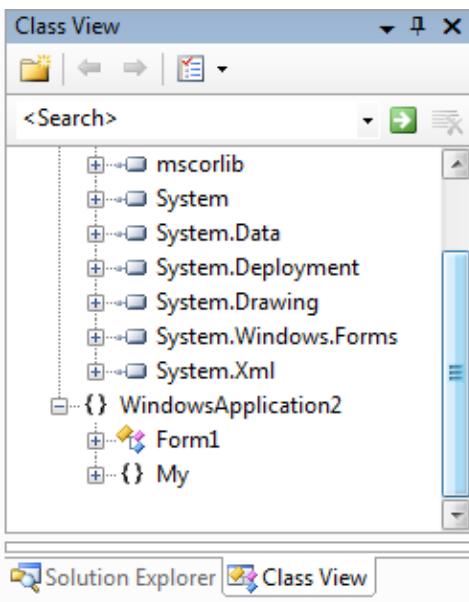
- タブを入れ子にしたり、水平タブと垂直タブを組み合わせることはしません。代わりにタブの数を減らすか、垂直タブのみを使用するか、またはドロップダウンリストなど別のコントロールを使用します。
- 水平タブはスクロール型にしません。水平スクロールは見つけにくいものです。垂直タブはスクロール型にしてもかまいません。

間違った例:



この例では、水平タブがスクロール型になっています。

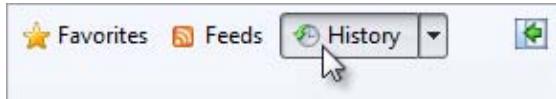
- タブがサイズ変更可能なウィンドウ上にある場合で、スクロールが必要になる場合は、ウィンドウではなくページにスクロールバーを付けます。タブは常に見える状態にして、スクロールしても隠れないようにします。



この例では、ウィンドウではなくタブ ページにスクロールバーが付いています。

- ひとめでタブとわかるようにして、別の種類のコントロールに見間違えられないようにします。

間違った例:



この例では、タブがコマンド ボタンのように見えます。

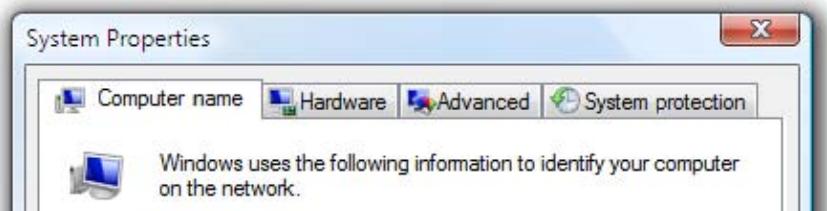
#### 対話操作

- 1つのページだけに適用されるコントロールは、タブ ページの枠内に配置します。
- ウィンドウ全体に適用されるコントロールは、タブ ページの外側に配置します。
- タブの変更に対して効果を割り当てるとはしません。タブはどのような順序でもアクセスできる必要があります。現在のタブを変更しても、副次的な影響が発生したり、設定が適用されたり、エラー メッセージが発生したりしないようにします。
- 最後に選択されたタブに、特別な意味を割り当てるとはしません。タブの選択は単なる移動であり、ユーザーが最後に選択したタブで何らかの設定が行われないようにします。
- 1つのページの設定は、他のページの設定とは独立したものになります。依存する設定どうしは、同じページ上に配置します。
- ユーザーが最後に表示したタブからまた作業を開始する可能性が高い場合は、そのタブが既定で選択されるよう、状態を維持します。ウィンドウごと、ユーザーごとに設定を維持し、該当しない場合は既定で最初のページが選択されるようにします。

#### アイコン

- タブ上にアイコンは配置しません。アイコンは、通常、煩雑になりがちで、画面領域を消費し、多くの場合は理解の向上につながりません。アイコンは、標準記号のように理解の補助になるものだけを使用します。

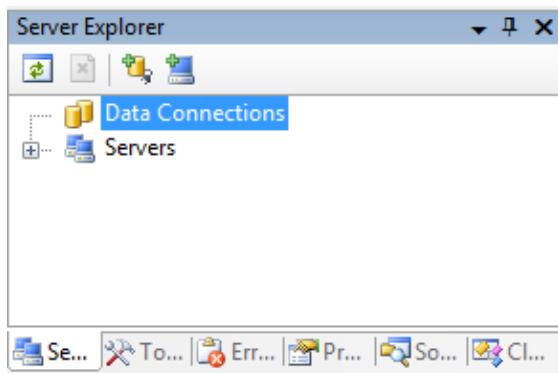
間違った例:



この例では、アイコンは煩雑さを生むだけで、ユーザーの理解の助けにはなっていません。

例外: 意味のあるラベルを表示するスペースがない場合は、明確に認識できるアイコンを使用できます。

正しい例:



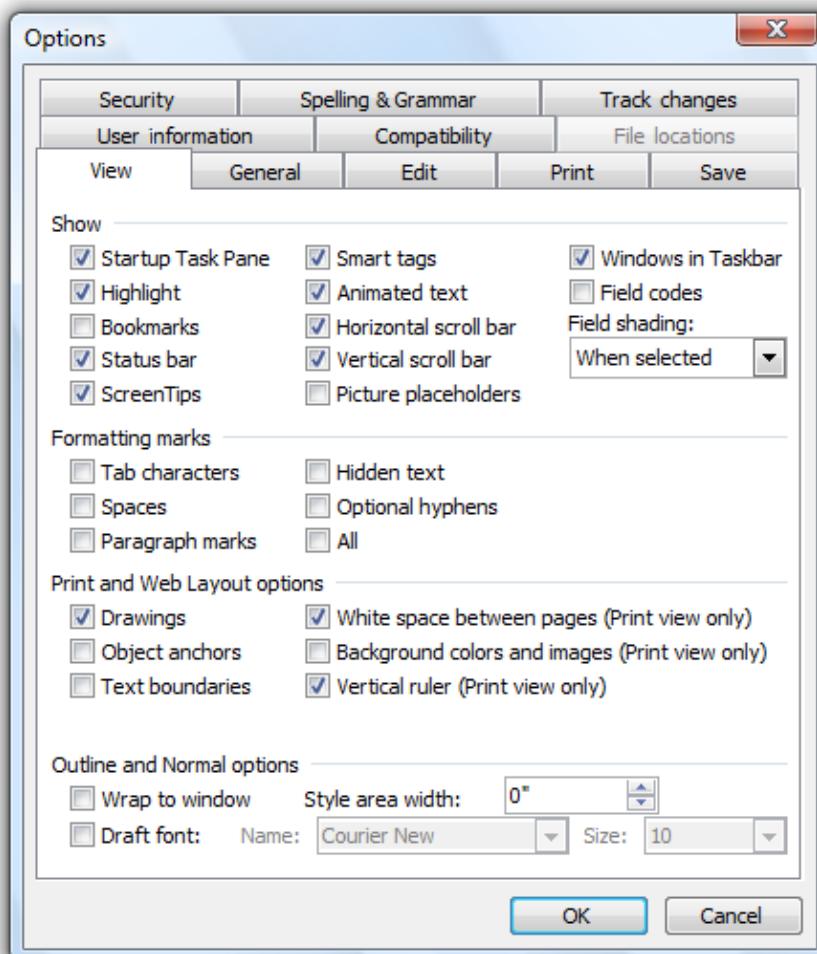
この例では、ウィンドウの幅が狭いため、ラベルよりもアイコンの方がタブを認識しやすいといえます。

- タブのグラフィックに製品ロゴは使用しません。タブはブランド化のためのものではありません。

#### 動的なウィンドウ領域のパターン

- タブページでスクロールバーは使用しません。タブは、ウィンドウの有効領域を拡大するという点でスクロールバーと類似しており、タブがあればスクロールバーは不要です。
- 簡潔なタブラベルを使用します。ページのコンテンツを1～2語あるいは6文字程度で明確に説明します。ラベルがローカライズされる場合は特に、長いラベルは画面領域を消費することになります。
- 限定的で意味のあるタブラベルを使用します。“全般”、“詳細設定”、“設定”など、どのタブにも当てはまるような汎用的なタブラベルの使用は避けます。
- 現在のコンテキストに該当しない、ユーザーが想定しないタブは削除します。不要なタブを削除することで、簡潔なUIになります、見落としがなくなります。

間違った例:

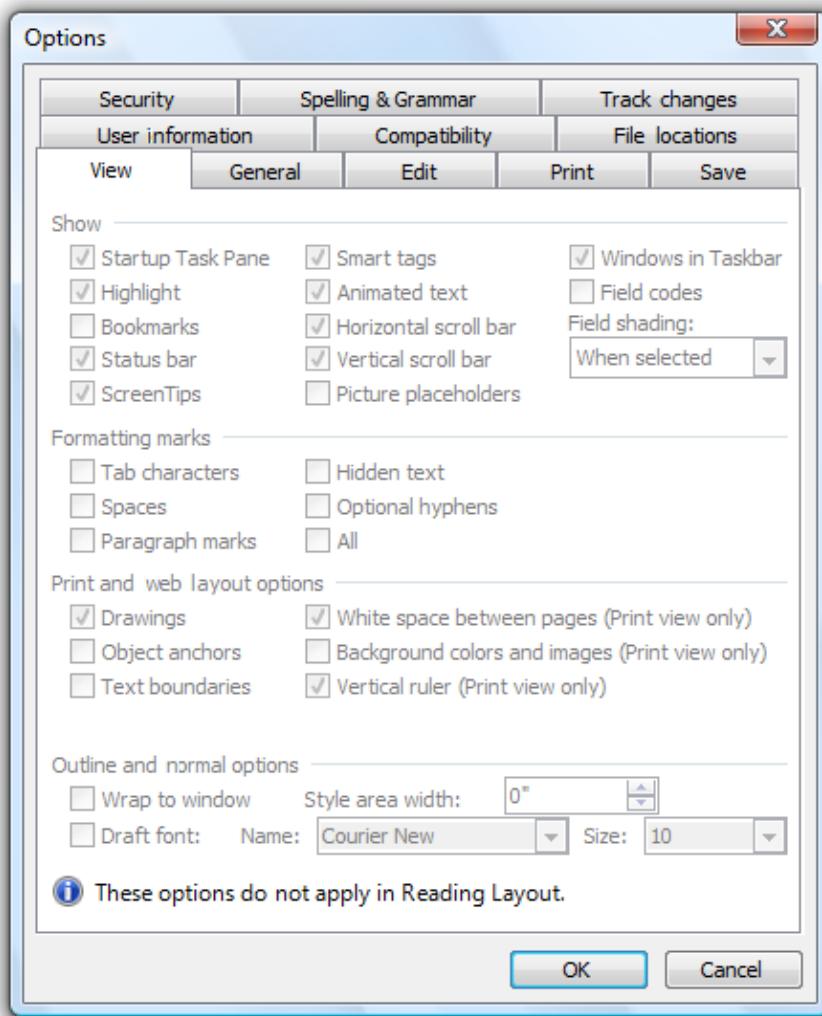


この例では、Microsoft® Word が電子メールエディターとして使用されている場合に、[ファイルの場所]

タブが無効にされていますが、これは間違います。このタブは無効にするのではなく、削除する必要があります。ユーザーはこのコンテキストにおいて、ファイルの場所を確認したり変更する操作は想定しません。

- 現在のコンテキストに該当しなくても、ユーザーが次のことを想定している可能性がある場合は、タブを無効にしません。
  - タブが表示される。
  - ページ上のコントロールを無効にできる。
  - コントロールが無効になっている理由を説明するテキストが表示される。

このような操作を行おうとするときにタブが無効になっていると、ユーザーにはその理由がわからず、また調べることもできず、特定の値を探すときに他のすべてのタブを確認しなくてはならなくなります。



この例では、閲覧レイアウトに [表示] のオプションはいずれも該当しませんが、ユーザーはタブラベルを見て何らかのオプションが適用される可能性があります。このため、ページは表示され、オプションは無効になっています。

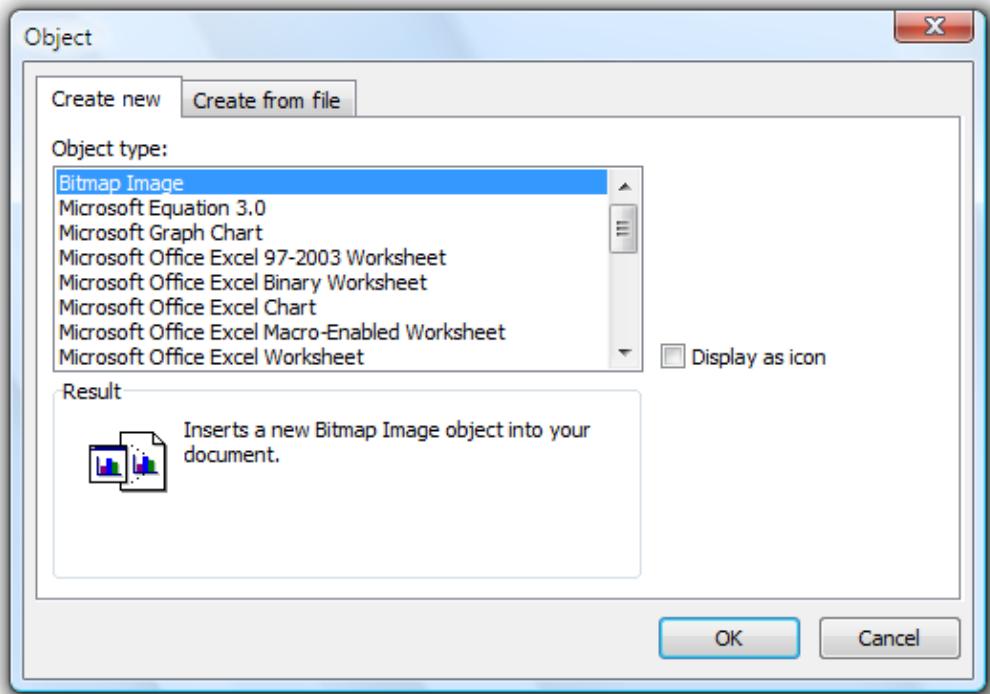
#### 複数ビューおよび複数ドキュメントのパターン

- ビューノードまたはドキュメント名をタブラベルに使用します。
- 長すぎるタブ名は避けます。必要に応じて、最大文字数の名前にするか、省略記号を使用して表示範囲内に収めます。ラベルがローカライズされる場合は特に、長いラベルは画面領域を消費することになります。
- 現在のコンテキストに該当しないタブは削除します。

#### 排他的なオプションのパターン

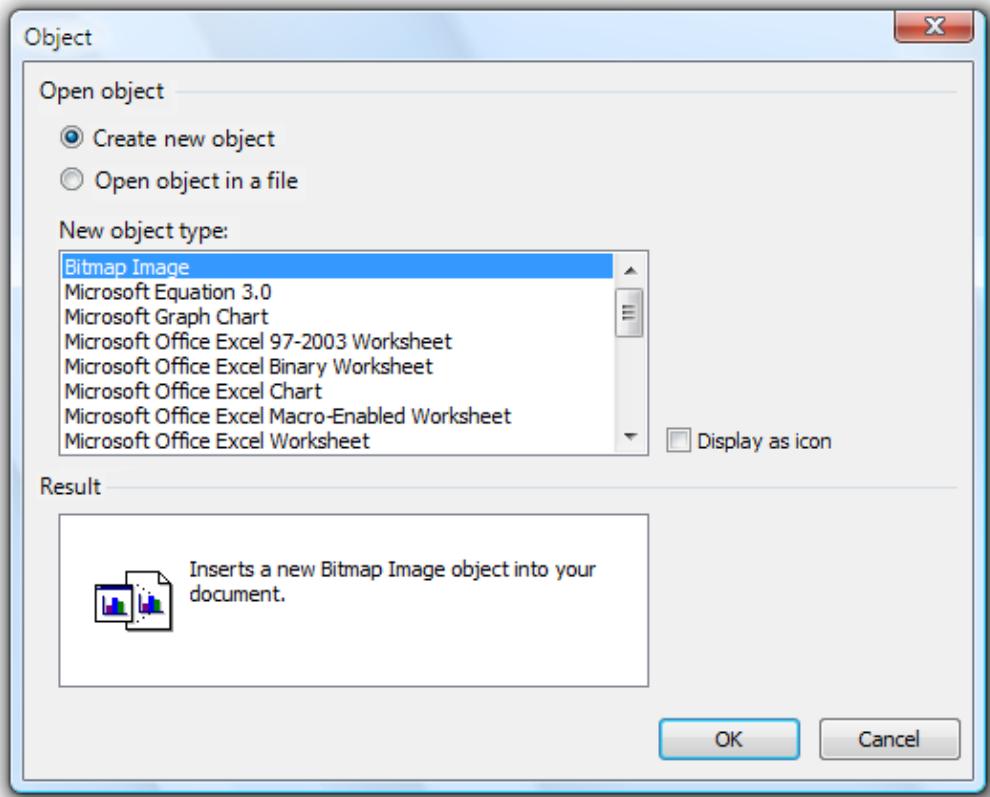
- このパターンは使用しません。代わりに、ラジオボタンまたはドロップダウンリストを使用します。

間違った例:



この例では、タブがラジオ ボタンの代わりに使用されていますが、正しい使用方法ではありません。

正しい例:



この例では、タブの代わりにラジオ ボタンが正しく使用されています。

## ラベル

- パターンに応じて、タブにラベルを指定します。動詞ではなく名詞を使用し、末尾に句読点は付けません。詳細については、前に挙げたパターンのガイドラインを参照してください。
- センテンススタイルの大文字化**を使用します。
- アクセスキー**は割り当てません。タブには専用のショートカットキーからアクセスできます (Ctrl + Tab キー、Ctrl + Shift + Tab キー、Ctrl + PageUp キー、Ctrl + PageDown キー)。有用なアクセスキーの選択には限りがあるので、タブにアクセスキーを割り当てずにおけば、他のコントロールへのアクセスキーの割り当てが容易になります。

## ドキュメント

タブに言及するときは、以下のことに留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用し、"タブ" という語を追加します。
- ユーザー操作を説明する場合は、"クリック" を使用します。
- ラベルは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。
- "タブ" という単独の名詞は、タブコントロールを指すためだけに使用します。同じ単語で他の対象を指すと、特に各国語にローカライズされる場合は、あいまいになる可能性があります。タブコントロール以外の対象は、"Tab キー"、"タブ位置"、"ルーラー上のタブ記号" のように限定し、意味がはっきりとわかるようにします。

例: [ツール] メニューの [オプション] をクリックし、次に [表示] タブをクリックします。

# テキスト ボックス

適切なコントロールかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

全般

編集可能なテキストボックス

数値テキストボックス

パスワードおよび PIN 入力

テキスト出力

データ出力

入力検証とエラー処理

プロンプト

推奨されるサイズと間隔

ラベル

ドキュメント

"テキスト ボックス" を使用すると、テキストまたは数値を、表示、入力、編集できます。

ラベル – *Display name:*

テキスト –

ボックス

典型的なテキスト ボックス

注: レイアウト、フォント、バルーンに関するガイドラインは、それぞれ別の項目として記載しています。

## 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- すべての有効値を例挙することが実際的かどうか。該当する場合は、代わりに单一選択リスト、リストビュー、ドロップダウンリスト、編集可能なドロップダウンリスト、またはスライダーを使用するようにします。
- 有効なデータは、完全に制約がないデータ、または、(長さや文字の種類など) 書式のみに制約があるデータのどちらかに該当するか。該当する場合は、テキスト ボックスを使用します。
- 専用のコモン コントロールが関連付けられるデータ型の値かどうか。たとえば、日付、時刻、IPv4 または IPv6 アドレスなどの該当する場合は、テキスト ボックスではなく、日付コントロールなどの適切なコントロールを使用します。
- データが数値の場合は、以下のことから判断します。
  - ユーザーが設定を相対量として考えるかどうか。該当する場合は、スライダーを使用します。
  - 設定変更の効果に対するフィードバックをすぐに得られることがユーザーの役に立つかどうか。該当する場合は、スライダーを使用します。テキスト ボックスを付けてもかまいません。たとえば、スライダーを使用すると、色相、彩度、輝度などの値変更の効果をすぐに確認できるので、色の選択がより簡単になります。

## デザインコンセプト

テキスト ボックスには、どんなものにも柔軟に対応できるというメリットがある一方、最小限の制約しか適用できないというデメリットもあります。編集可能なテキスト ボックスの制約は、次の 3 つだけです。

- オプションで、最大文字数を設定できる
- オプションで、入力を数値 (0 ~ 9) のみに制限できる
- スピン コントロールを使用する場合、スピン コントロールの選択肢を有効値に制限できる

テキスト ボックスには、長さやスピン コントロールの有無を別にすると、有効値や値の形式を示す視覚的な手がかりはありません。ユーザーにこのような情報を伝えるには、ラベルが必要になります。また、ユーザーが無効なテキストを入力した場合は、エラー メッセージを使用してエラーを処理する必要があります。

一般的なルールとしては、使用できる中で最も制約のあるコントロールを使用します。テキスト ボッ

クスのような制約のないコントロールを使用するのは、最後の手段にします。ただし制約を考える際に  
は、ローカライズを想定する必要があります。たとえば、米国の郵便番号を入力するように制約するコ  
ントロールは、ローカライズ向きではありません。一方、どのような郵便番号形式も受け入れる制約の  
ないテキスト ボックスは、ローカライズにも対応します。

## 使用パターン

テキスト ボックスは、さまざまな用途に対応できる、柔軟性のあるコントロールです。

<p>データ入力 短い文字列の入 力または編集用 の、制約のない 1 行テキスト ボックス</p> <p>特定形式での データ入力 特定形式での データ入力用 の、固定サイズ の短い 1 行テキ スト ボックスの セット</p> <p>補助付きデータ 入力 文字列の入力ま たは編集用の制 約のない 1 行テ キスト ボックス と、ユーザーが 有効値を選択す る助けとなるコ マンド ボタンの 組み合わせ</p> <p>テキスト入力 長い文字列の入 力または編集用 の、制約のない 複数行テキスト ボックス</p> <p>数値入力 数値の入力また は編集用、数 値専用 1 行テキ スト ボックス と、マウス入力 の助けとなるオ プションのスピ ン コントロー ルの組み合わせ</p> <p>パスワードおよ び PIN 入力 パスワードおよ び PIN の安全な 入力用、制約 のない 1 行テキ スト ボックス</p>	<p>Display name: <input type="text"/></p> <p>Product key: <input type="text"/> <input type="text"/> <input type="text"/> <input type="text"/></p> <p>特定形式でのデータ入力用テキスト ボックス 注: <a href="#">自動終了</a>機能を使用すると、1つのテキスト ボックスへの入力が終わったら自動的に次のテキスト ボックスへと入力フォーカスが移動します。この方法の欠点は、データを一括でコピーまたは貼り付け できることです。</p> <p>Backup file location: <input type="text"/> <input type="button" value="Browse..."/></p> <p>この例では、<a href="#">[参照]</a> コマンドにより、ユーザーは有効値を選択できます。</p> <p>Address: <input type="text"/></p> <p>制約のない複数行テキスト ボックス</p> <p>Wait: <input type="text" value="30"/> <input type="button" value="Up"/> <input type="button" value="Down"/> minutes</p> <p>数値入力用テキスト ボックス テキスト ボックスとそれに関連付けられているスピン コントロールの組み合わせは、<a href="#">スピニング ボック ス</a>と呼ばれます。</p> <p>Password: <input type="password"/></p> <p>パスワード入力用テキスト ボックス</p>
--	--

データ出力 短い文字列の表 示用の、読み取 り専用の 1 行テ キスト ボックス (常に枠なしで表 示)	静的テキストとは違い、テキストボックスを使用して表示されるデータは、スクロール、選択、コピーが可能です。スクロールはデータがコントロールより長い場合に便利です。
テキスト出力 長い文字列の表 示用の、読み取 り専用の複数行 テキスト ボックス	Location: C:\Desktop\Windows\UXGuide\Windows Vista Use Location: UXGuide\Windows Vista User Experience Guidelines

データ表示用の、読み取り専用の 1 行テキスト ボックス

Privacy information:

When you sign in to People Near Me and then use a collaborative program such as Windows Meeting Space, a list of available people is displayed. (If no one is available, no names appear.) Display names, computer names, and IP addresses are the only pieces of information visible to everyone signed in to People Near Me.

Two people can have the same display name, so before



データ表示用の読み取り専用テキスト ボックス

## ガイドライン

### 全般

- テキストボックスを無効にする場合は、関連付けられたラベル、指示ラベル、スピンコントロール、コマンドボタンも無効にします。
- 繰り返し使用される可能性の高いデータがある場合は、オートコンプリートを使用して、ユーザーの入力の負担を軽減します。データの例としては、ユーザー名、アドレスまたは住所、ファイル名が挙げられます。ただし、パスワード、PIN、クレジットカード番号、医療情報などの機密情報が入力される可能性のあるテキストボックスには、オートコンプリート機能は使用しません。
- 不要なスクロール操作が発生しないようにします。テキストボックスよりも大きなデータが予想され、レイアウトを崩すことなくテキストボックスを大きくできる場合は、スクロールが不要になるようにボックスのサイズを調整します。

間違った例:

Computer name:  
Jonathan-01.computer.

この例では、データに対応できるようにテキストボックスを長くする必要があります。

- スクロールバーについては、次のことに留意します。
  - 複数行テキストボックスに、水平スクロールバーは使用しません。代わりに垂直スクロールと行の折り返しを使用します。
  - 1行テキストボックスに、スクロールバーは一切使用しません。
- 数値入力には、スピンコントロールを使用できます。テキスト入力には、代わりにドロップダウンリストまたは編集可能なドロップダウンリストを使用します。
- 特定形式でのデータ入力以外には、自動終了機能は使用しません。自動的に入力フォーカスが移動すると、ユーザーを驚かせてしまうことがあります。

### 編集可能なテキストボックス

- 可能な場合は、入力テキストの長さを制限します。たとえば、有効な入力値が 0 から 999 までの数値の場合は、入力を 3 文字に制限する数値テキストボックスを使用します。特定形式でのデータ入力用テキストボックスは、すべて固定長の短いボックスにします。
- 各種データ形式に柔軟に対応できるようにします。ユーザーがさまざまな形式でテキストを入力する可能性が高い場合は、よく使われる形式にすべて対応できるようにします。たとえば、多くの名前や数値、ID の入力には、空白や句読点も追加されることが考えられます。多くの場合、大文字と小文字は区別されません。
- 入力される可能性のある形式に対応できない場合は、特定形式でのデータ入力を使用して形式を固定するか、ラベルで有効な形式を示します。

許容される例:

Serial number: (example: 1234-56-7890)

この例では、特定形式での入力を要求しています。

より良い例:

Serial number:

1234 56 7890

この例では、特定形式でのデータ入力パターンによって、形式を固定しています。

最も良い例:

Serial number:

この例では、テキスト ボックスはあらゆる形式に対応します。

- 最大入力文字数を設定する場合は、各種形式に柔軟に対応できるようにします。たとえば、有効なクレジットカード番号は最長 19 行になる可能性があるため、これより短い文字数に制限していると、長い形式での数値の入力が困難になります。
- ユーザーが長く複雑なデータを貼り付ける可能性の方が高い場合、特定形式でのデータ入力パターンは使用しません。特定形式でのデータ入力パターンは、ユーザーがデータを入力する可能性の方が高い状況に使用します。

IPv6 address:

この例では、特定形式でのデータ入力パターンは使用されていないため、ユーザーは IPv6 アドレスを貼り付けることができます。

- ユーザーが値全体を再入力する可能性の方が高い場合は、全テキストを入力フォーカスとして選択します。ユーザーが値を編集する可能性の方が高い場合は、テキストの最後尾にキャレットを置きます。

Password:

\*\*\*\*\*

この例では、ユーザーが値全体を再入力する可能性の方が高いため、値全体が入力フォーカスとして選択されています。

Tags: Landscape; Add a tag

この例では、ユーザーがテキスト全体を置き換えるよりもキーワードを追加する可能性の方が高いため、テキストの最後尾にキャレットが置かれています。

- 改行が入力値として有効な場合は常に、複数行テキスト ボックスを使用します。
- ファイルまたはパス用のテキスト ボックスには常に [参照] ボタンを用意します。

## 数値テキスト ボックス

- 最も適切な単位を採用し、単位をラベルに表示します。たとえば、リットルではなくミリリットル(またはその逆)、直接的な数値ではなくパーセンテージ(またはその逆)を使用するようにします。

正しい例:

Quantity (liters): 0.250

この例では、単位はラベルに表示されていますが、ユーザーは小数を入力する必要があります。

より良い例:

Quantity (milliliters): 250

この例では、テキスト ボックスに、より適切な単位が使用されています。

- 役に立つ場合は常にスピン コントロールを使用します。ただし、ユーザーが大きな数値をいくつも入力する必要

がある場合などは、スピンコントロールは現実的でないことがあります。スピンコントロールは次の場合に使用します。

- 入力値が小さな数値になる可能性が高い場合(通常 100 未満)
- ユーザーが既存の数値を少し変更する可能性が高い場合
- ユーザーがキーボードよりもマウスの使用を好む可能性が高い場合
- 次の場合は常に数値を右揃えにします。
  - 複数の数値テキストボックスがある場合
  - テキストボックスが縦方向に並んでいる場合
  - ユーザーが値を追加または比較する可能性がある場合

正しい例:

Airfare: 475.00  
Hotels: 229.35  
Meals: 161.78  
Total: 866.13

この例では、数値テキストが右揃えになっており、値を比較しやすくなっています。

間違った例:

Red: 255  
Green: 0  
Blue: 255

この例では、数値テキストが左揃えになっており、見にくくなっています。

- 金額は常に右揃えにします。
- 特定の数値に特別な意味を割り当てるとはしません。アプリケーション内部で特別な意味として使用している場合でも、テキストボックスの数値に特別な意味を割り当てるとはしません。このような値は、チェックボックスまたはラジオボタンで示します。

間違った例:

Maximum cached objects (use -1 to disable caching):  
-1

この例では、-1という値に特別な意味が割り当てられています。

正しい例:

Caching  
Maximum cached objects:

この例では、特別な値がチェックボックスで示されています。

## パスワードおよび PIN 入力

- 独自にボックスを作成するのではなく、常にパスワード用のコモンコントロールを使用します。パスワードと PINには、安全な処理のために特別な対処が必要となります。

その他のガイドラインと例については、「[パルーン](#)」を参照してください。

## テキスト出力

- 複数行にわたる長い読み取り専用テキストには、背景に白のシステムカラーを使用するようにします。背景を白にすると、テキストが読みやすくなります。灰色の背景に長いテキストがあると、ユーザーは読むのを面倒に感じます。

背景色の詳細については、「[フォント](#)」を参照してください。

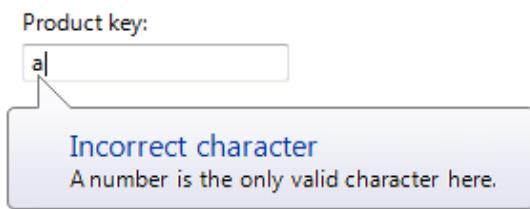
## データ出力

- 読み取り専用の1行テキストボックスには枠を付けません。枠があると、ユーザーはテキストが編集可能であるように感じます。
- 読み取り専用の1行表示のテキストボックスは無効にしません。無効にすると、ユーザーはテキストを選択してクリップボードにコピーできません。また、枠に収まりきらないデータがある場合にスクロールできません。
- ユーザーがテキストをスクロールしたりコピーする可能性がある場合を除き、読み取り専用の1行テキストボックスの上にはタブストップを適用しません。

## 入力検証とエラー処理

テキストボックスには、通常、有効な入力のみを受け入れるための制約がないため、状況に応じて入力を検証し問題を処理する必要があります。入力の問題と対処方法を以下に示します。

- ユーザーが無効な文字を入力した場合は、その文字を無視し、[入力の問題を示すバルーン](#)で有効な文字について説明します。



この例では、バルーンによって無効な入力文字であることが示されています。

- 入力データが無効な値であるか無効な形式になっている場合は、テキストボックスから入力フォーカスが移動する際に、入力の問題を示すバルーンを表示します。
- 入力データと、同じウィンドウ上の他のコントロールとの間に不整合がある場合は、入力全体が完了したとき（モーダルダイアログボックスでユーザーが[OK]をクリックしたときなど）に、エラーメッセージを表示します。

ユーザーがエラーを簡単に修正できない場合を除き、無効な入力データを消去することはしません。データを残しておくと、ユーザーは最初から入力をやり直すことなく間違いを修正できます。パスワードやPINなどは、ユーザーが簡単に修正できないので、正しくない場合は消去します。

その他のガイドラインと例については、「[エラーメッセージ](#)」および「[バルーン](#)」を参照してください。

## プロンプト

プロンプトとは、既定値としてテキストボックスの内部に置かれるラベルまたは短い説明のことです。静的テキストとは違い、ユーザーがテキストボックスに入力を開始するかテキストボックスに入力フォーカスが移動すると、プロンプトは画面から消去されます。



### 典型的なプロンプト

プロンプトは、次の場合一に使用します。

- テキストボックスがツールバーのような特別な領域にあり、ラベルや説明がない方が良い場合。
- 主にテキストボックスの目的を省スペースで示す場合。テキストボックスの使用中にユーザーが確認する必要のある重要な情報は、プロンプトでは示しません。

ユーザーに対して入力やボタンのクリックを指示するだけのプロンプトは使用しません。たとえば、「ファイル名を入力し、[送信]をクリックしてください」のようなプロンプトテキストは使用しません。

プロンプトを使用するときは、以下のことに留意します。

- プロンプトテキストは灰色で表示し、入力されたテキストは通常の黒色で表示します。プロンプトテキストと実際に入力されたテキストが混同されないようにします。
- プロンプトテキストは常に簡潔にします。文ではなく語句にすることもできます。

- センテンススタイルの大文字化を使用します。
- 末尾に句読点や省略記号は付けません。
- プロンプトテキストは編集可能にしません。ユーザーがテキストボックス内をクリックしたり、Tabキーでテキストボックスに移動したら、画面からプロンプトテキストを消去します。
  - 例外: テキストボックスに既定で入力フォーカスがある場合は、プロンプトを表示し、ユーザーが入力を開始したときに画面から消去します。
- テキストボックスから入力フォーカスが移動したときに、テキストボックスが空白のままの場合は、プロンプトテキストを復元します。

## 推奨されるサイズと間隔



## テキストボックスに推奨されるサイズと間隔

テキストボックスの幅は、入力サイズがどの程度かを示す手がかりになります。テキストボックスのサイズを設定するにあたっては、以下のことに留意します。

- 有効な最長データが適切に表示される幅を選択します。ほとんどの状況において、最も長い文字列でもユーザーがスクロールすることなく入力または表示できるようにします。
- ローカライズの対象となるすべてのテキスト(数値以外)について、30% (短いテキストの場合は最大 200%) の余白を追加します。
- 入力サイズを特に想定しない場合は、ウィンドウ上の他のテキストボックスやコントロールと幅を揃えます。
- 複数行テキストボックスは、行単位のサイズに設定します。

## ラベル

### テキストボックスのラベル

- すべてのテキストボックスにはラベルが必要です。ラベルは文ではなく語句にして、末尾にコロンを付けます。また、[静的テキスト](#)にします。

次の場合は例外です。

- 画面領域が貴重な場所に置くプロンプト付きテキストボックスの場合、ラベルは付けません。
- 特定形式でのデータ入力に使用するテキストボックスのグループは、まとめて1つのテキストボックスとしてラベルを付けます。
- ラジオボタンまたはチェックボックスに従属し、コロンで終わるラベルの後に配置されるテキストボックスには、追加のラベルを付けません。
- [メイン指示テキスト](#)と同じ内容を繰り返すようなコントロールラベルは省略します。この場合、メイン指示テキストにコロン(疑問文以外の場合)とアクセスキーを付けます。

許容される例:



この例では、テキストボックスのラベルで、メイン指示テキストと同じ内容が繰り返されています。

より良い例:

[Enter your billing address:](#)

この例では、冗長なラベルは削除され、メイン指示テキストにコロンとアクセスキーが指定されています。

- 一意な[アクセスキー](#)を割り当てます。アクセスキー割り当てのガイドラインについては、「[キーボード](#)」を参照してください。
- [センテンススタイルの大文字化](#)を使用します。
- ラベルはテキストボックスの左に配置するか、または上に配置してテキストボックスの左端に揃えます。ラベルを左に配置する場合は、ラベルのテキストとテキストボックス内のテキストの位置が水平になるようにします。

正しい例:

Name:  
Jonathan

Name: Jonathan

この例では、テキストボックスの上にあるラベルはテキストボックスの左端に揃えられ、テキストボックスの左にあるラベルはテキストボックス内のテキストと揃えられています。

間違った例:

Name:  
Jonathan

Name: Jonathan

間違った例では、テキストボックスの上にあるラベルはテキストボックス内のテキストと揃えられ、テキストボックスの左にあるラベルはテキストボックスの上端に揃えられています。

- ラベルの後ろに単位(秒、接続など)をかっこで指定することができます。
- テキストボックスで、(制約はないものの)少ない文字数を最大文字数として受け入れる場合は、ラベルで入力の最大文字数を示すことができます。テキストボックスの幅によっても、最大サイズを示します。

Password (maximum 8 characters):

この例では、ラベルで最大文字数が指示されています。

- ローカライズができないため、テキストボックスの内容(またはその単位ラベル)を文の一部にはしません。
- テキストボックスに複数アイテムの入力が可能な場合は、ラベルでアイテムの区切り方を明記します。

User names (separate names with a semi-colon):

この例では、ラベルでアイテムの区切り記号が指示されています。

- 入力が必須であることを示すためのガイドラインについては、「[必須の入力](#)」を参照してください。

## 指示ラベル

- テキスト ボックスに関する指示テキストが必要な場合は、ラベルの上に追加します。文を使用し、末尾に句点を付けます。
- センテンス スタイルの大文字化を使用します。
- 必須ではなく参考程度の追加情報は、短くまとめます。このような情報はラベルとコロンとの間に配置してかっこで囲むか、テキスト ボックスの下にかっこで囲まずに配置します。

Computer description:

For example: "Kitchen Computer" or "Mary's Computer".

この例では、追加情報がテキスト ボックスの下に配置されています。

### プロンプトのラベル

- プロンプト テキストは常に簡潔にします。文ではなく語句にすることもできます。
- センテンス スタイルの大文字化を使用します。
- 末尾に句読点や省略記号は付けません。
- ユーザーに対し、テキスト ボックス横のボタンで処理される情報の入力を求める場合は、単に横のボタンを配置するだけにします。ユーザーに対してボタンのクリックを指示するプロンプトは使用しません。たとえば、"ファイルをドロップし、[送信] をクリックしてください" のようなプロンプト テキストは使用しません。

## ドキュメント

テキスト ボックスに言及するときは、以下のことに留意します。

- 入力や貼り付けを必要とする対話操作を示す場合は、"入力する" を使用します。それ以外に、ユーザーが一覧で値を選択したり [参照] ボタンを使用するなど他の方法を使用してテキスト ボックスに情報を挿入できる場合は、"入力する" または "指定する" を使用します。
- 読み取り専用テキスト ボックス内での選択を示す場合は、"選択する" を使用します。
- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用し、"ボックス" という語を追加します。アクセスキーを示すかっこや下線付き文字、およびコロンは含めません。テキスト ボックスを示すために、"テキスト ボックス" または "フィールド" という語は使用しません。
- ラベルは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例: [パスワード] ボックスにパスワードを入力し、[OK] をクリックします。

- テキスト ボックスに特定の形式が必要な場合は、最も一般的な許容される形式だけをドキュメントに記載し、その他の形式はユーザーによる発見に任せます。データ形式に柔軟性を持たせることが、ドキュメントの複雑さにつながらないようにします。

正しい例:

部品のシリアル番号を、1234-56-7890 の形式で入力します。

間違った例:

部品のシリアル番号を、以下のいずれかの形式で入力します。

1234567890  
1234-56-7890  
1234 56 7890

## ツールヒントと情報ヒント

適切なコントロールかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

タイムアウト

配置

ツールヒント

情報ヒント

スタートメニューの情報ヒント

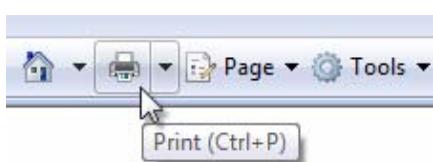
クリック起動のツールヒント

コントロールパネルの情報ヒント

アイコン

ドキュメント

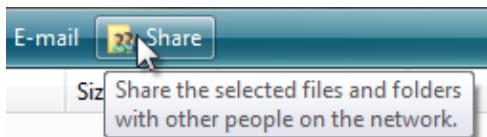
"ツールヒント"とは、ラベルのないツールバーのコントロールやコマンドボタンなど、ラベルのないコントロールをポイントしたときに表示される小さなポップアップウィンドウのことです。



ツールバーのボタンに表示される典型的なツールヒント

ツールヒントはとても便利であり、関連コントロールとして、ツールヒント以上の詳細テキストを示すことができる情報ヒントというコントロールもあります。

"情報ヒント"とは、ポイントされたオブジェクトの簡潔な説明を表示する小さなポップアップウィンドウのことです。説明が表示されるオブジェクトには、ツールバーのコントロール、アイコン、グラフィック、リンク、Windows® エクスプローラーのオブジェクト、スタートメニューのアイテム、タスクバーのボタンなどがあります。情報ヒントは段階的表示の形式で表示されるため、情報ヒントを使用すると、画面上に常に説明テキストを表示しておく必要がなくなります。



典型的な情報ヒント

ここでは、説明の目的上、ツールヒントと情報ヒントをまとめて "ヒント" と呼びます。

ヒントは、知られていないかなじみのないオブジェクトについて、ユーザーインターフェイス(UI)で直接説明が表示されない場合に、ユーザーの理解を助けるものです。ヒントは、ユーザーがオブジェクトをポイントすると自動的に表示され、ユーザーがコントロールをクリックするか、マウスを動かすか、ヒントのタイムアウトになったときに画面から消えます。

開発者向け情報: 情報ヒントのコントロールはありません。情報ヒントは、ツールヒントのコントロールで実装されます。この2つの違いは、実装方法ではなく使用方法にあります。

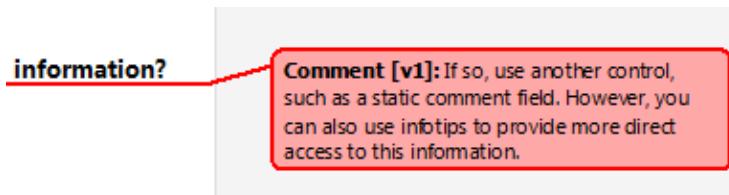
注: バルーン、ツールバー、ヘルプに関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なコントロールかどうかの判断基準

以下の点に基づいて判断します。

- ポイント時に表示する情報かどうか。該当しない場合は、別のコントロールを使用します。ヒントは、ユーザー操作の結果として表示するもので、一方的に表示するものではありません。これとは対照的に、バルーンは、通知と同様に一方的に表示できます。バルーンには、ソースを示す吹き出しが付いています。
- コントロールにテキストラベルがあるかどうか。該当しない場合は、ツールヒントでラベルを提供します。ほとんどのコントロールには、ツールヒントではなくラベルを付ける必要があることに注意してください。ツールバーのコントロールと、グラフィックラベルが付いたコマンドボタンには、ツールヒントを付けます。

- オブジェクトについて補足説明や詳細情報が得られると便利かどうか。該当する場合は、情報ヒントを使用します。ただしヒントのテキストは、主要タスクに不可欠なものではなく補足的なものである必要があります。不可欠なテキストは、ユーザーがわざわざ見つけ出さなくても済むように、UIに直接表示します。
- 補足情報がエラー、警告、状態についての情報かどうか。該当する場合は、バルーン、エラーメッセージ、またはステータスバーなど別のUIを使用します。例外として通知領域アイコンには、情報ヒントで状態情報を表示できます。
- ユーザーがヒントに対して対話操作を行う必要があるかどうか。該当する場合は、バルーンなど別のコントロールを使用します。ヒントはマウスを動かすと消えるため、ユーザーが対話操作を行うことはできません。
- ユーザーが補足情報を印刷するかどうか。該当する場合は、静的なコメントフィールドなど別のコントロールを使用します。ただし、より直接的に情報を表示するために、情報ヒントを使用することもできます。



この例では、*Microsoft Word* で静的なコメントフィールドが使用されており、ユーザーはコメントを印刷できます。

- ユーザーがヒントをわずわらしく感じるようなコンテキストかどうか。該当する場合は、何もしないことも含めて別の解決策を検討します。どうしてもヒントを使用する場合は、ユーザーがヒントの非表示を選択できるようにします。

ヒントを適切に使用すると、ユーザーとのコミュニケーションを向上できます。設計を改良して対処できる場合にヒントで代用することは避けます。グラフィック、ボタンなどのオブジェクトを理解するために、ユーザーがいつもヒントを確認しなければならないような設計は、良くない設計であり修正する必要があります。

## デザイン コンセプト

ヒントは、ユーザーインターフェイスを簡略化する優れた方法です。ヒントが用意されていると、ユーザーは必要なときに最小限の努力で情報を得ることができます。また、ヒントでは画面領域を効率的に使用できるので、画面がすっきりします。しかし、うまく設計されていないヒントは、ユーザーの役に立つどころか邪魔になります。このようなヒントにならないようにするには、以下に示すデザインコンセプトに従います。

### 見つけやすさ

ヒントは、ユーザーが一定時間オブジェクトをポイントすると自動的に表示されます。この少し時間を置いて表示されるしくみは、ヒントを便利にする一方で、気付かれにくくしてしまうこともあります。

ある程度ユーザーが慣れてくると、ユーザーは、ツールバーのボタン、グラフィックボタン、スタートメニューのアイテム、通知領域アイコンなどの特定の標準オブジェクトにヒントが表示されることに気付くようになるため、それほど心配はありません。

標準的ではない場所では、ユーザーがヒントに気付くまでにもっと時間がかかります。ホットスポットやポインタの変更など、オブジェクトにヒントがあることを示す視覚的なヒントはありません。さらに悪いことに、UIを使い始めたばかりのユーザーなどは、マウスポインターをあちこち移動させることができます。オブジェクトにヒントがあることをユーザーが知るには、過去の経験か試行錯誤のどちらかになります。

一貫した方法で使用することにより、ヒントは見つけやすく、予測しやすいものになります。いくつかのオブジェクトにヒントを用意した場合は、ユーザーが補足情報を必要とする類似のオブジェクトにもすべてヒントを用意します。ヒントは有益かつ自明でない情報である必要もあるため、これは難しいこともあります。

見つけやすく一貫して役に立つヒントを提供することが難しい場合は、ヒントがなくても理解できるコントロールラベルを使用する、インプレースの補足テキストを付けるなど、別の設計を考慮してください。

### 適切な情報

ヒントとして適切な情報には次の特徴があります。

- ・ 簡潔。ヒントに使用されるポップアップ ウィンドウは、短い文または語句のテキストを書式付きで表示するのに適しています。書式が設定されていない長いテキストは、読みづらく、煩雑になります。
- ・ 有益。ヒントのテキストには、参考になる情報が書かれている必要があります。明らかにわかっていることや、画面上に既にあるものの繰り返しであってはなりません。
- ・ 補足的。ヒントのテキストは、常にユーザーが目にするとは限らないため、必ずしも読む必要のない補足情報にとどめます。重要な情報は、ヒントがなくても理解できるコントロール ラベルやインプレースの補足テキストを使用して伝えるようにします。
- ・ 不変。ユーザーは一度確認したヒントが次回変わっているとは考えないため、状態情報などの動的なコンテンツの変化には気付かない可能性があります。ただし通知領域アイコンに表示されるヒントは例外です。これらのアイコンは状態を示すことが主な目的であるため、ユーザーはヒント情報の変化に気付きやすいといえます。

## 適切なタイムアウト

適切なタイミングでヒントが自動表示され消去されることは、ユーザーが各自の UI 環境をコントロールするうえで欠かせない条件です。ヒントには、次の 3 つのタイムアウト値があります。

- ・ 初回表示。ポイントした状態でヒントが表示されるまでの時間です。既定値は 0.5 秒です。
- ・ 再表示。ターゲット間を移動中に、1 つのターゲットをポイントした状態でヒントが表示されるまでの時間です。既定値は 0.1 秒です。
- ・ 消去。ヒントが自動的に消去されるまでの時間です。既定値は 5 秒です。

初回表示と再表示の値が小さすぎると、意図しないときにヒントが表示されることが多くなり、ユーザーの邪魔になります。逆に値が大きすぎると、ヒントの表示が遅いと感じられたり、ヒントが発見されない場合があります。消去までの既定の時間は、ツールヒントに使用されているような短いヒントテキストに適した時間になっています。情報ヒントのテキストはもう少し長いため、表示時間も長くする必要があります。

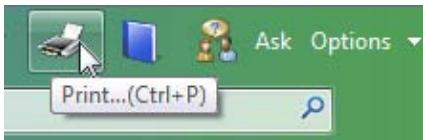
## 適切な配置

ヒントは、ポイントされるオブジェクトの近くに配置します。通常はポインターの後方か、可能であればポインターの前方に配置し、ユーザーのオブジェクト操作の邪魔になる位置には配置しないようにします。ヒントがオブジェクトを隠してしまう場合は、ヒントをポインターから離し、オブジェクトに隣接する位置に配置します。オブジェクトとヒントとの関係性が明らかである限り、この配置でも問題はありません。ユーザーがプログラムのヒントを消すためだけにポインターを移動することがないようにします。

## アクセシビリティ

ヒントは、アクセシビリティに関しては特殊な効果を持ちます。ヒントは、通常、オブジェクトをポイントすることにより表示されますが、キーボードでアクセスできるコントロールに付いているヒントであれば、[スクリーンリーダー](#)にも対応します。簡潔、有益、不变かつ補足的な情報を示す適切なヒントは、アクセシビリティ全般の向上につながります。事実、代替テキストとしてのヒントパターンは、グラフィックのアクセシビリティに推奨される方法です。ただし、ヒントが不適切に使用されると、重要な情報や動的な情報を取得するのが難しくなり、アクセシビリティの障害になります。

キーボードでアクセスできないコントロールにヒントが必要な場合は、コントロールへのアクセス方法を複数提供します。



この例では、ツールバーのボタンを使用しても、キーボードでアクセスできないツールバーのボタンの代わりに [印刷] コマンドのキーボードショートカットを使用しても、印刷を実行できます。

### 最も重要な点

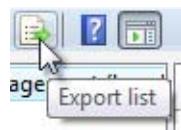
適切な位置に適切なタイミングで、簡潔、有益、不变かつ補足的な情報を表示する、見つけやすいヒントを設計します。

## 使用パターン

ヒントにはいくつかの使用パターンがあります。

**ツールヒント** このパターンのヒントはラベルとして機能するため、対象コントロールに応じて、ラベルのガイドラインに従ったテキストを使用します。

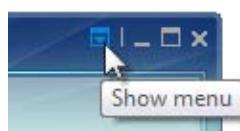
コントロールまたはグリフに対し  
てラベルを表示  
します。



この例では、ツールヒントがコマンドのラベルになっています。

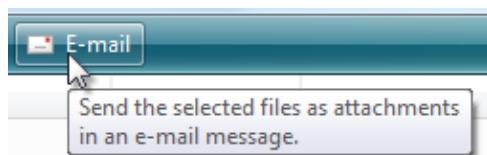


この例では、ツールヒントがグラフィック ボタンのラベルになっています。



この例では、ツールヒントがグリフのラベルになっています。

**情報ヒント** 情報ヒントは、[ツールバー](#)のコントロール、[アイコン](#)（アイコン オーバーレイも含む）、[リンク](#)、[タブ](#)、[段階的表示コントロール](#)、カスタム コントロールなどのオブジェクトやコントロールについて説明するときには使用します。



この例では、情報ヒントでコントロールとオブジェクトについての補足情報が提供されています。

**代替テキストとしての情報ヒント** このパターンは、視力に障害を持ち、スクリーンリーダーを使用するようなユーザーを主な対象にしています。

ト

アクセシビリティのため、グラフィックに説明を提供します。



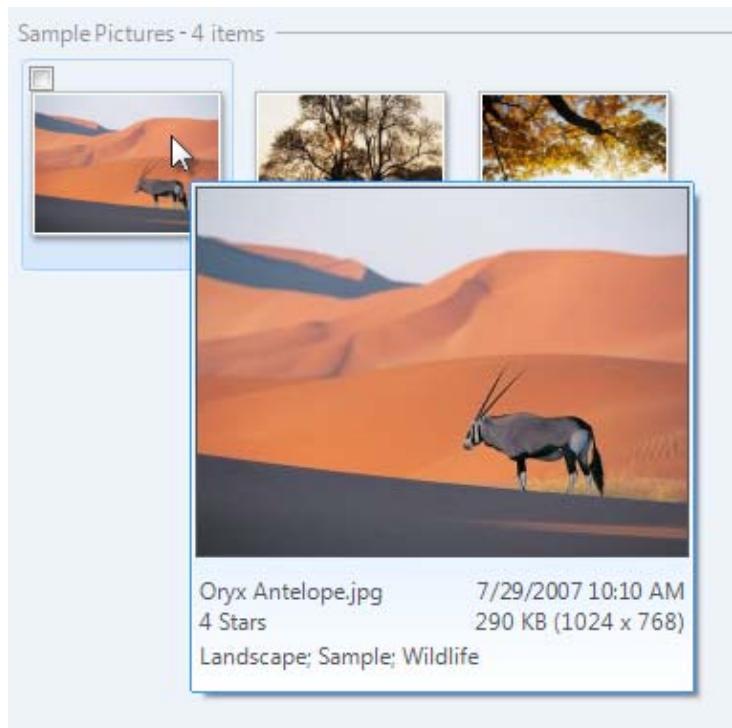
この例では、情報ヒントでスタートメニューのグラフィックが説明されています。

サムネイル アイテムの縮小  
アイコンを表示しま  
す。

サムネイルを使用すると、ウィンドウやドキュメントが、簡単に認識できるグラフィックで表示されます。

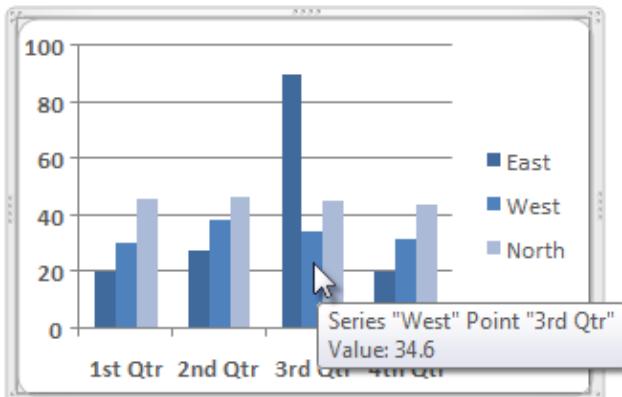
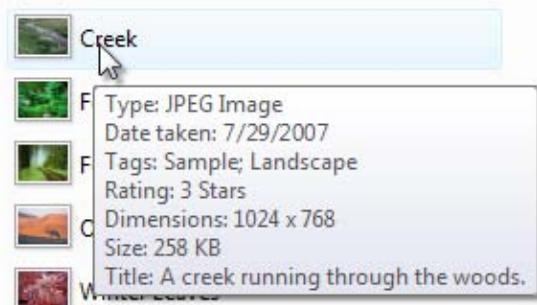


この例では、Windows® タスク バーのアイテムにサムネイルヒントが表示されています。



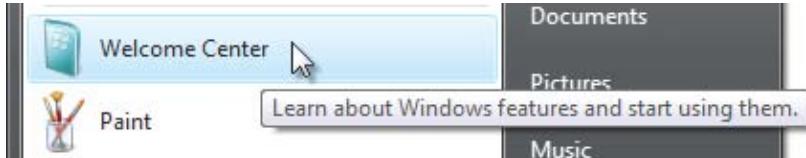
この例では、Windows フォト ギャラリーのアイテムにサムネイルヒントが表示されています。

**詳細な情報ヒント** 情報ヒントは、オブジェクトに関する詳細情報を表示したり、データを提供するのに効果的な方法です。  
**オブジェクトについて** 関する詳細情報を表示します。



この例では、情報ヒントでオブジェクトに関する詳細情報とデータが提供されています。

スタートメニューの情報ヒント  
スタートメニューのアイテムを説明します。  
この例では、スタートメニューには、プログラム名のほか、[ドキュメント]、[ピクチャ]、[コントロールパネル]などのWindowsの重要なアクセス先が表示されます。このパターンのヒントは、スタートメニューのアイテムを説明します。ヒントの内容は、通常、プログラムやアクセス先の短い説明、ユーザーが実行できる主なタスクなどです。この説明は、スタートメニューの検索ボックス用にもインデックス登録され、ユーザーが必要なプログラムを検索するときの助けとなります。



この例では、情報ヒントでスタートメニューのプログラムが説明されています。

コントロールパネルの情報ヒント  
このパターンのヒントでは、コントロールパネルのカテゴリやアイテムをユーザーが間違いなく選択できるようにするための補足情報が提供されます。

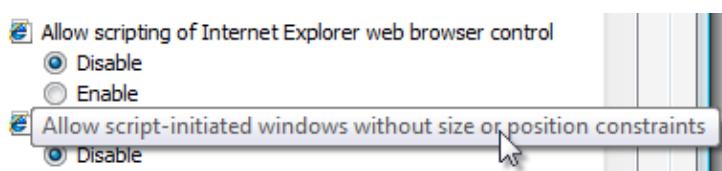
コントロールパネルのカテゴリまたはタスクを説明します。



この例では、情報ヒントで、コントロールパネルの「ユーザー アカウント」カテゴリが説明されています。

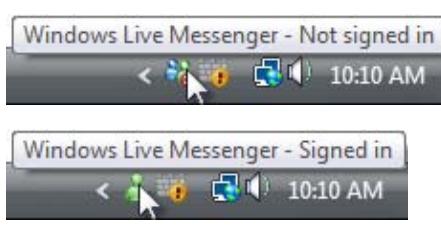
完全な名前を表示する情報ヒント  
このパターンのヒントを使用すると、より小さい領域にアイテムを表示でき、水平スクロールも不要です。長さが不明な動的コンテンツには特に、このヒントが重要になります。他のパターンとは違い、リストやツリーでは、このヒントはソースオブジェクトに直接覆いかぶさるように表示されます。

アイテムの名前が省略されたりすべて表示されない場合に、完全な名前を表示します。



この例では、ポイントしたアイテムの完全な名前が情報ヒントで表示されています。

状態の情報ヒント  
通知領域のアイコンに状態情報を表示します。  
ユーザーは一度確認したヒントが次回変わっているとは考えないため、ヒントは、通常、不变である必要があります。ただし通知領域のアイコンは例外です。これらのアイコンは状態を示すことが目的であり、また、状態テキストに使用できる画面領域はありません。



この例では、情報ヒントで通知領域のアイコンに状態情報が表示されています。

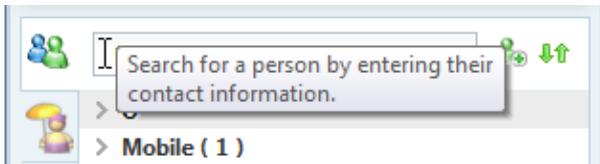
## ガイドライン タイムアウト

- 初回表示と再表示のタイムアウトには、既定値を使用します。次の場合は例外です。
  - オブジェクトの横に、大きすぎない関連サムネイルを、間を置くことなくすぐに表示できる場合。ただし、大きすぎるサムネイル(小さいグラフィックオブジェクトに対する大きいサムネイルヒントなど)や、関連オブジェクトを覆うサムネイルの場合は、初回表示のタイムアウトには既定値を使用します。
- ツールヒントの場合、消去タイムアウトには、5秒の既定値を使用します。
- 情報ヒントの場合、消去タイムアウトは無効にします。開発者向け情報: 技術的には消去タイムアウトを無効にすることはできないので、最大値に設定します。
- アクセシビリティの観点から、タイムアウト値を最大値以外の値に設定する必要がある場合は、固定の時間にする代わりに SPI\_GETMOUSEHOVERTIME および SPI\_GETMESSAGEDURATION システムパラメーターの倍数を使用します。この方法を使用すると、ユーザーの速度に合わせてタイムアウトを調節できます。

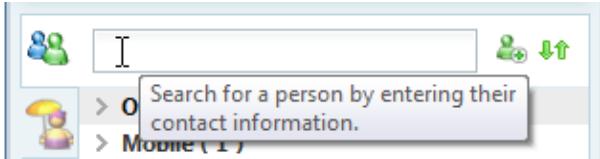
## 配置

- ユーザーが確認または操作しようとするオブジェクトがヒントで隠れないようにします。ポインターとヒントが離れることになっても、ヒントは常にオブジェクトの脇に配置します。オブジェクトとヒントとの関係性が明らかである限り、多少離れても問題はありません。
  - 例外: リストやツリーで使用される、完全な名前を表示するツールヒント。

間違った例:



正しい例:



正しい例では、情報ヒントはキャレットから離れていて、検索ボックスを隠さないように配置されています。

間違った例:

**Jonathan, 7/29/2007 10:10:35 AM inserted:**  
Avoid covering the object the user is about to view or interact with. Always place the tip on the side of the object, even if that requires separation between the pointer and the tip. Some separation isn't a problem as long as the relationship between the object and its tip is clear. Exception: Full name tooltips used in lists and trees.

- Avoid covering the object on the side of the object, even if that requires separation between the pointer and the tip. Some separation isn't a problem as long as the relationship between the object and its tip is clear.**
- Exception: Full name tooltips used in lists and trees.**

正しい例:

**Jonathan, 7/29/2007 10:10:35 AM inserted:**  
Avoid covering the object the user is about to view or interact with. Always place the tip on the side of the object, even if that requires separation between the pointer and the tip. Some separation isn't a problem as long as the relationship between the object and its tip is clear. Exception: Full name tooltips used in lists and trees.

- Avoid covering the object the user is about to view or interact with. Always place the tip on the side of the object, even if that requires separation between the pointer and the tip. Some separation isn't a problem as long as the relationship between the object and its tip is clear.**
- Exception: Full name tooltips used in lists and trees.**

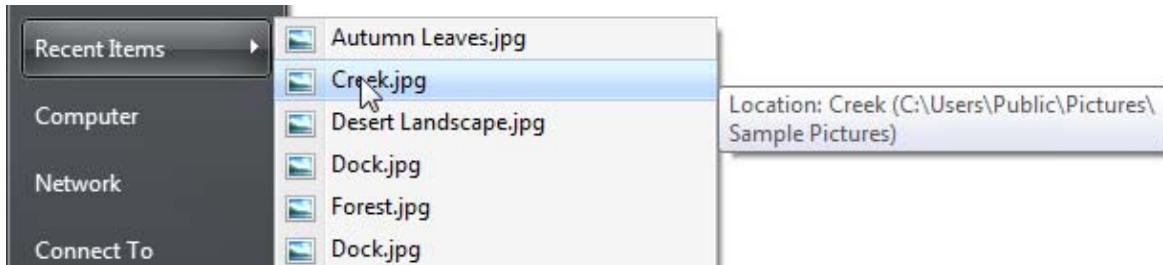
正しい例では、下にあるテキストの方がずっと有用であるため、情報ヒントはテキストの邪魔にならないように配置されています。

- アイテムの集まりに対しては、次に来るオブジェクトをユーザーが確認または操作する可能性がある場合、オブジェクトが隠れないようにします。水平にアイテムが並んでいる場合は、右にヒントを配置しないようにし、垂直にアイテムが並んでいる場合は、下にヒントを配置しないようにします。

間違った例:



正しい例:



間違った例では、ユーザーが次に操作する可能性の最も高いオブジェクトが、ヒントによって隠れています。

- ユーザーの妨げとなる可能性のある大きなヒントなどは、その情報が大多数のユーザーにとって有益であることを確認します。大多数のユーザーに有益とは言えない、邪魔になる可能性のあるヒントは、オプションにするか削除します。このようなヒントが表示されると、大多数のユーザーは、ヒントを消すためにターゲットオブジェクトからポインターを離さなくてはならなくなります。

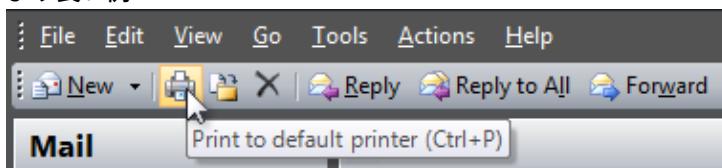
#### ツールヒント

- ラベルのないコントロールにラベルを付けるには、ツールヒントを使用します。通常ツールヒントを備えているコントロールとしては、ツールバーのボタン、グラフィックボタン、段階的表示コントロールがあります。テキストボックス、コンボボックスなどプロンプトの付いたコントロールは、ラベルがあるのが普通です。他のすべてのコントロールにも、明示的なラベルがあると考えられます。
- 語句を使用し、末尾に句読点は付けません。
- センテンススタイルの大文字化を使用します。
  - 例外: このガイドラインは、Windows Vista®用の新しいものです。古いアプリケーションでは、大文字/小文字の形式が混在しないよう、タイトルスタイルの大文字化を採用してもかまいません。
- 追加情報が必要なコマンドのラベルには、省略記号を追加します。
- 通常のラベルと同様に、ツールヒントは短くします(通常、5語あるいは15文字以内)。ただし、あいまいになるよりは具体的になる方を優先します。

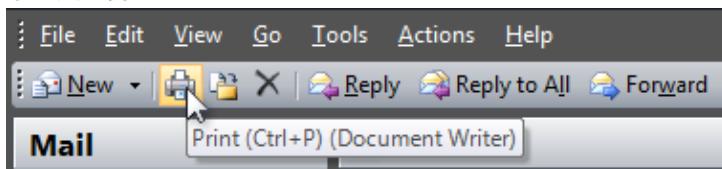
許容される例:



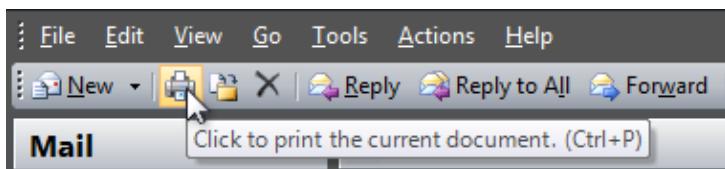
より良い例:



最も良い例:



間違った例:



以上の例では、最も良い例が簡潔かつ具体的です。間違った例は、不必要に説明が長くなっています。

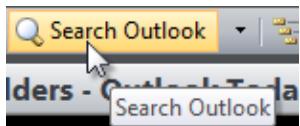
- ツールバーのボタンにラベルが付けられている場合でも、有益であれば、ツールヒントで詳細を提供してもかまいません。既にラベルにある情報を繰り返したり言い直すことはしません。

正しい例:



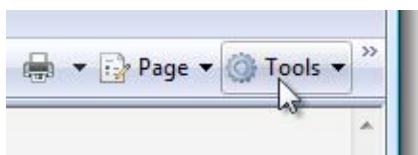
この例では、ツールヒントで検索機能が説明されています。

間違った例:



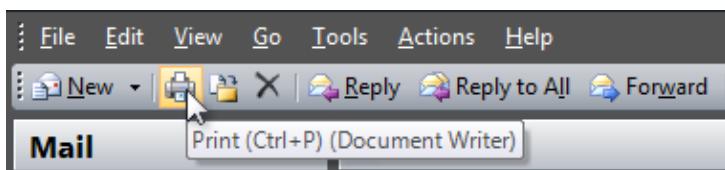
この例では、ツールヒントは、既にラベルにある情報を繰り返すだけになっています。

- 一貫性を維持する目的でのみ、ラベル付きコントロールにツールヒントを付与する必要はありません。



この例では、ラベルのないツールバーのボタンにはツールヒントが表示され、ラベルのあるツールバーのボタンには表示されません。

- 適切である場合には常に、キーボードショートカットと既定値を提供して、ツールヒントがより便利になるようにします。この追加情報はあっても役立つことがあります。キーボードショートカットと既定値を提供することで、ラベル付きコントロールのツールヒントは、ただのラベルの繰り返しではない有益な情報になります。この追加テキストは、ツールヒントが簡潔かどうかを判断する際は考慮に入れないようにします。



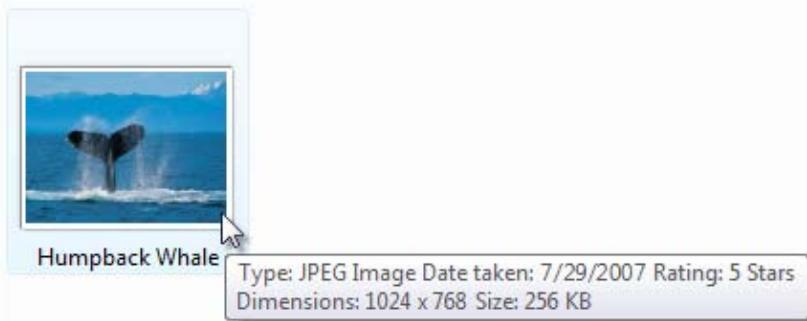
この例では、Word のツールバーのツールヒントで、既定値とキーボードショートカットが提供されています。

## 情報ヒント

- 標準的でない場所にある情報ヒントは、容易に発見できるように有用性よりも一貫性を重視します。ユーザーが補足情報を必要とする可能性が高いオブジェクトには、そのいくつかがわかりきった情報ヒントになる可能性があるとしても、すべてヒントを用意します。こうすることで、ユーザーは表示されない情報ヒントを待つことがなくなります。
  - 例外: 有益な情報ヒントが表示されるオブジェクトが数個だけになる場合は、情報ヒントの使用をやめて、代わりに、ヒントがなくても理解できるコントロール ラベルやインプレースの補足テキストを使用します。
- 文を使用し、末尾に句点付けます。
  - 例外: 通知領域アイコンの情報ヒントには、末尾に句読点を付けません。
- センテンス スタイルの大文字化を使用します。
- 未来形ではなく現在形を使用します。
- 同じ文法構造を使用します。同じ文法構造にするには、同じ役割を果たす語句を同じ形式で使用する必要があります。

- 例外: 完全な名前を表示する情報ヒント パターンの場合、情報ヒントのテキストでは、対象コントロールと正確に同じ言い回し、大文字と小文字、および句読点を使用します。
- 情報ヒントは大きくならないようにします。大きな情報ヒントは読みづらく、対象オブジェクトの邪魔にならずに表示することが困難です。
- 内容が読みやすいように、情報ヒントに書式を設定します。書式が設定されていない大きなテキストブロックは、読みづらくなります。

間違った例:



正しい例:



正しい例では、テキストに書式が設定され、読みやすくなっています。

- 情報ヒント内で初めて略語を使用する場合は、正式名称の後にかっこ付きで略語を表記します。たとえば、「動的ホスト構成プロトコル(DHCP)」のように表記します。

#### スタートメニューの情報ヒント

- スタートメニューの情報ヒントでは、アイテムを簡潔に説明し、ユーザーがアイテムに対して実行できる主なタスクを示します。
- 有益な情報を提供します。ユーザーができることに焦点を当て、アイテムの名前を繰り返したり、説明の中でアイテムの名前を使用することは避けます。
- 具体的にします。一般的な動詞や、"その他のタスク" のようなどこにでも当てはまる語句は使用しません。重要な情報は、具体的に一覧にします。一覧にしない場合は、情報ヒントの一覧が完全な一覧でないことをユーザーが理解している必要があります。
- 簡潔にします。25ワード(全角で約75文字)以内で表記します。これより長い情報ヒントは、ユーザーに読む気を失わせます。
- 現在時制の動詞の命令形で始めます(英語の場合)。"create(作成)"、"edit(編集)"、"show(表示)"、"send(送信)"などを使用します。"管理する"、"開く"のような一般的な動詞はスタートメニューのほとんどのアイテムに当てはまるため、より具体的な動詞を優先し、要点だけを伝えます。

間違った例:



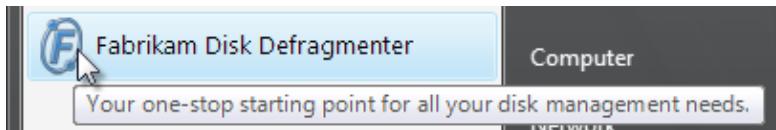
より良い例:



間違った例では、情報ヒントに一般的な動詞が使用されています。より良い例では、特定の動詞で要点が示されていますが、“その他の”という不要な語も使用されています。

- 宣伝文句のような表現は使用しません。

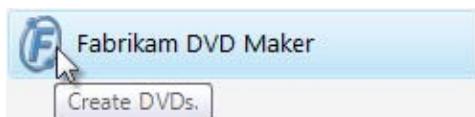
間違った例:



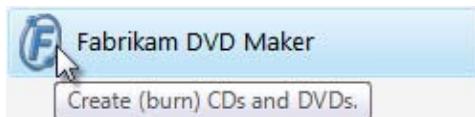
この例では、情報ヒントが宣伝文句のようになっています。

- このパターンの情報ヒントは、スタートメニューの検索ボックス用にもインデックス登録されるので、ユーザーが検索する可能性が高い用語を使用して、プログラムの重要なタスクを説明します。また、キーワードや一般的な類義語を使用するようにします。

間違った例:



正しい例:



正しい例では、情報ヒントに一般的な類義語が入っています。

- センテンススタイルの大文字化を使用します。
- 開発者向け情報: スタートメニューの情報ヒントのテキストは、アイテムの Comment フィールドから表示されます。

## クリック起動のツールヒント

- 次の形式でツールヒントを使用します: (完全なプログラム名) の起動
- 末尾に句読点は付けません。
- プログラムや機能について説明する追加テキストは使用しません。ユーザーは、クリック起動バーに表示されているプログラムを選択した時点で、既に目的を認識しています。

## コントロールパネルの情報ヒント

- コントロールパネルの情報ヒントでは、コントロールパネルのタスクや構成するハードウェアとソフトウェアを簡潔に説明します。
- コントロールパネルの名前とアイコンには、情報ヒントが必要です。個々のタスクにツールヒントは表示しません。
- 有益な情報を提供します。ユーザーができることに焦点を当て、コントロールパネルのアイテムの名前を繰り返したり、説明の中でアイテムの名前を使用することは避けます。
- 具体的にします。一般的な動詞や、“その他のハードウェア”的な用語などにでも当てはまる語句は使用しません。重要な情報は、具体的に一覧にします。一覧にしない場合は、情報ヒントの一覧が完全な一覧でないことをユーザーが理解している必要があります。

間違った例:



正しい例:



正しい例では、構成されるハードウェアの種類が具体的に一覧表示されています。

- 簡潔にします。25ワード(全角で75文字程度)以内で表記します。これより長い情報ヒントは、ユーザーに読む気を失わせます。
- 現在時制の動詞の命令形で始めます(英語の場合)。

正しい例:

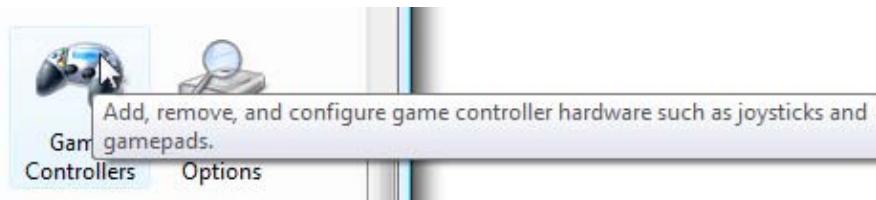
インターネットの表示と接続の設定を行います。  
視覚、聴覚、モビリティの設定を調整します。

- 要点だけを伝えます。"...の外観と機能の設定を表示または構成するために使用します"、"...のオプションを提供します"のような、コントロールパネルのどのアイテムにも当てはまる言い回しは使用しません。
- 宣伝文句のような表現は使用しません。

間違った例:

ディスク構成のどのようなニーズにも、1か所で対応できます。

- このパターンの情報ヒントは、コントロールパネルの検索ボックス用にもインデックス登録されるので、ユーザーが検索する可能性が高い用語を使用して、アイテムを説明します。よく使用されるタスクやオブジェクトに共通の類義語を使用するようにします。



この例では、ユーザーが検索する可能性が高い用語でアイテムが説明されています。

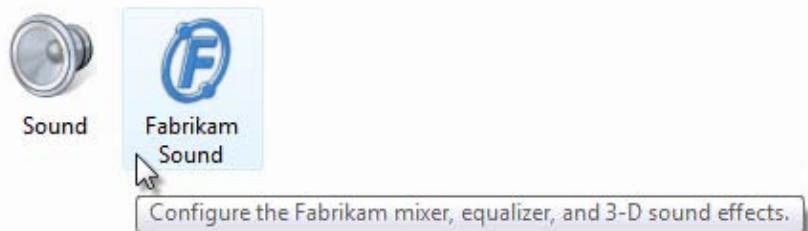
- コントロールパネルのアイテムが他と混同される可能性がある場合は、情報ヒントでその違いを説明します。

間違った例:



この例では、どちらのコントロールパネルのアイテムもサウンドを設定するのですが、情報ヒントでその違いが説明されていません。

正しい例:



この例では、2つのアイテムの違いがヒントではっきりと説明されています。

## アイコン

Windows の以前のバージョンとは違い、Windows Vista では、ヒントにもアイコンを使用できます。

- ツールヒントにはアイコンを使用しません。
- 情報ヒントには、アイコンが認識や理解の助けになったり、アイコンによってコンテキストが提供される場合のみ、アイコンを使用します。ほとんどの情報ヒントにはアイコンを使用しません。



この例では、情報ヒントにアイコンが使用され、アイコンとその意味を関連付けるのに役立っています。

- アイコンには [Aero style](#) を使用し、目立ちすぎないようにします。

一般的なアイコンのガイドラインと例については、「[アイコン](#)」を参照してください。

## ドキュメント

ヒントに言及するときは、以下のことに留意します。

- プログラミングおよびその他の技術文書では、ヒントの種類 ("ツールヒント" または "情報ヒント") を示します。それ以外のドキュメントでは、単に "ヒント" と呼びます。
- "ツールヒント" (スペースあり)、"ツールのヒント" などの表記は間違っています。
- ユーザー操作を説明する場合は、"ポイントする" を使用します。

## ツリー ビュー

適切なコントロールかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

提示方法

対話操作

ツリー構成

チェックボックスのツリー ビュー

推奨されるサイズと間隔

ラベル

ドキュメント

"ツリー ビュー" では、ユーザーは階層的に整理されたオブジェクトのコレクションを表示し、單一または複数のオブジェクトを選択するという対話操作を行うことができます。

ツリーにおいて、データを含むオブジェクトは "リーフノード" と呼ばれ、データ以外のオブジェクトを含むオブジェクトは "コンテナーノード" と呼ばれます。最上位の單一のコンテナーノードは "ルートノード" と呼ばれます。ユーザーは、"プラスとマイナスの展開ボタン" をクリックして、コンテナーノードを展開したり折りたたむことができます。



典型的なツリー ビュー

注: レイアウトおよびメニューに関するガイドラインは、それぞれ別の項目として記載しています。

## 適切なコントロールかどうかの判断基準

階層データがあるからといって、必ずしもツリー ビューを使用する必要はありません。リスト ビューの方が簡単で効果的な場合も数多くあります。リスト ビューの特長を次に挙げます。

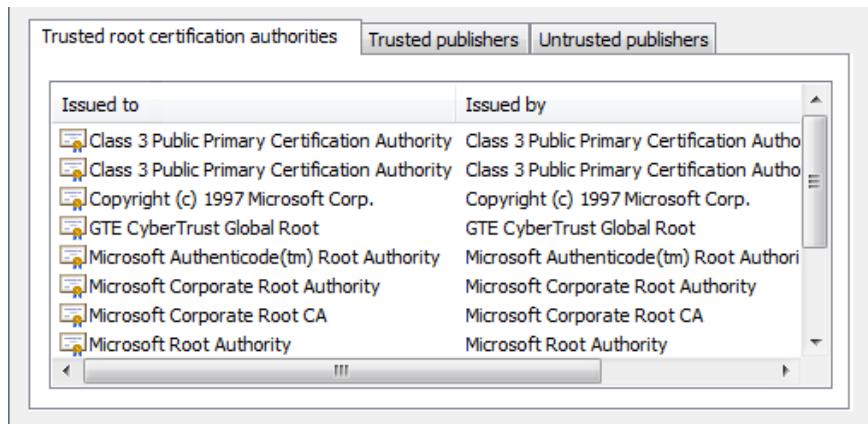
- 数種類のビューがサポートされます。
- 詳細表示では、任意の列を基準にしてデータを並べ替えることができます。
- データを 2 レベルの階層にグループ化して整理できます。

リスト ビューを使用する場合は、以下に示す方法で階層情報を消化できます。

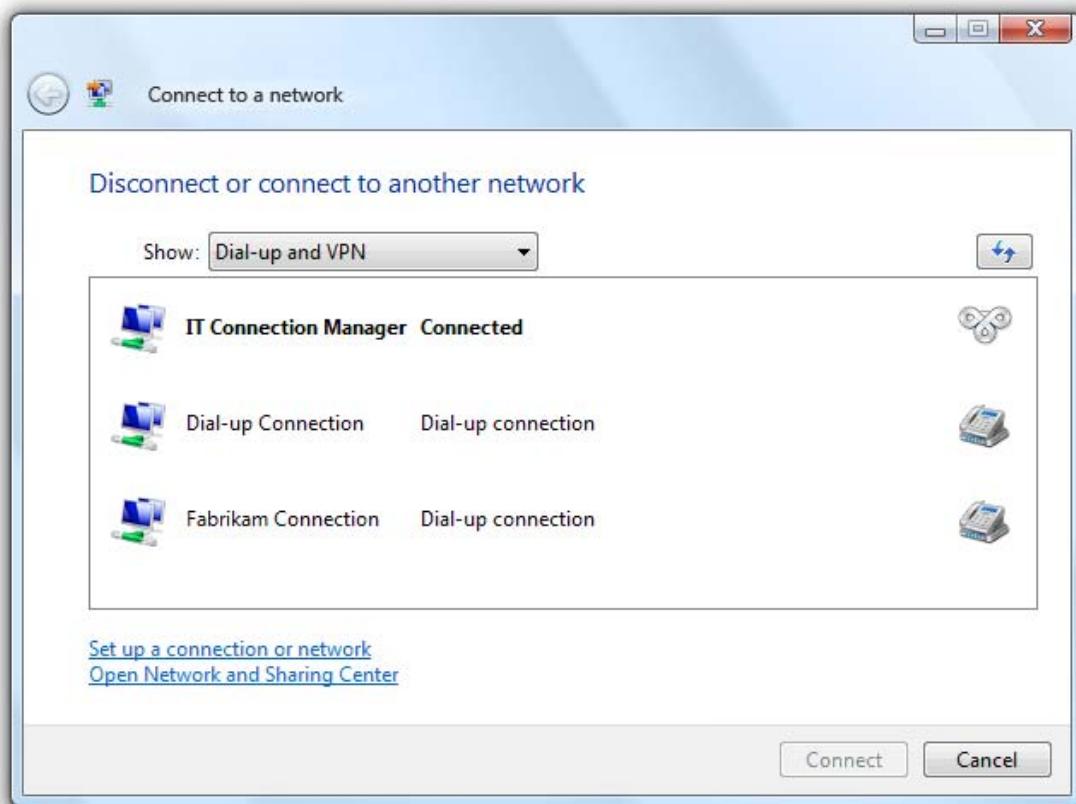
- ルートノードがある場合は、不要なことが多いので削除します。
- 最上位のコンテナーは、リスト ビュー グループ、タブ、ドロップダウンリスト、または展開可能な見出しで置き換えます。

Name	Category	Workgroup
Media Devices (4)		
Fabrikam	Media Devices	
Microsoft	Media Devices	
Jonathan	Media Devices	
Guest	Media Devices	
Computer (526)		
New guest	Computer	Workgroup
Jonathan	Computer	Workgroup
Guest	Computer	Workgroup
Fabrikam	Computer	Workgroup
Microsoft	Computer	Workgroup

この例では、リスト ビュー グループが最上位のコンテナーに使用されています。



この例では、タブが最上位のコンテナーに使用されています。



この例では、ドロップダウンリストが最上位のコンテナーに使用されています。

- 関連付けられたコントロールで、選択したコンテナーの内容が表示される場合は、そのコントロールで階層の下位レベルを表示できます。

Tabs  
 Text Boxes  
 Tooltips and Infotips  
 Tree Views  
 Notifications  
 Search Boxes  
 Status Bars  
 Commands  
 Text  
 Interaction

## Tree Views

[Is this the right control?](#)  
[Design concepts](#)  
[Usage patterns](#)  
[Guidelines](#)  
[Recommended sizing and spacing](#)  
[Labels](#)

With a tree view, users can view and interact with a hierarchically arranged collection of objects, using either single selection or multiple selection.

この例では、下位レベルのコンテナーがドキュメント ウィンドウで表示されています。

ルート ノードを除く 3 レベル以上の階層を表示する場合は、ツリー ビューを使用する必要があります。

ツリー ビューの使用が適切かどうかは、以下の点に基づいて判断します。

- 階層化されたデータかどうか。該当しない場合は、別のコントロールを使用します。

- 階層のレベルが、ルートを除いて 3 レベル以上あるどうか。該当しない場合は、リスト ビュー グループ、タブ、ドロップダウンリスト、展開可能な見出しなどを使用するようにします。
- アイテムに補助データがあるかどうか。該当する場合は、データを活かすため、詳細表示モードのリスト ビューを使用するようにします。
- 下位レベルのデータが、独立したサブタスクに関係しているかどうか。該当する場合は、関連付けられたコントロール内、またはコマンド ボタンやリンクを使用して表示される別のウィンドウ内で表示するようにします。
- 対象ユーザーが詳しい知識のあるユーザーかどうか。詳しい知識のあるユーザーであればツリーを使い慣れていますが、アプリケーションが初級ユーザー向けの場合、ツリービューの使用は避けます。
- それぞれのアイテムが、大部分のユーザーにとって理解しやすく自然な、単一の階層的カテゴリに分けられるかどうか。該当する場合は、データはツリービューに適しています。複数のビューや並べ替えが必要になる場合は、代わりにリスト ビューを使用します。
- ユーザーが下位レベルのデータ表示を必要とする状況が、(常にではなく) たまにあるかどうか。該当する場合は、データはツリービューに適しています。

注: ツリービューのように見えるコントロールが、リスト ビューを使用して実装される場合もあります。そのような場合は、実装ではなく使用法に基づいてガイドラインを採用してください。

## デザイン コンセプト

ツリーの目的はデータを整理して見つけやすくすることですが、データをツリー内で見つけやすいものにするには工夫が必要です。ツリービューとその構成を考えるときには、以下の原則に留意します。

### 予測のしやすさと見つけやすさ

ツリービューは、オブジェクト間の関連性を基本とします。ツリーが最も効果を発揮するのは、明確で一般的な関連性をオブジェクトが相互排他的に形成している場合です。この場合、すべてのオブジェクトは、簡単に特定できる単一のコンテナーにマッピングされます。

大きな問題として、1 つのオブジェクトを数種類のノードで表示可能な場合が挙げられます。たとえば、音楽の再生に使用でき、大容量のハードディスクを備え、USB ポートに接続するハードウェア デバイスを、ユーザーはどこで探すでしょうか。おそらく、"マルチメディア"、"ストレージ"、"USB"、場合によっては "ハードウェア リソース" など、数種類のコンテナーノードのどれを探してもおかしくはありません。1 つの解決方法は、状況に関係なく、単一の最も適切なコンテナーの下に各オブジェクトを配置することです。もう 1 つの解決方法は、該当するすべてのコンテナーの下に各オブジェクトを配置することです。前者は簡潔ですっきりとした階層になり、後者はオブジェクトを見つけやすい階層になります。どちらにも利点と潜在的な問題があります。

ユーザーは、ツリーのレイアウトをよく理解していない場合でも、しばらくツリーを操作すると関連性について一定のメンタルモデルを形成するようになります。このメンタルモデルが正しくない場合は混乱を招きます。たとえば、音楽プレーヤーを、"マルチメディア"、"ストレージ"、および "USB" の各コンテナーの配下に置いて、見つけやすくしたとします。ユーザーがこのデバイスを最初に "マルチメディア" 内で見つけた場合、そのユーザーは音楽プレーヤーのようなデバイスはすべて "マルチメディア" コンテナー内に表示されると考え、デジタル カメラなどの同様のデバイスも "マルチメディア" コンテナーにあると想定して表示されなかった場合、混乱することになります。

ツリーの設計にあたっては、見つけやすさと予測のしやすさのバランスがとれた、混乱の少ないユーザー モデルを作成することが課題となります。

### 1 レベルのアイテム数とレベルの深さ

ユーザビリティ調査によると、ユーザーは、レベルの深いツリーよりも 1 レベルのアイテム数の多いツリーでオブジェクトを簡単に見つけることができるようになっています。このため、ツリーを設計するときには、レベルを深くするよりも 1 レベルのアイテム数を多くします。理想的には、ツリーに含めるレベルはルート ノードを除いて 4 つ以内とし、よくアクセスされるオブジェクトは最初の 2 レベル以内に表示します。

### 他の原則

- ユーザーは、必要なオブジェクトが見つかったら探すのをやめ、オブジェクトが見つかる可能性のある他の場所を探すことはありません。このようなとき、ユーザーは、最初に見つけたパス以外にオブジェクトはないと仮定することができます。
- 大きく複雑なツリーでは、オブジェクトを探すのがたいへんです。手動でオブジェクトを探し回るのは疲れるため、ある程度探ししたと考えた時点でユーザーはオブジェクトを探すのをやめます。このため、大きく複雑なツリーでは、文字列検索、索引、フィルタリングなどの他のアクセス方法も提供する必要があります。
- プログラムによっては、ユーザーが独自のツリーを作成できることもあります。自分で作成したツリーはメンタル モデルに一致することもある一方、無計画であったり管理が不十分なことがあります。たとえば、ファイルシステム、電子メール プログラム、およびお気に入り一覧には、通常、同じような種類の情報が格納されますが、ユーザーはめったに同じ方法で整理しようとはしません。

## 最も重要な点

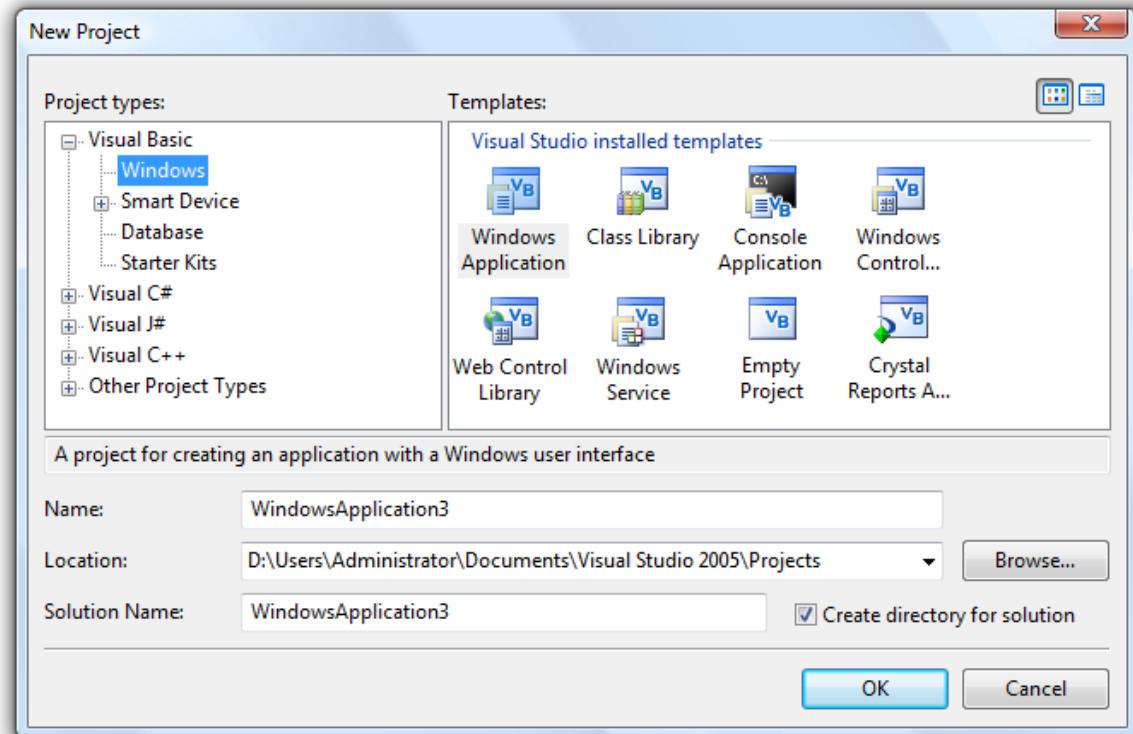
最も重要な点は、ツリービューを使用するメリットとデメリットを慎重に検討することです。階層データがあるからといって、必ずしもツリービューを使用する必要はありません。

## 使用パターン

ツリービューにはいくつかの使用パターンがあります。

**コンテナーノードだけで構成されるツリー**  
通常このパターンのツリービューには、選択したコンテナーの内容を表示するコントロールが関連付けられます。ユーザーは、一度に1つのコンテナーのみに対話操作を行うことができます。

ユーザーは一度に1つのコンテナーを表示し、対話操作を行うことができます。



この例では、ツリービューはコンテナーノードだけで構成され、選択したノードの内容が、関連付けられたリストビュー コントロール内に表示されています。

**コンテナーノードとリーフノードで構成されるツリー**  
通常このパターンのツリービューには、選択したコンテナーまたはリーフの内容を表示するコントロールが関連付けられます。リーフへの対話操作を可能にすると、多くの場合、複数選択のサポートも必要になります。

ユーザーはコンテナーとリーフを表示し、対話操作を行うことができます。

Check Boxes

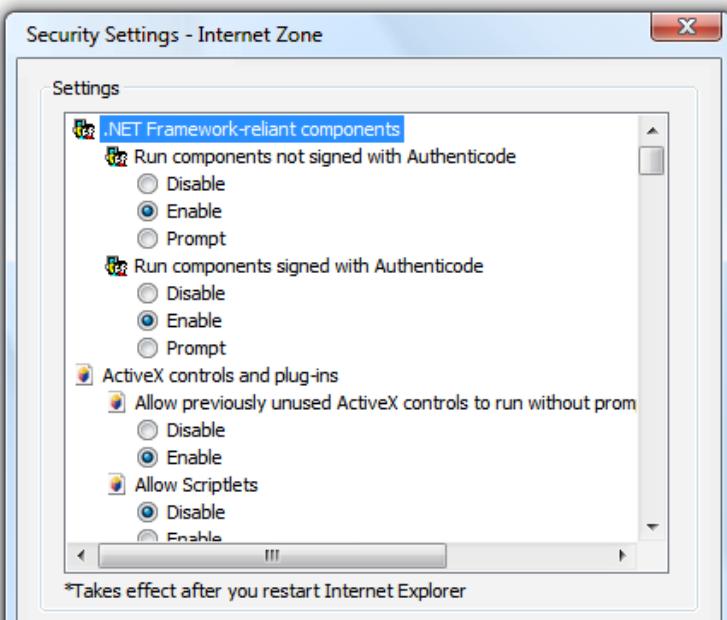
[Is this the right control?](#)  
[Usage patterns](#)  
[Guidelines](#)  
[Recommended sizing and spacing](#)  
[Labels](#)

With a **checkbox**, **users make a decision**  
Consequently, **checkboxes should be us**

Clock  
 Volume  
 Network

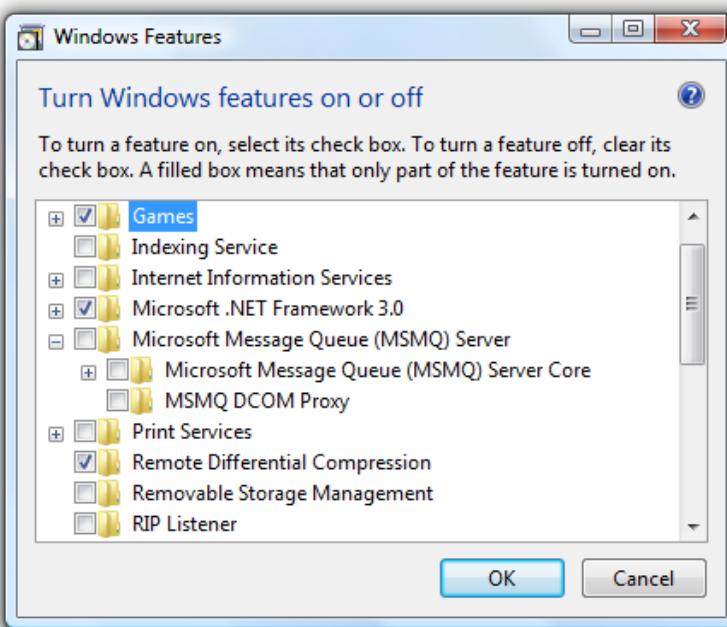
この例では、ツリービューはコンテナーノードとリーフノードの両方で構成されています。複数選択がサポートされているため、開かれたアイテムの内容が、関連付けられたコントロール内でタブを使用して表示されています。

ツリービューを構造的なリストで構成することもできます。この場合、コンテナーは見出しとなり、リーフはオプションとなります。



この例では、ツリーのリーフはオプション、コンテナーはオプションのカテゴリになっています。

**チェック ボックスのツリー ビュー**  
ユーザーは、任意の数のアイテムを選択でき、まったく選択しないこともできます。



この例では、チェック ボックスのツリー ビューで、機能の有効化/無効化を選択できます。

**ツリービュービルダー**  
ユーザーは、一度に1つのコンテナーまたは

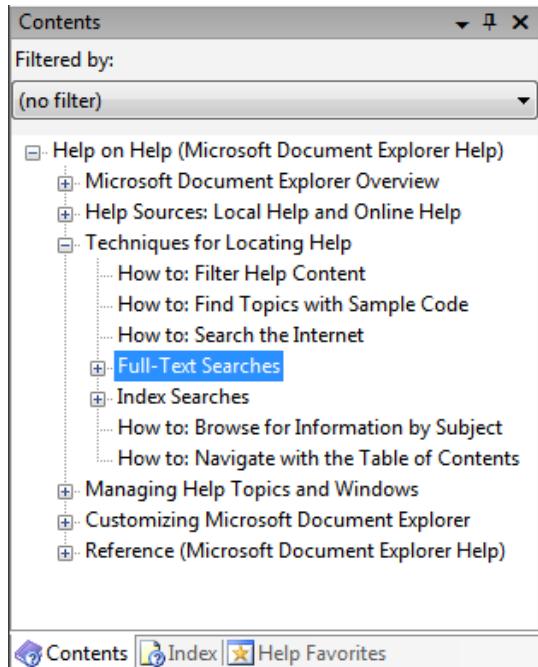
リーフを追加することによってツリーを作成できます(場合によっては順序も設定可能)。



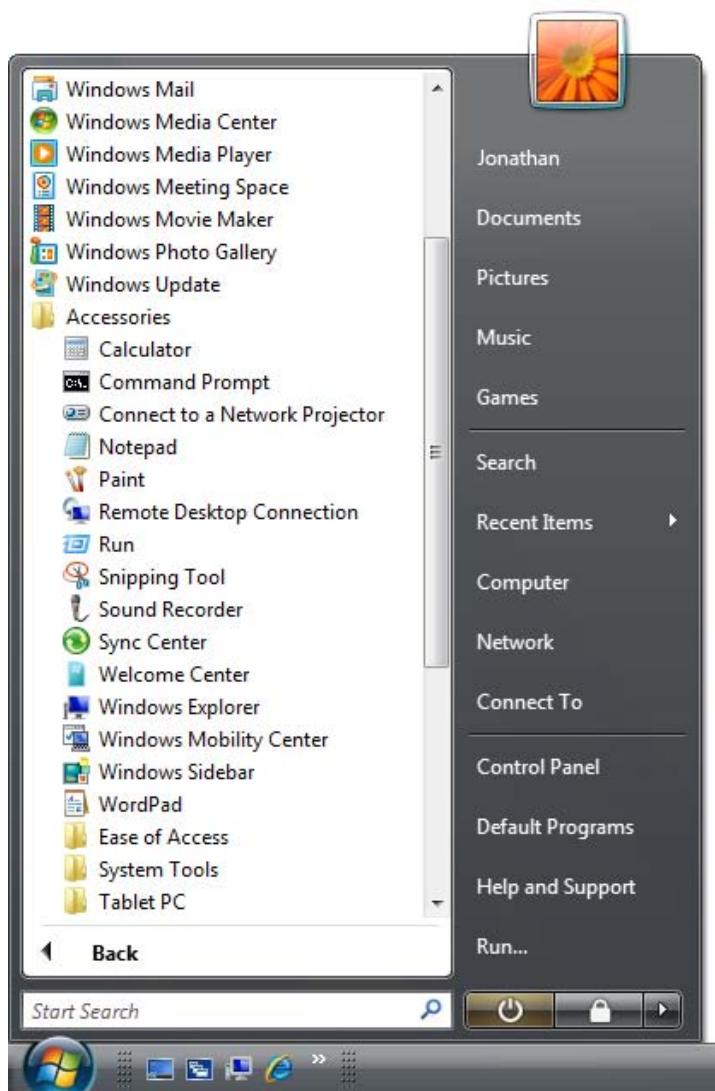
この Windows Internet Explorer の例では、ユーザーはダイアログ ボックスを使用して独自のお気に入り

一覧を作成できます。

代替のアクセス方法があるツリーでは、文字列検索、索引、フィルタリングなどの他のアクセス方法も提供する必要があります。



この例では、ユーザーは目次、索引、およびお気に入りから情報にアクセスすることもできます。ユーザーによっては、目次タブよりも索引タブや検索タブの方が便利なこともあります。

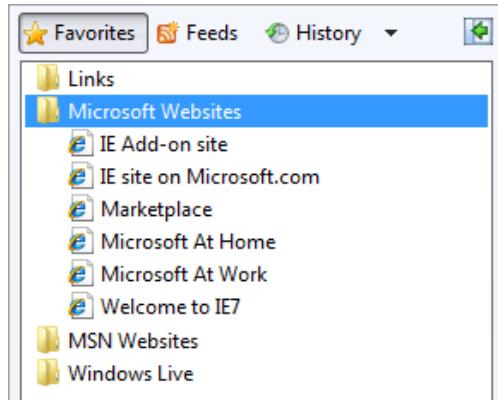


この Windows のスタートメニューの例では、ユーザーは検索ボックスに名前の一部を入力して、プログラム、ファイル、および Web ページにアクセスすることもできます。

## ガイドライン

### 提示方法

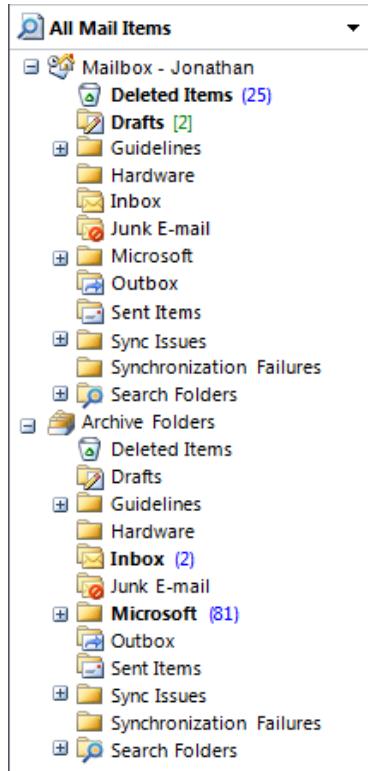
- コンテナー内では、アイテムを論理順に並べ替えます。名前はアルファベットまたは五十音順、数字は番号順、日付は時系列に基づいて並べ替えます。
- Always Show Selection 属性を使用します。この属性を使用すると、ユーザーはコントロールに入力フォーカスがないときでも選択アイテムをすぐに特定できます。
- ツリーが目次として機能する場合は、Single Expand 属性を使用してツリーの管理を簡単にします。こうすると、ツリーの関連した部分だけが展開されます。
- 空のツリーを提示しないようにします。ユーザーがツリーを作成する場合は、ツリーの初期値として、ユーザーに役立つと思われる指示や例などのアイテムを設定します。



この例では、お気に入り一覧の初期値としていくつかの例が設定されています。

- 折りたたむ意味がないコンテナー ノードは、折りたたみ可能にしません。折りたたみ可能にすると、不要な複雑性が生まれます。
- 読み込みのパフォーマンスが問題となる場合は、ツリーの最初の 2 レベルのコンテナーのみを既定で表示します。追加データは、ユーザーがツリーの分岐を展開するときに、その都度読み込まれるように設定できます。
- ユーザーがコンテナーを展開するか折りたたんだ場合、次回も同じ状態でツリービューが表示されるようにします。ただし、既定の状態で開始する方がユーザーにとって都合が良いと考えられる場合は例外です。状態を保持する単位は、ツリービューごと、ユーザーごととします。
- 高レベルのコンテナーの内容が類似している場合は、視覚的な表現も取り入れて、内容を区別できるようにします。

間違った例:



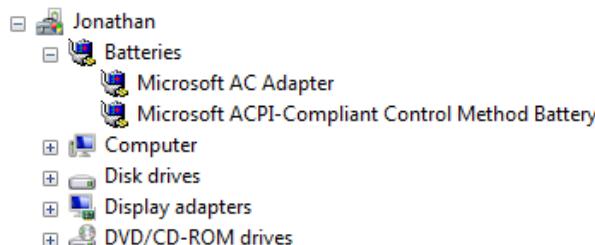
この例では、メールボックスと保管フォルダーの内容が類似しています。ツリーを下位に展開していくと、ユーザーはツリー内の位置を把握するのが難しくなり、混乱することになります。セクションに応じてアイコンをわずかに変えると、この問題に対処できます。

- 接続線を再検討します。接続線はコンテナー ノードとリーフ ノードの関係を明示するものですが、大幅な理解の向上につながるわけではありません。接続線があると表示は煩雑になります。特に、ノードどうしが近い場合や、スクロールするほど遠く離れている場合は、接続線は役に立ちません。

正しい例:



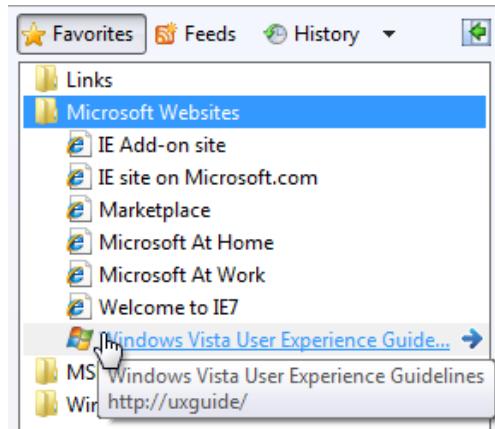
より良い例:



接続線は理解の向上にはあまり役立ちません。

#### 対話操作

- ダブルクリック時の動作を提供するようにします。ダブルクリックしたときに、アイテムを選択して既定のコマンドを実行したときと同じ結果が得られるようにします。
- ダブルクリック時の動作は他の方法でも実現できる動作にします。同じ結果をもたらすコマンド ボタンやコンテキストメニューのコマンドが常に存在している必要があります。
- アイテムに詳細な説明が必要な場合は、[情報ヒント](#)を使用して説明を提供します。



この例では、情報ヒントに追加情報を表示しています。

- 関連するコマンドのコンテキストメニューを提供します。関連するコマンドの例としては、[切り取り]、[コピー]、[貼り付け]、[削除]、[名前の変更]、[プロパティ]などがあります。
- ツリー ビューを無効にするときは、関連付けられたラベルやコマンド ボタンも無効にします。

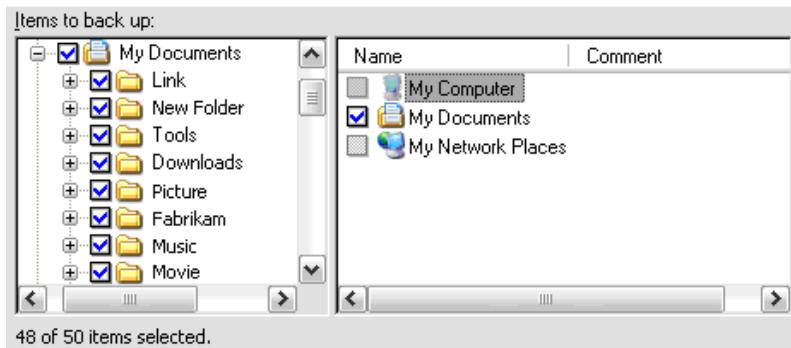
#### ツリー構成

- 大部分のユーザーにとって理解しやすい自然な階層構造を使用します。
- 自然な構造にできない場合は、見つけやすさと予測のしやすさのバランスがとれた、混乱の少ないユーザー モデルを使用するようにします。
- リスクを少なくしてアイテムをより見つけやすいツリーにするには、次のことを条件として、アイテムを複数のコンテナー内に配置します。
  - アイテムが、他の類似のアイテムに関連していないこと (ユーザーが間違った関連付けに混乱しないようにするため)。
  - 重複して配置するアイテムの数はごく少数に抑えること (ツリーが肥大化しないようにするため)。
- 効率の良い、できるだけ簡単な階層構造を使用します。このためには、次の点に留意します。
  - よくアクセスされるオブジェクトはルート ノードを除いて最初の 2 レベル以内に配置し、あまりアクセスされないオブジェクトは階層の下の方に配置します。
  - 不要なコンテナーは削除し、中間レベルの冗長なコンテナーは統合します。

- 理想的には、ツリーに含めるレベルは 4 つ以内とし、よくアクセスされるオブジェクトは最初の 2 レベル以内に表示します。
- ルートノードが本当に必要かどうかを考えます。ツリー全体に対してユーザーがコマンドを実行できるようにする必要がある場合は、ルートノードを提供します(コマンドはコンテキストメニューなどで提供します)。ツリー全体へのコマンド実行がない場合は、ルートノードを省略すると、より簡潔で使いやすいツリーになります。
- 文字列検索や索引などツリーに代替のアクセス方法がある場合は、最も役立つ内容に重点を置いた、閲覧しやすいツリーにします。代替のアクセス方法がある場合、ツリーであらゆる内容を網羅する必要はありません。簡単なツリーの方が、ユーザーは最も役立つ内容を容易に見つけることができます。

#### チェック ボックスのツリー ビュー

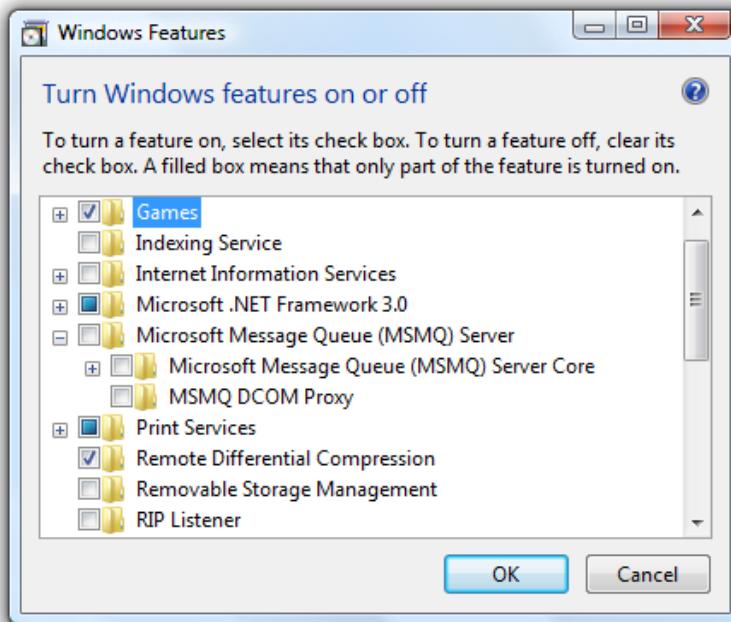
- ツリーの下部に、選択アイテムの数を表示します。ユーザーが複数のアイテムを選択する傾向が強い場合は特に、この情報は必要です。この情報があることで、ユーザーは正しく選択したことを確認できます。



この例では、ツリーの下部に選択アイテムの数が表示されており、2つのアイテムが選択されていないことがはっきりとわかります。

- 潜在的なアイテムの数が多く、すべてのアイテムをオンまたはオフにする操作の必要性が高い場合は、[すべて選択] および [すべてクリア] のコマンド ボタンを追加します。
- コンテナー内のアイテムが一部選択されていることを示すには、混在状態のチェック ボックスを使用します。

正しい例:



この例では、混在状態のチェック ボックスで、アイテムが一部選択されていることが示されています。

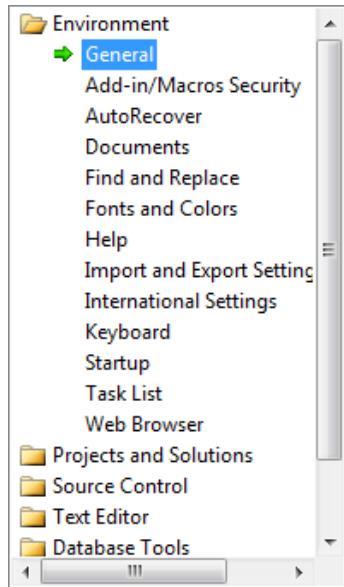
#### 推奨されるサイズと間隔



## ツリー ビュー コントロールに推奨されるサイズと間隔

- ツリー ビューの幅は、水平にスクロール操作を行わなくてもよいサイズにします。ツリー全体が展開されたときに、大部分のアイテムが見えるようにします。
- ローカライズ用に 30% の余白を残しておきます。
- ツリー ビューの高さは、垂直方向のスクロール操作ができるだけ少なくて済むサイズにします。ツリー ビューを少し長くすると垂直方向のスクロールバーの必要性が減る場合は、ツリー ビューを少し長くします。または、スペースに余裕がある分だけ長くします。

間違った例:



この例では、ツリー ビューの幅と高さを少し広げると、ほとんどの場合スクロールバーが不要になります。このツリーは、一度に 1 つのコンテナーだけを開くことができるツリーです。

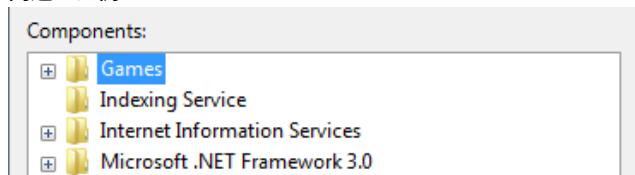
- ツリー ビューを大きくするとユーザーが使いやすい場合は、ツリー ビューとその親ウィンドウのサイズを変更できるようにします。こうすると、ユーザーはツリー ビューのサイズを必要に応じて調整できます。

## ラベル

### コントロール ラベル

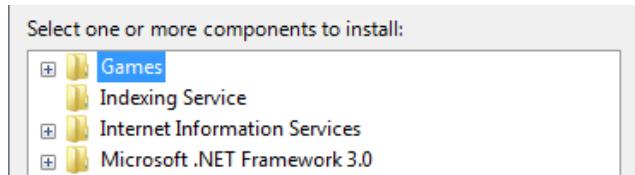
- すべてのツリー ビューにはラベルが必要です。ラベルは文ではなく語句にして、末尾にコロンを付けます。また、[静的テキスト](#)にします。
- 一意な[アクセスキー](#)を割り当てます。割り当てのガイドラインについては、「[キーボード](#)」を参照してください。
- [センテンススタイルの大文字化](#)を使用します。
- コントロールの上部にラベルを配置し、ラベルの左端とコントロールの左端を揃えます。
- 複数選択をサポートするツリー ビューのラベルでは、複数選択が可能なことを明記します。チェック ボックスのツリー ビューのラベルでは、特に明記する必要はありません。

間違った例:



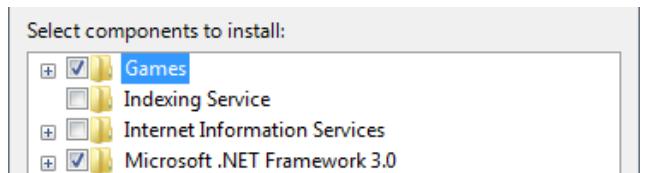
この例では、複数選択についての情報がラベルにありません。

より良い例:



この例では、複数選択が可能なことがラベルに明記されています。

最も良い例:



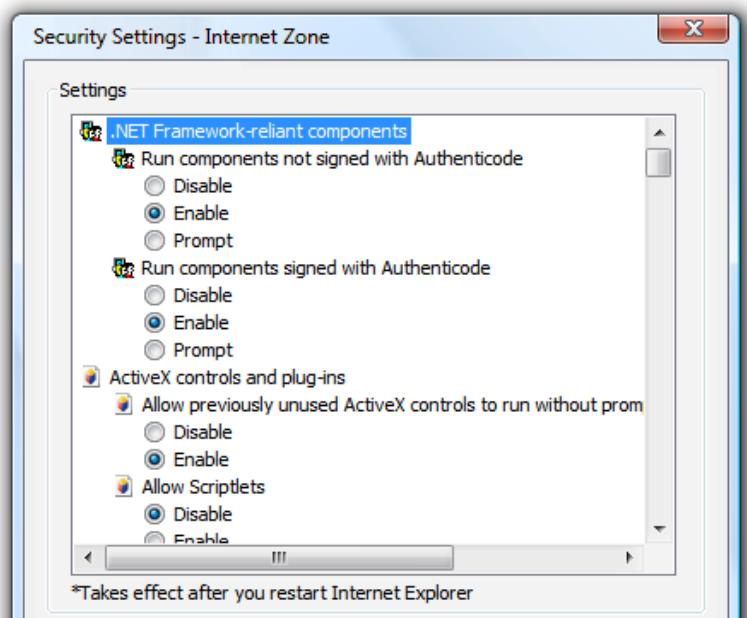
この例では、複数選択が可能なことがチェック ボックスで明示されているため、ラベルに明記する必要はありません。

#### データ テキスト

- センテンス スタイルの大文字化を使用します。

#### 指示テキスト

- ツリー ビューに関する指示テキストが必要な場合は、ラベルの上に追加します。文を使用し、末尾に句点を付けます。
- センテンス スタイルの大文字化を使用します。
- 必須ではなく参考程度の補足説明は、短くまとめます。この情報は、ラベルとコロンの間、ラベルの代わりに使用する場合はメイン指示テキストの後、またはコントロール下部のいずれかの位置に配置します。



この例では、補足説明がコントロールの下部に配置されています。

#### ドキュメント

ツリー ビューに言及するときは、以下のこと留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、アクセスキーを示すかっこや下線付き文字、およびコロンは含めません。"の一覧" という語を追加するか、通常のリストとコンテキストを区別する必要がある場合は、"の階層化された一覧" という語を追加します。
- ツリー アイテムに言及するときは、大文字と小文字の区別を含め、アイテムのテキストを正確に引用します。
- "ツリー ビュー" としてツリー ビューに言及するのは、プログラミングおよびその他の技術文書内のみとします。それ以外の場合は、多くのユーザーにとって "ツリー" という用語はわかりにくいので、"の一覧" や "の階層化された一覧" を使用します。
- ユーザー操作を説明する場合は、データの場合は "選択する" または "オン(オフ)にする" を使用し、プラスとマイナスのボタンの場合は "展開する" と "折りたたむ" を使用します。
- ラベルとツリー アイテムは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例: [目次] の一覧で、[ユーザーインターフェイス設計] を選択します。

ツリー ビュー内のチェック ボックスに言及するときは、以下のこと留意します。

- 大文字と小文字の区別を含め、ラベルのテキストを正確に引用し、"チェック ボックス" という語を追加します。アクセスキーを示すかっこや下線付き文字は含めません。
- ユーザー操作を説明する場合は、"オンにする" と "オフにする" を使用します。
- ラベルは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例: [バックアップの項目] の一覧で、[マイ ドキュメント] チェック ボックスをオンにします。

## コマンド

ここでは、Windows® ベースのアプリケーションで各コマンドを使用するためのガイドラインについて説明します。

- メニュー
- ツールバー
- リボン

# メニュー

使用パターン

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

ガイドライン

全般

メニューバー

メニューバーの非表示化

メニュー カテゴリ

メニュー項目の構成と順序

サブメニュー

提示方法

タブメニュー

コンテキストメニュー

行頭文字およびチェックマーク

アイコン

アクセスキー

ショートカットキー

標準メニュー

省略記号の使用

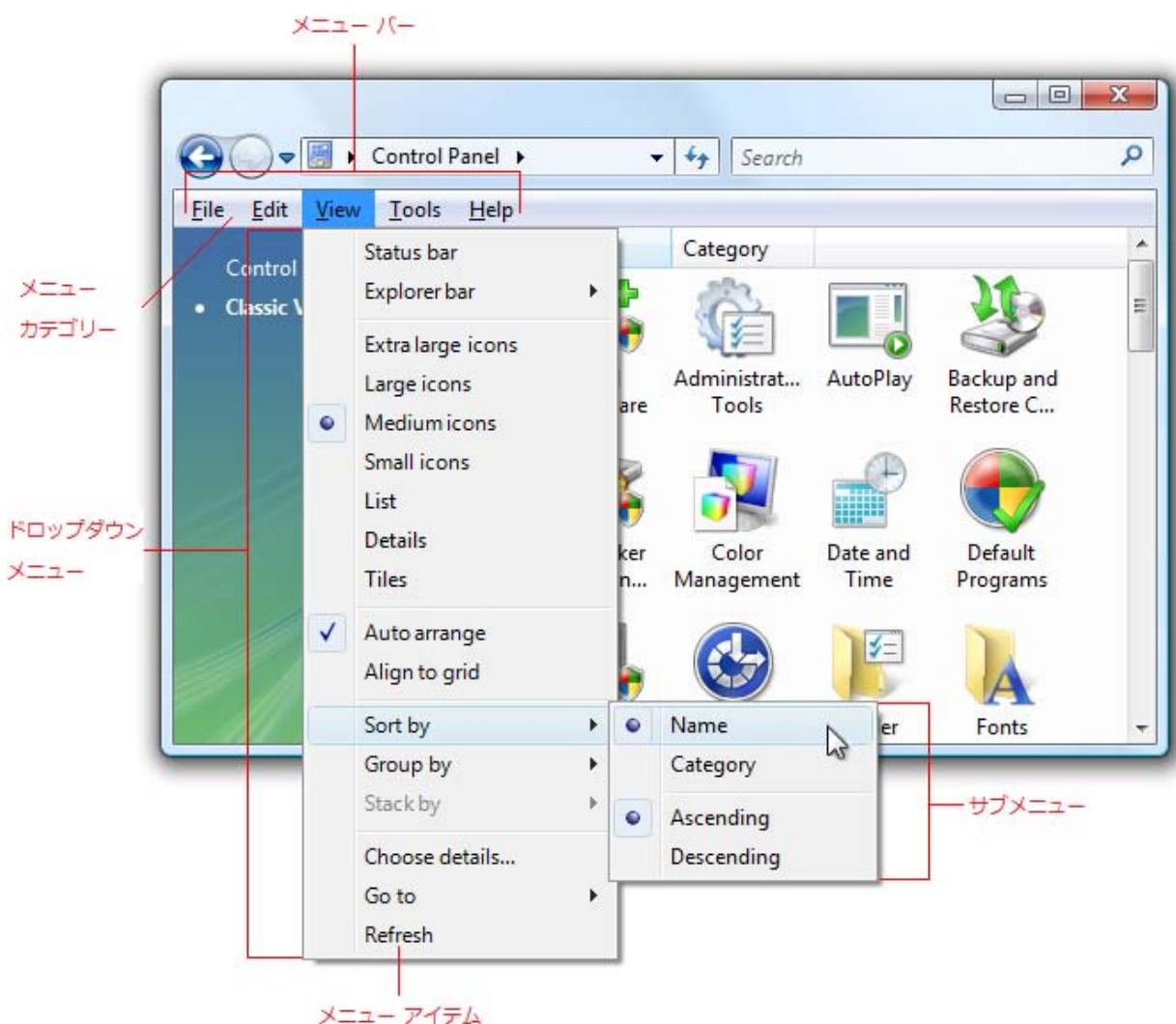
ラベル

ドキュメント

"メニュー" とは、現在のコンテキストでユーザーに提供されるコマンドまたはオプションの一覧です。

"ドロップダウンメニュー" とは、必要なときにマウスでクリックするかマウス ポインターを重ねると表示されるメニューです。通常は表示されないため、画面スペースを節約するための効率的な手段となります。"サブメニュー" または "カスケードメニュー" とは、必要なときにメニュー内から表示される二次的なメニューで、サブメニュー ラベルの末尾の矢印によって示されます。"メニュー項目" とは、メニュー内にある個々のコマンドまたはオプションです。

メニューは多くの場合、"メニューバー" から表示されます。メニューバーとは、一般にウィンドウ上部付近に配置されている、ラベル付き "メニュー カテゴリ" の一覧です。一方、"コンテキストメニュー" とは、コンテキストメニューがサポートされているオブジェクトまたはウィンドウ範囲を右クリックすると表示されるドロップダウンメニューです。



ドロップダウンメニューとサブメニューが表示された典型的なメニューバー

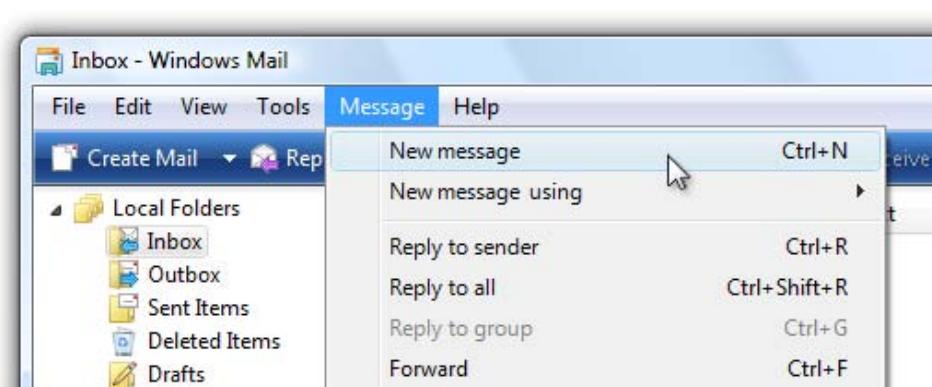
注: コマンドボタン、ツールバー、キーボード、スタートメニューに関するガイドラインは、それぞれ別の項目として記載しています。

## 使用パターン

メニューにはいくつかの使用パターンがあります。

**メニューバー** メニューバーはよく使用され、見つけやすいだけでなく、スペースを効率的に使用できます。

メニューバーでは、ドロップダウンメニュー上にコマンドとオプションが表示されます。



Windows® Mail のメニューバー

**ツールバー メニュー** ツールバーメニューは、主にメニューボタンおよび分割ボタンのコマンドで構成されるツールバーで、少数の直接コマンドが含まれる場合もあります。

ツールバーとして実装されたメニューバーで

す。

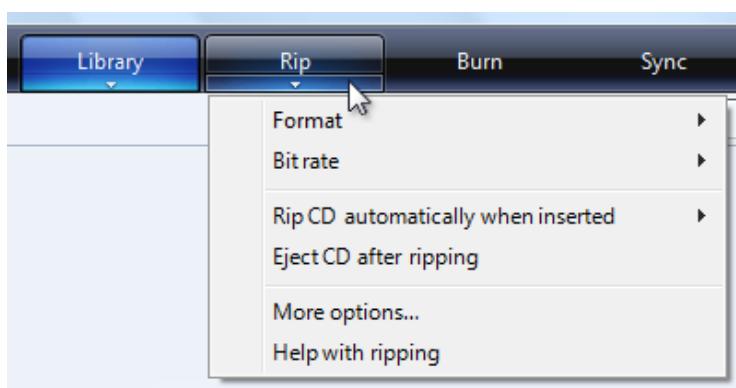


Windows フォト ギャラリーのツールバー メニュー

この使用パターンのガイドラインについては、「[ツールバー](#)」を参照してください。

タブ メニュー  
タブに関連する  
いくつかのコマ  
ンドとオプショ  
ンをドロップダ  
ウン メニュー上  
に表示する、タ  
ブ内のボタンで  
す。

メニューを持つタブは通常のタブと同じように見えますが、その下部にはドロップダウン矢印の付いたボタンがあります。ボタンをクリックすると、タブの選択を行うのではなく、ドロップダウンメニューが表示されます。



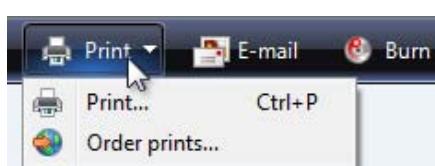
Windows Media Player に使用されているタブ メニュー

メニュー ボタン  
関連するいくつ  
かのコマンドを  
ドロップダウン  
メニューに表示  
するコマンドボ  
タンです。

メニュー ボタンは通常のコマンド ボタンと同じように見えますが、領域内にドロップダウン矢印があります。ボタンをクリックすると、コマンドを実行するのではなく、ドロップダウンメニューが表示されます。

分割ボタンはメニュー ボタンとよく似ていますが、コマンドのバリエーションであり、ボタンの左部

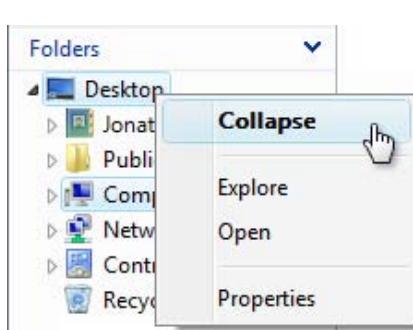
分をクリックするとラベル上のアクションが直接実行されます。



関連するいくつかのコマンドをまとめたメニュー ボタン

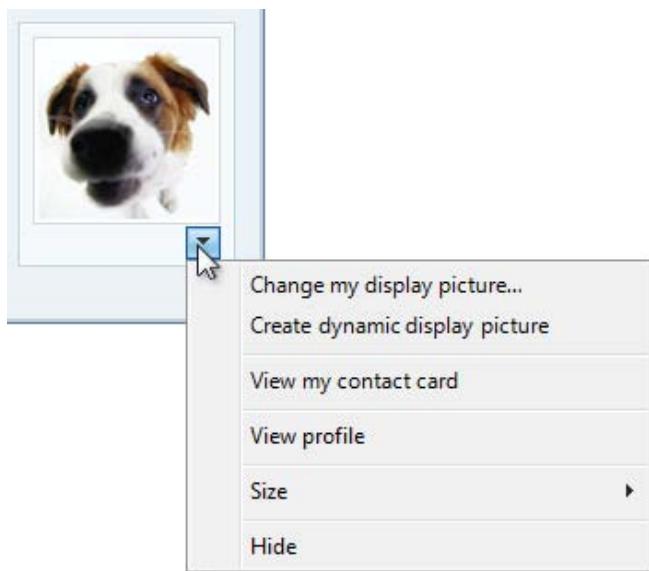
コンテキストメ  
ニュー  
現在のコンテキ  
ストに関連する  
いくつかのコマ  
ンドとオプショ  
ンを表示するド  
ロップダウンメ  
ニューです。

コンテキストメニューは、コンテキストメニューがサポートされているオブジェクトまたはウィンドウ範囲を右クリックするとドロップダウンメニューとして表示されます。



Windows エクスプローラーのコンテキスト メニュー

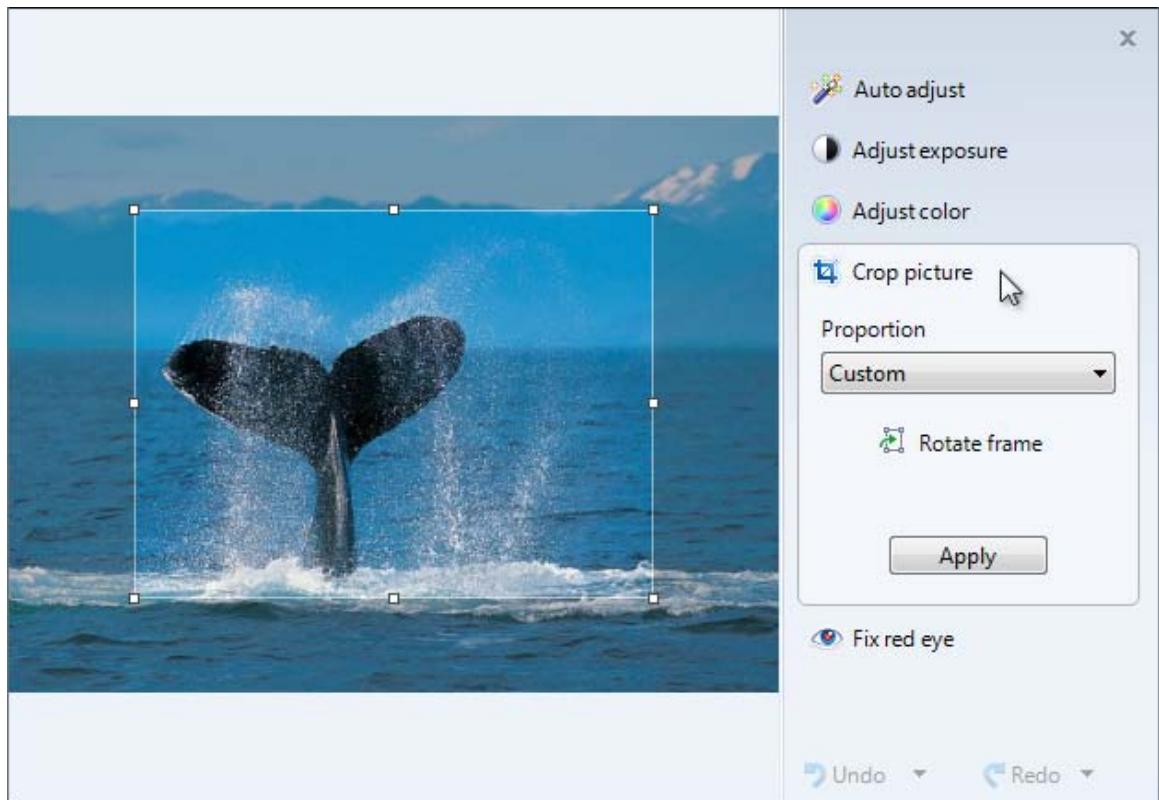
コンテキストメニューはメニュー選択方法として最適であるものの、すべてのユーザーに適した方法  
が必要な場合は、メニューのドロップダウン矢印ボタンを使用できます。



メニューのドロップダウン矢印ボタンを使用して表示されたコンテキストメニュー

作業ウィンドウメニュー コンテキストメニューとは異なり、必要とするかどうかに関係なく、ウィンドウ枠内に自動的に表示されます。

選択されたオブジェクトまたはプログラム モードに関連するいくつかのコマンドです。



Windows フォト ギャラリービューアーの作業ウィンドウメニュー

### 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

#### メニューバー

以下の基準を適用します。

- ・ ウィンドウがメインウィンドウかどうか。
- ・ メニュー項目が多数存在するかどうか。
- ・ メニュー カテゴリが多数存在するかどうか。
- ・ メニュー項目の大多数がプログラム全体とメイン ウィンドウに適用されるかどうか。
- ・ 全ユーザーが使用する必要のあるメニューであるかどうか。

該当する場合は、メニューバーを使用することを検討します。

## ツール バー メニュー

以下の基準を適用します。

- ・ ウィンドウがメイン ウィンドウかどうか。
- ・ ウィンドウにツールバーがあるかどうか。
- ・ ごく少数のメニュー カテゴリが存在しているかどうか。
- ・ 全ユーザーが使用する必要のあるメニューであるかどうか。

該当する場合は、メニュー バーの代わりに、またはメニュー バーに加えてツール バー メニューを使用することを検討します。

## タブ メニュー

以下の基準を適用します。

- ・ ウィンドウがメイン ウィンドウかどうか。
- ・ ウィンドウにタブがあるかどうか。各タブが、特定の目的のためのタスクをまとめるために使用されているかどうか（タブを使用して別のビューを表示するのではない）。
- ・ 各タブに適用されるメニュー カテゴリが 1 つ存在するかどうか。
- ・ 多数のコマンドとオプションが存在するが、それぞれのタブにはごく少数存在するだけであるかどうか。

該当する場合は、メニュー バーの代わりにタブ メニューを使用することを検討します。

## コンテキスト メニュー

以下の基準を適用します。

- ・ 選択されたオブジェクトまたはウィンドウ範囲に適用する、コンテキスト依存コマンドとオプションがいくつか存在するかどうか。
- ・ これらのメニュー項目が重複しているかどうか。
- ・ ターゲット ユーザーがコンテキスト メニューに慣れているかどうか。

該当する場合は、必要としているオブジェクトやウィンドウ範囲に、コンテキスト メニューを提供します。

ブラウザベースのプログラムの場合は、コンテキスト依存コマンド用に作業 ウィンドウ メニューがよく使用されます。現在のところ、ブラウザベースのプログラムのコンテキスト メニューは、汎用的で有用性がないと考えられています。

## 作業 ウィンドウ メニュー

以下の基準を適用します。

- ・ ウィンドウがメイン ウィンドウかどうか。
- ・ 選択されたオブジェクトまたはプログラム モードに適用する、コンテキスト依存コマンドとオプションがいくつか存在するかどうか。
- ・ 少数のメニュー カテゴリが存在しているかどうか。
- ・ 全ユーザーが使用する必要のあるメニューであるかどうか。

該当する場合は、コンテキスト メニューの代わりに作業 ウィンドウ メニューを使用することを検討します。

## デザイン コンセプト

ユーザー エクスペリエンスを向上させる効果的なメニューを以下に示します。

- ・ プログラムの種類、ウィンドウの種類、コマンドの使用法、およびターゲットとするユーザーに適した[コマンドの提示方法](#)を使用します。
- ・ 適切な場合は[標準メニュー構成](#)を使用し、[適切に構成](#)します。
- ・ [メニュー バー](#)、ツール バー、コンテキスト メニューを効果的に使用します。
- ・ [アイコン](#)を効果的に使用します。
- ・ [アクセス キー](#)および[ショートカット キー](#)を効果的に使用します。

## 最も重要な点

プログラムの種類、ウィンドウの種類、コマンドの使用法、およびターゲットとするユーザーに適したコマンドの提示方法を選択します。

詳細と例については、「[メニューのデザインコンセプト](#)」を参照してください。

## ガイドライン

### 全般

- メニュー バーを除き、すべてのメニュー パターンには、プルダウン メニューの存在を示すため [ドロップダウン矢印](#)が必要です。メニュー バーにはメニューはもちろん存在しますが、他のメニュー パターンではそうではありません。
- メニュー 項目の名前を動的に変更しないでください。これは予想外の動作であり、混乱を招きます。たとえば、[縦長] オプションを選択内容に応じて [横長] に変更しないでください。モードについては、代わりに[行頭文字やチェックマーク](#)を使用します。
  - 例外: オブジェクト名をベースとするメニュー 項目の名前は動的に変更させることができます。たとえば、最近使用したファイルまたはウィンドウの名前の一覧については、動的に変更できます。

### メニュー バー

- メニュー カテゴリが 3 個以下のメニュー バーを作成しないようにします。ごく少数のコマンドしか存在しない場合は、代わりにツール バー メニューのような軽い方法や、コマンド ボタンとリンクのような直接的な方法を選択します。
- メニュー カテゴリが 10 個を超えないようにします。メニュー カテゴリが多すぎると煩雑になり、メニュー バーが使いにくくなります。
- ツール バーまたは直接コマンドによって大半のユーザーが必要とするコマンドをすべて提供できる場合は、メニュー バーを非表示にします。ツール バー メニューの [メニュー バー] チェックマーク オプションを使用して、表示と非表示の切り替えができるようにします。



この例では、Windows Internet Explorer® に [メニュー バー] オプションが用意されています。

詳細については、「[メニュー バーの非表示化](#)」を参照してください。

### メニュー バーの非表示化

通常、ツール バーとメニュー バーは、競合することなくそれぞれの長所を活用でき、ツール バーとメニュー バーは組み合わせることによって最大限の効果が得られます。

- ツール バーのデザイン上、メニュー バーを表示するとしつこく感じる場合は、既定でメニュー バーを非表示にします。

- キーボードユーザーにとってはメニュー バーの方が使いやすいため、メニュー バーを完全に削除するのではなく、非表示にします。
- メニュー バーを復元する手段として、[表示] メニュー カテゴリ (メイン ツール バー) または [ツール] メニュー カテゴリ (補助 ツール バー) で [メニュー バー] チェックマーク オプションを用意します。詳細については、「[標準メニューと分割ボタン](#)」を参照してください。

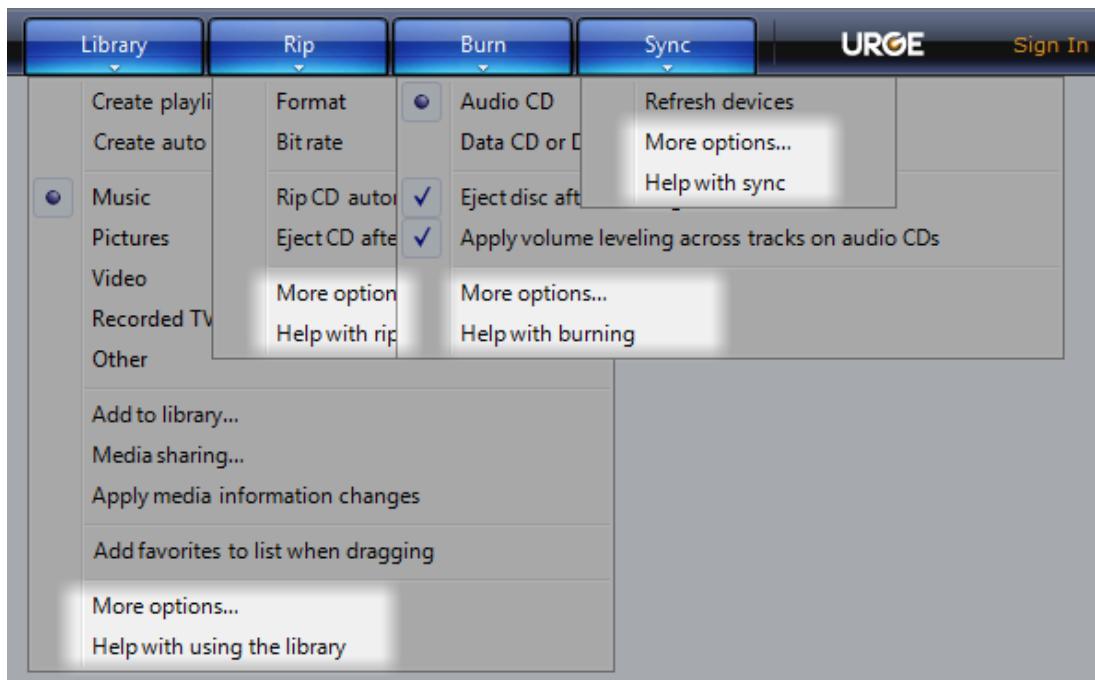
## メニュー カテゴリ

- メニュー カテゴリには 1 語の名前を選択します。複数の語を使用するとカテゴリ間の切れ目がわかりにくくなります。
- ドキュメントを作成または表示するプログラムについては、[ファイル]、[編集]、[表示]、[ツール]、[ヘルプ] のような [標準メニュー](#) カテゴリを使用します。そうすると、よく使用するメニュー項目の場所が予測できるようになります。
- その他の種類のプログラムについては、コマンドとオプションを、プログラムの目的とユーザーのタスクや目標に応じた、有用で自然なカテゴリに整理することを検討します。プログラムに適切ではない場合は、標準メニュー構成を使用する必要はありません。
- 非標準のメニュー カテゴリの使用を選択する場合は、適切なカテゴリ名を選択する必要があります。詳細については、「[ラベル](#)」を参照してください。
- タスク指向のメニュー カテゴリを、汎用的なカテゴリよりも優先します。タスク指向のカテゴリを使用すると、メニュー項目を見つけやすくなります。



この例では、Windows Media Player にタスク指向のメニュー カテゴリが使用されています。

- メニュー カテゴリに含まれるメニュー項目が 1、2 個だけにならないようにします。理にかなっている場合は、サブメニューを使用して他のメニュー カテゴリに統合します。
- 以下の場合に限り、同じメニュー項目を複数のカテゴリに配置します。
  - メニュー項目が論理的に複数のメニュー カテゴリに属している。
  - その項目を単一のメニュー カテゴリに配置すると発見しにくくなることが、データによって明らかになっていきます。
  - 複数のカテゴリに含まれるのは、見つけにくいメニュー項目が 1、2 個のみである。
- 異なるメニュー項目を同じ名前で複数のカテゴリに配置しないようにします。たとえば、複数のカテゴリにそれぞれ異なる [オプション] メニュー項目を配置しないようにします。
  - 例外: タブ メニューのパターンでは、各タブ メニューにそれぞれ異なる [オプション] や [ヘルプ] メニュー項目が含まれることがあります。



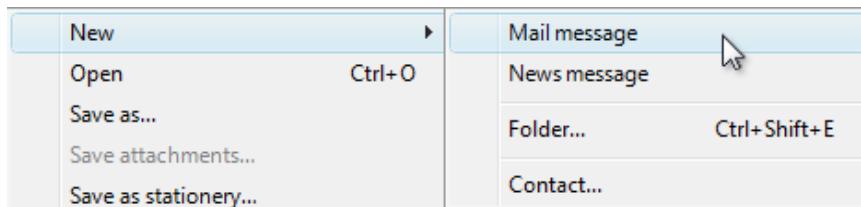
この例では、Windows Media Player の各タブ メニューに、[オプション] および [ヘルプ] メニュー項目が含まれています。

## メニュー項目の構成と順序

- メニュー項目を、7個以下の関連性の強い項目から成るグループにまとめます。この場合、サブメニューは親メニューの1つのメニュー項目として数えます。
- 1つのメニュー階層に25個よりも多くの項目を配置しないようにします(サブメニュー項目は含めず)。
- メニュー内のグループ間には区切り記号を配置します。区切り記号とは、メニューの幅に合わせて引かれた1本の線です。
- メニュー内では、論理的な順序に従ってグループを配置します。論理的な順序が存在しない場合は、最も使用頻度の高いグループを先頭に配置します。
- グループ内では、論理的な順序に従って項目を配置します。論理的な順序が存在しない場合は、最も使用頻度の高い項目を先頭に配置します。数値項目(拡大率の%値など)は数字順に配置します。

## サブメニュー

- 必要のない場合はサブメニューを使用しないようにします。サブメニューの使用には多くの手間がかかるうえに、一般に、メニュー項目を探し出すことが難しくなります。
- 使用頻度の高いメニュー項目をサブメニューに配置しないようにします。そうすると、こうしたコマンドを使用する際の効率が悪くなります。ただし、よく使用するコマンドが通常、ツールバーなどから直接アクセスされる場合は、サブメニューに配置することができます。
- 以下の場合に、サブメニューの使用を検討します。
  - 親メニューに多数の項目(20以上)が含まれるため、サブメニューを使用するとすっきりする。または、サブメニューが7個より多い項目から成るグループの一部となっている。
  - サブメニュー内の項目は、親メニュー内の項目よりも使用頻度が低い。
  - サブメニューには3個以上の項目が含まれる予定である。
  - 同一の語から始まるコマンドが3つ以上ある。この場合は、その語をサブメニューのラベルに使用します。



この例では、「[新規作成]」サブメニューが、「[新しいメッセージ]」、「[新しいニュース メッセージ]」、「[新しいフォルダー]」、「[新しい連絡先]」として別々のコマンドになります。

- 3階層までのメニューを使用します。つまり、メインメニューのほかに、2階層までのサブメニューを作ることができます。2階層のサブメニューはめったに使用されないものである必要があります。

## 提示方法

- 現在のコンテキストに適用されないメニュー項目は、削除するのではなく無効にします。そうすると、メニューバーの内容は一貫性のあるものとなり、発見しやすくなります。次の場合は例外です。
  - コンテキストメニュー カテゴリの場合、現在のコンテキストに適用されないコンテキストメニュー項目は、無効化するのではなく、削除します。特定のオブジェクトの種類が選択されたときなど、あるメニュー カテゴリが特定のモードに対してのみ表示されるときは、コンテキストに依存しています。詳細については、コンテキストメニューの削除と無効化のガイドラインを参照してください。
  - メニュー項目を無効にするタイミングを算出するとパフォーマンスに重大な問題が生じる場合は、メニュー項目はアクティブのままにし、必要に応じてその選択結果をエラーメッセージに含めておきます。

## タブメニュー

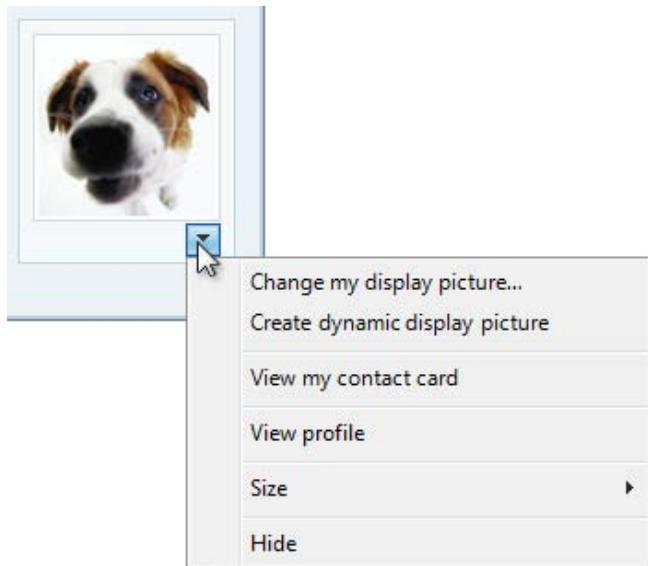
- 各タブメニューには、コンテキストに特有の「[オプション]」および「[ヘルプ]」メニュー項目を含めることができます。この点が、他のすべてのメニュー パターンと大きく異なっています。それぞれのタブが、特定の目的のためのタスクをまとめるために使用されるので、タブメニュー間で重複があっても混乱しません。

## コンテキストメニュー

- コンテキストメニューは、コンテキスト依存コマンドとオプションに対してのみ使用します。メニュー項目は、プログラム全体でなく、選択された(またはクリックされた)オブジェクトまたはウィンドウ範囲にのみ適用される必要があります。
- コンテキストメニューの使用によってのみ、コマンドを使用できるようにすることは避けます。コンテキストメニューはショートカットキーと同じように、コマンドの実行とオプションの選択の代替手段です。たとえば「[プロパ

ティ] コマンドは、メニュー バーを使用するか、Alt + Enter キーのアクセス キーで使用できます。

- コンテキスト依存コマンドとオプションをいくつかまとめるとメリットが得られる場合は常に、オブジェクトとウインドウ範囲に対してコンテキストメニューを用意します。通常、多くのユーザーは、右クリックによってコンテキストメニューを表示しようと考えます。
- すべてのユーザーを対象とするコンテキストメニューには、メニューのドロップダウン矢印ボタンを使用します。コンテキストメニューは、通常、上級ユーザーを対象とするコマンドとオプションを提供するのに適しています。ただし、コンテキストメニューがメニュー選択方法として最適であり、すべてのユーザーを対象とする必要がある場合は、メニューのドロップダウン矢印ボタンを使用できます。



この例では、コンテキストメニューを表示するのにメニューのドロップダウン矢印ボタンが使用されています。

#### メニュー項目の構成と順序

- メニュー項目を、7個以下の関連性の強い項目から成るグループにまとめます。
- コンテキストメニューをシンプルで直接的、効率的なものにするために、サブメニューは使用しないようにします。
- コンテキストメニュー内の項目数は、15個を上限とします。
- メニュー内のグループ間には区切り記号を配置します。区切り記号とは、メニューの幅に合わせて引かれた1本の線です。
- 次の順序に従ってメニュー項目を配置します。

##### 主コマンド (最もよく使用されるもの)

開く

実行

再生

印刷

<区切り記号>

##### オブジェクトをサポートする副コマンド

<区切り記号>

##### 転送コマンド

切り取り

コピー

貼り付け

<区切り記号>

##### オブジェクト設定

<区切り記号>

##### オブジェクトコマンド

削除

名前の変更

<区切り記号>

プロパティ

#### 提示方法

- 既定のコマンドを太字で表示します。実用的な場合は、先頭のメニュー項目にします。既定のコマンドは、ユーザーが

オブジェクトをダブルクリックするか、選択して Enter キーを押すと実行されます。

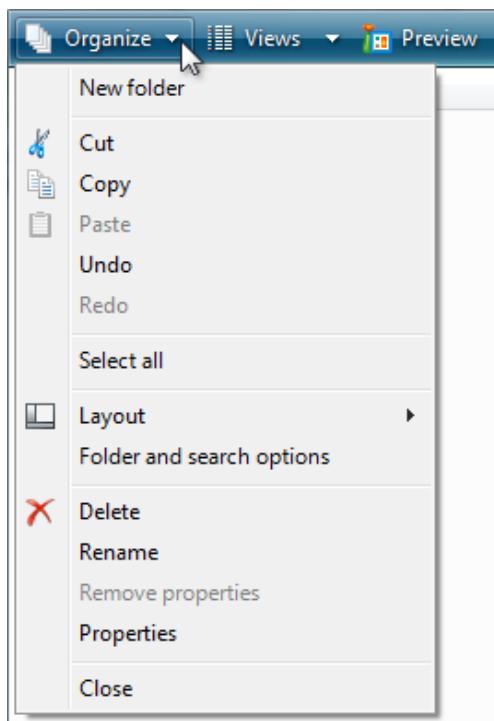
- 現在のコンテキストに適用されないコンテキストメニュー項目は、無効化するのではなく、削除します。そうすると、コンテキストメニューがコンテキストを反映したものになり、効率がよくなります。
  - 例外: 以下に示すように、使用できるものと期待されているメニュー項目の場合は、適用されない項目を無効にします。
    - [切り取り]、[コピー]、[貼り付け]、[削除]、[名前の変更]など、関連する標準のコンテキストメニュー命令を常に用意している。
    - 常に、一連の関連コマンドを実行できるようにコマンドが用意されている。たとえば、[戻る]を配置する場合は常に[進む]も用意し、[切り取り]を配置する場合は常に[コピー]と[貼り付け]を用意しているなど。

## 行頭文字およびチェックマーク

- オプションとなるメニュー項目には行頭文字およびチェックマークを使用することができます。コマンドには使用できません。
- 相互に排他的ないくつかの選択肢から 1 つを選び出すには、行頭文字を使用します。グループには必ず 2 つ以上の行頭文字が存在することになります。詳細については、「[ラジオ ボタン](#)」を参照してください。
- 独立した設定の有効と無効を切り替えるには、チェックマークを使用します。オンとオフの状態が、互いに正反対の状態を表していない場合は、代わりに行頭文字セットを使用します。詳細については、「[チェック ボックス](#)」を参照してください。
- 混在状態のチェックマークには、チェックマークなしでメニュー項目を表示します。混在状態は、複数の選択対象に、このオプションがすべてのオブジェクトではなく一部のオブジェクトに対して設定されていることを示すために使用します。したがって、個々のオブジェクトにはオンかオフのいずれか一方の状態が適用されています。混在状態は、独立したアイテムの第 3 の状態としては使用しません。
- 関連するチェックマークまたは行頭文字のセットの間に区切り記号を配置します。区切り記号とは、メニューの幅に合わせて引かれた 1 本の線です。

## アイコン

- 次のメニュー項目には、メニュー項目アイコンの使用を検討します。
  - 最も使用頻度の高いメニュー項目。
  - アイコンが標準的でなじみ深いメニュー項目。
  - アイコンがコマンド内容を適切に表現しているメニュー項目。
- アイコンを使用する場合、すべてのメニュー項目にアイコンを表示する必要があると考える必要はありません。わかりにくいアイコンを使用すると有用ではないうえに、見た目が煩雑になり、重要なメニュー項目に注目できなくなります。



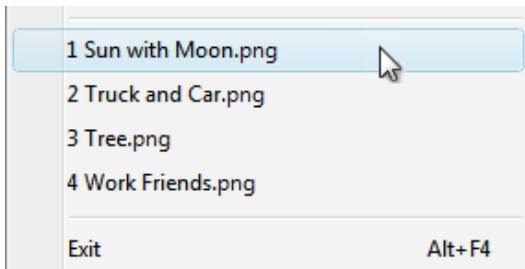
この例の [編集] メニューには、最もよく使用されるメニュー項目に対してのみアイコンが表示されています。

- メニュー アイコンを [Aero style のアイコンのガイドライン](#)に準拠させるようにします。

詳細と例については、「[アイコン](#)」を参照してください。

## アクセス キー

- すべてのメニュー項目にアクセス キーを割り当てます。例外はありません。
- 可能な限り、[標準的なアクセス キーの割り当て](#)に従って、よく使用されるコマンドにアクセス キーを割り当てます。アクセス キーを一貫して常に割り当てる事はできません。しかし、特に、頻繁に使用されるコマンドには一貫性を持たせることをお勧めします。
- 動的なメニュー項目(最近使用されたファイルなど)については、数値的にアクセス キーを割り当てます。



この例では、Windows のペイント プログラムで、数字のアクセス キーを最近使用されたファイルに割り当てています。

- メニュー階層内で一意となるアクセス キーを割り当てます。別のメニュー階層ではアクセス キーを再び使用することができます。
- 次のようにして、アクセス キーを見つけやすくなります。
  - 最も使用頻度が高いメニュー項目については、ラベルの 1 番目または 2 番目(できれば 1 番目)の単語の先頭文字を選択します。
  - 使用頻度の低いメニュー項目については、ラベル内の特徴的な子音または母音を選択します。
- できるだけ幅の広い文字を使用します。たとえば、w、m、大文字などを使用します。
- できるだけ特徴のある子音または母音を使用します。たとえば、"Exit"(終了)の "x" などを使用します。
- 下線が見えにくくなる文字を使用しないようにします。以下に、問題点が大きいものから順に示します。
  - 1 ピクセルしか幅のない文字(i、lなど)。
  - ディセンダーのある文字(g、j、p、q、yなど)。
  - ディセンダーのある文字の隣にある文字。

その他のガイドラインと例については、「[キーボード](#)」を参照してください。

## ショートカット キー

- ショートカット キーは、最もよく使用されるメニュー項目にのみ割り当てます。使用頻度の低いメニュー項目の場合、代わりにアクセス キーが使用されるので、ショートカット キーは必要ありません。
- ショートカット キーがタスクを実行する唯一の方法にならないようにします。マウスやキーボードの Tab キー、方向キー、アクセス キーも使用できるようにする必要があります。
- よく知られているショートカット キーについては、標準の割り当てを使用します。Windows プログラムで使用され、よく知られているショートカット キーについては、「[Windows のキーボードショートカット キー](#)」を参照してください。
- よく知られているショートカット キーに別の機能を割り当てないようにします。よく知られているショートカット キーはユーザーに記憶されているので、機能に一貫性がないと、ストレスやエラーの原因になります。Windows プログラムで使用され、よく知られているショートカット キーについては、「[Windows のキーボードショートカット キー](#)」を参照してください。
- システム全体に作用するショートカット キーをプログラムに割り当てないようにします。プログラムのショートカット キーは、プログラムに入力フォーカスがある場合にのみ効果を発揮します。
- ショートカット キーはすべてドキュメントに記載します。そうすると、ショートカット キーの割り当てを確認する際に役立ちます。
  - 例外: コンテキストメニュー内にはショートカット キーの割り当てを表示しません。コンテキストメニューでは効率が最優先されるので、ここにはショートカット キーの割り当てを表示しません。
- 非標準のキー割り当てについては、次のようにします。
  - 標準の割り当てが行われていないショートカット キーを選択します。標準のショートカット キーを再割り当てないようにします。

- ・ プログラム全体で一貫して非標準のキー割り当てを使用します。別のウィンドウでは別の意味を持つようには割り当てません。
- ・ 可能であれば、使用頻度の高いコマンドの場合は特に、記憶を助けるキー割り当てを選択します。
- ・ 選択されたオブジェクトに適用されるコマンドなど、小規模の効果を発揮するコマンドにはファンクションキーを使用します。たとえば、F2 キーを押すと選択項目の名前を変更できます。
- ・ ドキュメント全体に適用されるコマンドなど、大規模な効果を発揮するコマンドには Ctrl キーと組み合わせて使用します。たとえば、Ctrl キーを押しながら S キーを押すと、現在のドキュメントが保存されます。
- ・ 標準のショートカットキーの操作を拡張または補完するコマンドには Shift キーと組み合わせて使用します。たとえば、Alt キーを押しながら Tab キーを押すと、開いているメイン ウィンドウが順に表示されますが、Alt キーと Shift キーを押しながら Tab キーを押すと、逆の順序で表示されます。同じように、F1 キーを押すとヘルプが表示されますが、Shift キーを押しながら F1 キーを押すと、コンテキストに応じたヘルプが表示されます。
- ・ ショートカットキーに次の文字を使用しないでください。@ £ \$ {} [] \ ~ | ^ ' < >。これらの文字は、言語ごとに異なるキーの組み合わせが必要か、ロケールに特有のものです。
- ・ Ctrl キーと Alt キーの組み合わせは使用しないでください。Windows では、一部の言語バージョンでこの組み合わせを AltGR キーと解釈し、英数字を生成するためです。
- ・ プログラムで多数のショートカットキーを割り当てる場合、割り当てのカスタマイズ機能を用意します。そうすると、ユーザーが他製品と競合するショートカットキーを割り当て直すことができ、他製品からの移行が実現します。ほとんどのプログラムでは、この機能が必要になるほど多くのショートカットキーを割り当てません。

その他のガイドラインと標準のショートカットキー割り当てについては、「[キーボード](#)」を参照してください。

## 標準メニュー

- ・ ドキュメントを作成または表示するプログラムには、標準メニュー構成を使用します。標準メニュー構成が使用されていると、よく使用するメニュー項目が予測可能になり、発見しやすくなります。
- ・ その他の種類のプログラムには、妥当な場合にのみ標準メニュー構成を使用します。コマンドとオプションは、プログラムの目的とユーザーのタスクや目標に応じた、有用で自然なカテゴリに整理することを検討します。

## 標準メニュー バー

標準メニュー バーの構成を次に示します。この一覧は、メニュー カテゴリとメニュー項目のラベル、順序(区切り記号を含む)、アクセスキーとショートカットキー、省略記号を示すものです。

### ファイル(E)

新規作成(N)	Ctrl + N
開く(O)...	Ctrl + O
閉じる(C)	
<区切り記号>	
上書き保存(S)	Ctrl + S
名前を付けて保存(A)...	
<区切り記号>	
送信(D)	
<区切り記号>	
印刷(P)...	Ctrl + P
印刷プレビュー	
ページ設定	
<区切り記号>	
1<ファイル名>	
2<ファイル名>	
3<ファイル名>	
...	
<区切り記号>	
終了(X)	Alt + F4 (通常はショートカットキーなし)

### 編集(E)

元に戻す(U)	Ctrl + Z
繰り返し(R)	Ctrl + Y
<区切り記号>	
切り取り(I)	Ctrl + X
コピー(C)	Ctrl + C
貼り付け(P)	Ctrl + V

<区切り記号>  
すべて選択(L) Ctrl + A  
<区切り記号>  
削除(D) Del (通常はショートカットキーなし)  
<区切り記号>  
検索(E)... Ctrl + F  
次を検索 F3 (通常はコマンドなし)  
置換(E)... Ctrl + H  
ジャンプ... Ctrl + G

#### 表示(V)

ツールバー(I)  
ステータスバー(B)  
<区切り記号>  
ズーム(Z)  
拡大 Ctrl + +  
縮小 Ctrl + -  
<区切り記号>  
全画面表示(E) F11  
最新の情報に更新(R) F5

#### ツール(I)

...  
<区切り記号>  
オプション(O)

#### ヘルプ(H)

<プログラム名> ヘルプ F1  
<区切り記号>  
<プログラム名> バージョン情報(A)

#### 標準ツールバー メニューのボタン

標準ツールバー メニューのボタンを次に示します。この一覧は、メニュー カテゴリとメニュー項目のラベル、順序(区切り記号を含む)、ショートカットキー、省略記号を示すものです。

#### ツール

全画面表示 F11 ([検索]も使用される場合はアクセスキーを再割り当て)  
ツールバー (メニュー バーコマンドがここに配置されます)  
<区切り記号>  
印刷...  
検索...  
<区切り記号>  
ズーム  
文字サイズ  
<区切り記号>  
オプション

#### 編集

新しいフォルダー Ctrl + N  
<区切り記号>  
切り取り Ctrl + X  
コピー Ctrl + C  
貼り付け Ctrl + V  
<区切り記号>  
すべて選択 Ctrl + A  
<区切り記号>  
削除 Del (通常はショートカットキーなし)  
名前の変更  
<区切り記号>  
オプション

#### ページ

新しいウィンドウを開く Ctrl + N

<区切り記号>

ズーム

文字サイズ

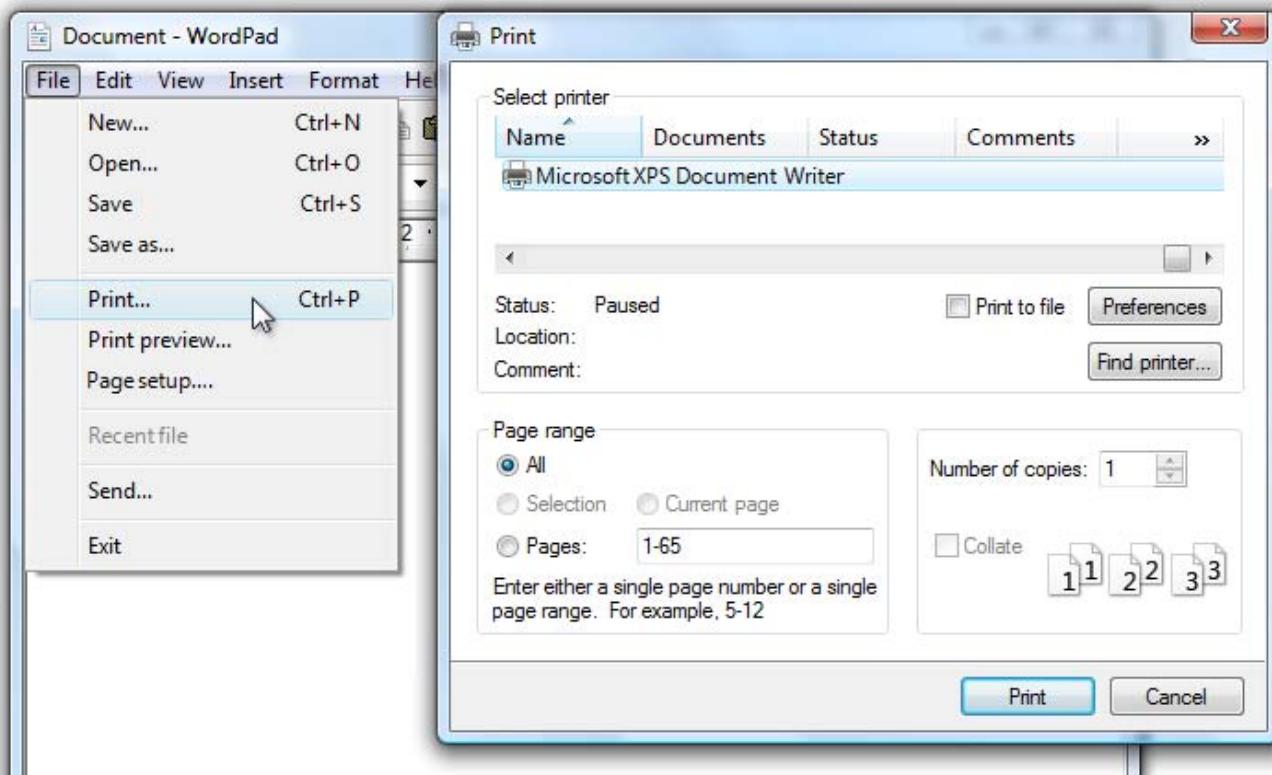
## 標準コンテキストメニュー

標準コンテキストメニューの内容を次に示します。この一覧は、メニュー項目のラベル、順序(区切り記号を含む)、アクセスキー、省略記号を示すものです。コンテキストメニューにはショートカットキーは表示されません。

- 開く(O)
- 実行(R)
- 再生(P)
- 編集(E)
- 印刷(N)...
- <区切り記号>
- 切り取り(I)
- コピー(C)
- 貼り付け(P)
- <区切り記号>
- 削除(D)
- 名前の変更(M)
- <区切り記号>
- <オブジェクト名>を固定する(L) (チェックマーク)
- プロパティ(R)

## 省略記号の使用

メニュー命令は即座に実行する操作のために使用されるものですが、その操作を実行するためには、さらに詳しい情報が必要な場合があります。追加の情報(確認を含む)が必要な命令であることは、ラベルの末尾に省略記号(...)を付けて示します。



この例では、[印刷...] コマンドをクリックすると、情報を入力するための [印刷] ダイアログ ボックスが表示されます。

操作の実行前に、さらに細かい選択ができるよう、場合によっては操作を完全に取り消すことができるようするために、省略記号を適切に使うことが大切です。省略記号によって与えられた視覚的な手掛かりにより、ユーザーは安心してソフトウェアを使用できます。

ただし、新しいウィンドウを表示する操作すべてで省略記号を使う必要はありません。操作を実行するのに追加の情報入力が必要な場合だけに使います。たとえば、[バージョン情報]、[詳細]、[ヘルプ]、[オプション]、[プロパティ]の各コマンドをクリックしたときには別のウィンドウを表示する必要がありますが、ユーザーからの追加情報を必要としません。したがって、省略記号は必要ありません。

省略記号を付けるかどうか判断に迷う場合(ラベルに動詞がないなど)は、ユーザーが何を目的としてこのコマンドを実行しようとするかを基準として判断します。想定される目的が単にウィンドウに表示される情報を見ることであれば、省略記号は不要です。

正しい例:

他の色...

バージョン情報

1つ目の例では、ユーザーは一般に色の選択を目的としてこのコマンドを実行するので、省略記号を付けます。2つ目の例では、多くの場合ユーザーはバージョン情報を見ようとしてこのコマンドを実行するので、省略記号は不要です。

注: メニュー コマンドに省略記号が必要かどうか判断する際、**権限の昇格**について考慮する必要はありません。コマンドの実行に必要な情報ではなく、実行権限を確認するためのものだからです。権限の昇格が必要である旨はセキュリティ シールドで示されます。

## ラベル

- センテンススタイルの大文字化を使用します。
  - 例外: 古いアプリケーションでは、大文字/小文字の形式が混在しないように、タイトルスタイルの大文字化を採用することもできます。

## メニュー カテゴリ名

- 1語の名詞または動詞を使用してメニュー カテゴリ名を作成します(英語の場合)。ラベルに複数の単語を含めると、単語の数だけラベルがあると思われる可能性があります。
- 動詞ベースのメニュー名を優先します。ただし、"作成する(Create)"、"表示する(Show、View)"、または"管理する(Manage)"の場合は、動詞を省略します。たとえば、以下のメニュー カテゴリには動詞が含まれていません。
  - 表
  - ツール
  - ウィンドウ
- 非標準のカテゴリ名については、メニューの内容を明確に正しく説明する具体的な1語を使用します。名前によってメニュー内の全項目を説明するほど汎用化する必要はありませんが、メニュー内容が表示された際に違和感のない程度には予測可能なものである必要があります。

## メニュー項目名

- 動詞、名詞、または名詞句で始まるメニュー項目名を使用します(英語の場合)。
- 動詞ベースのメニュー名を優先します。ただし、以下の場合は動詞を省略します。
  - 動詞が、"作成する(Create)"、"表示する(Show、View)"、または"管理する(Manage)"である。たとえば、以下のコマンドには動詞が含まれていません。
    - バージョン情報
    - 詳細設定
    - 全画面表示
    - 新規作成
    - オプション
    - プロパティ
- 繰り返しを避けるために、動詞をメニュー カテゴリ名と同じにする。たとえば、[挿入] メニュー カテゴリ内では、"テキストの挿入"、"表の挿入"、"画像の挿入"ではなく、[テキスト]、[表]、[画像] を使用します。
- 具体的な動詞を使用します。"変更する"、"管理する"など、汎用的でわかりにくい動詞を使用しないようにします。
- 単一のオブジェクトに適用されるコマンドには名詞の単数形を使用します(英語の場合)。複数のオブジェクトの場合は、複数形を使用します。
- よく似たコマンドを区別するには、必要に応じて修飾子を使用します。例: 上に行を挿入、下に行を挿入。
- 相互補完的なコマンドのペアの場合は、補完的であることがよくわかる名前を選択します。たとえば、[追加] と [削除]、[表示] と [非表示]、[挿入] と [削除] のようにします。
- メニュー項目名は、技術ではなく、ユーザーの目的と作業に基づいて選択します。

正しい例:



間違った例:



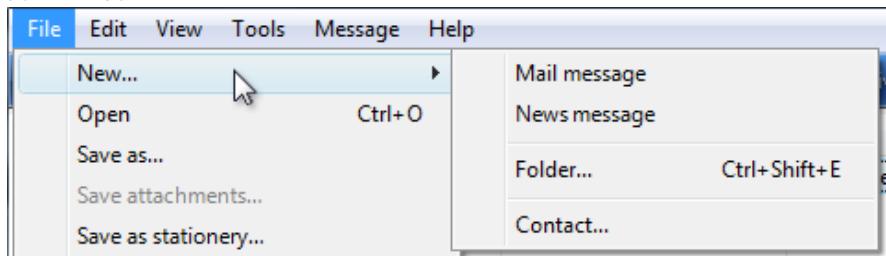
間違った例では、メニュー項目の名前が技術ベースのものになっています。

- メニュー項目名は、次に示す目的に沿って使い分けます。
  - オプション: プログラムオプションを表示します。
  - カスタマイズ: 特に機械的なUI構成に関するプログラムオプションを表示します。
  - 個人設定: 一般的に使用される個人設定の概要を表示します。
  - 基本設定: 使用しません。代わりに[オプション]を使用します。
  - プロパティ: オブジェクトのプロパティウィンドウを表示します。
  - 設定: メニュー ラベルには使用しません。代わりに[オプション]を使用します。

サブメニュー名

- サブメニューを表示するメニュー項目のラベルには、省略記号を含めません。サブメニューの矢印は、別の選択が必要であることを示しています。

間違った例:



この例の[新規作成]メニュー項目には、誤って省略記号が含まれています。

## ドキュメント

メニューに言及するときは、以下のこと留意します。

- メニューを表示または非表示にするコマンドでは、"メニュー バー"とします。"クラシック メニュー"は使用しないでください。
- メニューにはラベルを使用します。大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、アクセスキーを示すかっこや下線付き文字、および省略記号は含めません。
- メニュー カテゴリには、"<カテゴリ名> メニューの"を使用します。メニュー項目の場所がコンテキストから明らかなる場合、メニュー カテゴリに言及する必要はありません。
- メニュー項目のユーザー操作を説明する場合は、"クリック"を使用し、"メニュー"または"コマンド"は追加しません。"選択(choose、select、pick)"は使用しません。技術文書以外では、メニュー項目を"メニュー項目"としません。
- メニュー オプションのチェックマークを外すことは、"クリックしてチェックマークをオフにします"という表現にします。"クリア"は使用しません。
- コンテキストメニューは"コンテキストメニュー"とし、"ショートカットメニュー"は使用しません。
- プログラミング文書以外では、メニューの説明に"カスケード"、"プルダウン"、"ドロップダウン"、または"ポップアップ"を使用しません。
- 使用できないメニュー項目は"利用不可の"と表現し、"薄く表示された"、"無効になった"、"灰色表示の"とはしません。"無効"は、プログラミング文書で使用します。
- ラベルは半角の角かっこ([ ])で囲み、可能な場合は太字にします。

例:

- ドキュメントを印刷するには、[ファイル] メニューの [印刷] をクリックします。
- [表示] メニューの [ツールバー] をポイントし、[書式設定] をクリックします。

# メニューのデザイン コンセプト

## メニュー

### コマンドの提示方法

メニューを効果的に使用するには、さまざまなコマンドの提示方法の特徴を理解することが効果的です。

- メニュー バー。メニュー バーは、プログラム内で利用可能なすべての最上位コマンドをカタログのように一覧表示するのに適しています。プログラムを初めて使用するユーザーの場合は、メニュー バーのすべてのコマンドを確認し、以下のような事項を調べようとします。
  - プログラムで実行できること。
  - プログラムに用意されているコマンド。
  - よく使用するコマンドのショートカットキー。

効果的なメニュー バーとは、包括的で適切に構成され、見ればすぐわかるものです。効率的であることが望ましいですが、決定的な事項ではありません(実現できないこともあります)。メニュー バーは、特に新しいユーザーにとって、何よりも学習と発見のためのツールであると考えてください。

- ツール バー。ツール バーは、頻繁に使用する即時型のコマンドに、迅速かつ便利にアクセスするのに最適です。包括的であったり、処理内容が理解できるものである必要はなく、直接的で効率的であることが重要です。
- コマンド ボタン。コマンド ボタンとは、少数の主コマンドを公開するためのシンプルな視覚的、直接的手段です。ただし、拡張性に乏しいため、数個以上のコマンドを表示するにはメイン ウィンドウのメニューを使用する必要があります。
- コンテキスト メニュー。コンテキスト メニューとは、シンプルで直接的な、コンテキストに依存するものです。現在のコンテキストに適用されるコマンドとオプションのみが表示されるため、効率のよさも備わっています。コンテキスト メニューはポインターの現在の場所に表示されるため、メニューを表示するためにマウスを動かす必要がありません。ただし、通常は画面上に表示されていません。コンテキスト メニューは、上級ユーザーに、コンテキスト依存コマンドとオプションを附加的に提示する場合に限り有効であると考えてください。

以上のようなさまざまなトレードオフがあるため、プログラムではコマンドの提示方法をさまざまに組み合わせて使用する必要が生じることがあります。たとえば、フル機能型のアプリケーションではメニュー バー、タスク バー、およびコンテキスト メニューを使用することが多いのに対して、シンプルなプログラムでは一般に、コマンド ボタンと標準コンテキスト メニューのみを使用します。

#### 最も重要な点

プログラムの種類、ウィンドウの種類、コマンドの使用法、およびターゲットとするユーザーに適したコマンドの提示方法を選択します。

### 効果的なメニュー バー

メニュー バーはこれまで最もよく使用してきたメニューの種類ですが、すべての種類のプログラムやウィンドウに適しているわけではありません。次に、効果的に使用するための要素をいくつか示します。

#### シンプルであることの維持

メニュー バーは、ダイアログ ボックスには使用しません。ユーティリティのようなシンプルなプログラムに使用することも避ける必要があります。このようなウィンドウのコマンドは、シンプルで直接的、しかもわかりやすい外観を維持する必要があります。メニュー バーは本質的に学習と発見のためのツールであり、シンプルなウィンドウにはそうした要素は必要ありません。ダイアログ ボックスの場合は、[コマンド ボタン](#)(メニュー ボタン、分割ボタンを含む)、[コマンド リンク](#)、およびコンテキスト メニューを使用します。

#### 画面領域の効率的な利用

メニュー バーは画面領域を効率的に使用できますが、処理が重くなるため、それほど多数のコマンドが存在しない場合や頻繁に使用されない場合は、代替の方法を検討する必要があります。たとえば、ブ

ログラムに既にツールバーが存在し、少數のドロップダウンメニューのみが必要な場合は、ツールバーメニューを選択する方がよいといえます。

### 一貫性のあるメニューにする

学習や発見の必要性から、ユーザーはメニュー バーには一貫性があるものと考えています。つまり、前回メニューを使用したときと同じメニュー項目が表示されることが期待されています。現在のコンテキストに応じてメニュー項目を有効にしたり無効にすることはできますが、メニュー項目やサブメニューを追加/削除すべきではありません。ただし、ドキュメントの読み込み中など、プログラムの状態の明らかな変化に基づいて、メニュー カテゴリ全体を追加/削除することはできます。

しかし、メニュー項目が無効になっていると、ユーザーにはその理由を確認する必要が生じ、混乱をまねくことがあります。理由がはっきりしない場合、試行錯誤と論理的な推測によってユーザーが問題を明らかにしなければならなくなります。このような場合は、項目を有効にしたままで、適切なエラーメッセージを表示し、問題を明確に説明する方が効果的です。

### メニュー バーの構成

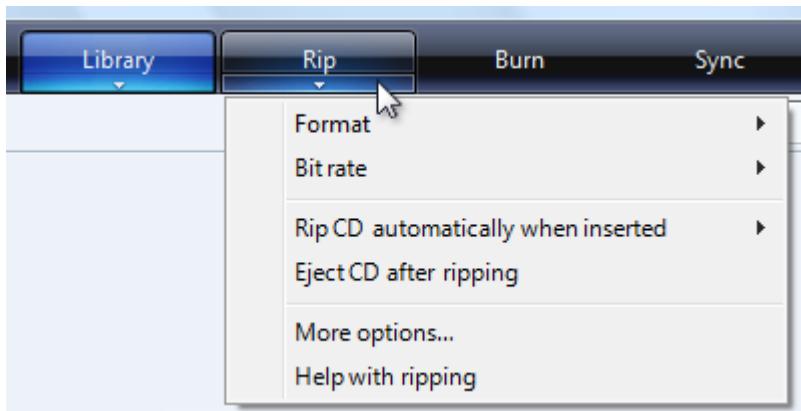
メニュー バーでは、メニュー項目がツリー構造で整理されています。ただし、ツリーを使用することにはジレンマがあります。ツリーは、メニュー項目を整理し、発見しやすくするためのものですが、メニュー ツリー内のすべてのメニュー項目を発見することは難しくなります。あまりよく知られていなかつたり、複数のメニュー カテゴリに属することができるメニュー項目の場合は特に、見つけにくくなります。たとえば、[デバッグ] と [ウィンドウ] というカテゴリを持つメニューがあるとします。デバッグ ウィンドウ コマンドを探すときに、どちらのカテゴリを使用すればよいのかは簡単に判断できません。

標準メニュー カテゴリを使用すると、このジレンマに対処することができます。たとえば、ユーザーが [終了] コマンドを探す場合は、標準の配置場所を想定して [ファイル] メニューを確認します。標準的ではないメニュー項目の場合で、複数のカテゴリに属することができるために探し出すのに手間がかかると考えられる場合は、複数のカテゴリに、発見しにくいメニュー項目を 1、2 個だけ配置しておきます。これ以上になると、メニュー バー全体の操作性が損なわれます。

### 標準メニュー構成

標準メニュー構成が使用されていると、よく使用するメニュー項目が予測可能になり、発見しやすくなります。ただし、このようなカテゴリは、大部分のアプリケーションがドキュメント ファイルの作成や表示のために使用されていたときのものであり、[ファイル]、[編集]、[表示]、[ツール]、[ヘルプ] メニュー カテゴリで構成されています。この標準構成は、Windows エクスプローラーのような他の種類のプログラムにはほとんど価値がありません。

プログラムに適切ではない場合は、標準メニュー構成を使用する必要はありません。メニュー項目を、プログラムの目的とユーザーのタスクや目標に応じた、有用で自然なカテゴリに整理することを検討します。



この例の Windows Media® Player では、主要なタスクに合わせて非標準のメニューを使用しています。

非標準のメニュー カテゴリの使用を選択する場合は、適切なカテゴリ名をデザインする必要があります。これらの名前には具体的な 1 語を使用し、正確にその内容を説明する必要があります。カテゴリ名によってメニュー内の全項目を説明するほど汎用化させる必要はありませんが、メニュー内容が表示さ

れた際に違和感のない程度には予測可能なものである必要があります。

ただし、プログラムが主にドキュメントの作成と表示に使用される場合はほぼ確実に、標準メニュー構成を使用する必要があります。組み込みの Windows アプリケーションの多くでは標準メニューが使用されなくなったことから、標準メニューが廃止されたとは判断しないでください。廃止されたのではなく、ドキュメント作成を主な目的としないプログラムには、それ以上に適切な方法が存在するということです。

## メニュー バーとツール バーの比較

多くのプログラムには、メニュー バーとツール バーの両方が用意されています。優れたメニュー バーと優れたツール バーに求められる属性は異なるため、メニュー バーのコマンドとツール バーのコマンドとの間に正確な対応関係が存在する必要はありません。

優れたメニュー バーは、利用可能なすべての最上位コマンドをカタログのように一覧表示できるものであり、優れたツール バーは、頻繁に使用するコマンドに簡単かつ便利にアクセスできます。ツール バーの目的はユーザーの生産性を高めることであって、ユーザーを訓練することではありません。ツール バーからコマンドにアクセスする方法を知ると、そのコマンドをメニュー バーから使用することはほとんどなくなります。

それぞれのメニューのメリットを存分に発揮することに重点を置いてください。メニューの種類間に一貫性を持たせることについては深刻に考えなくてもかまいません。

詳細については、「[ツール バーのデザイン コンセプト](#)」を参照してください。

## メニュー バーとコンテキスト メニューの比較

コンテキスト メニューはコンテキストに依存するため、以下の点でメニュー バーと異なっています。

- コンテキスト メニューには、現在のコンテキストにのみ適用される項目が表示されるため、一般に、無効にされている項目がありません。メニュー バーには機能の完全なカタログとして以上の働きがあるため、包括的で一貫性がある必要があります。
- コンテキスト メニューはショートカット メニューとも呼ばれますが、メニュー バーのように学習ツールとして作成されたものではないため、ショートカット キーは表示されません。シンプルで効率的であることが重要です。
- コンテキスト メニューには特定の順序があり、最もよく使用される項目が先頭に(主コマンド)、転送コマンドがその次に、プロパティが最後に表示されます。この順序は効率と予測可能性を考慮しています。一方、メニュー バーのメニューは、コマンド間の関連性と使用頻度に基づいて並べられています。

## メニューのアフォーダンス

メニュー バーにはアフォーダンスがありません。つまり、視覚的特性によって使用方法が示唆されません。メニュー バーでは、ユーザーはエクスペリエンスによってメニュー バーを理解し、標準的な外観と位置によって認識します。

他のメニュー パターンも同じく視覚的特性によって認識できないうえに、標準的な位置がないため、[ドロップダウン矢印](#)を使用してプルダウン メニューの存在を示します。この矢印が必要であるかどうかが、コマンドの提示方法を選択する際の要素になることもあります。プログラムのウィンドウにメニュー 矢印が散乱している場合は、メニュー バーの使用を検討します。

## メニューでのアイコンの使用

Windows® ではテーマが付けられたメニューを使用すると、メニュー項目のアイコンを表示することができます。アイコンの表示には次のメリットがあります。

- 最もよく使用するメニュー項目を強調することができる。
- 目立つアイコンを使用すると、ユーザーは使用頻度の高いメニュー項目をすばやく見分けることができる。
- 工夫されたデザインのアイコンを使用すると、そのメニュー項目の意味を伝えることができる。

メニュー アイコンの使用を決定しても、すべてのメニュー項目にアイコンを使用する必要はありません。実際、最も重要なメニュー項目にのみアイコンを使用すると、アイコンの効果が大きくなります。

コマンドはアクション(動詞)であるのに、アイコンによってオブジェクト(名詞)を表示するものであるため、コマンドアイコンについては特にデザインが難しくなります。このため、ほとんどのコマンドアイコンには標準的な記号またはアクションを示唆するオブジェクトの画像が使用されます。

## アクセシビリティ

メニュー項目には、アクセスキーとショートカットキーを使用して直接アクセスできるようにする必要があります。そうすれば、作業スピードの速いパワーユーザーなど、キーボードの使用を好むユーザーにとって便利になります。

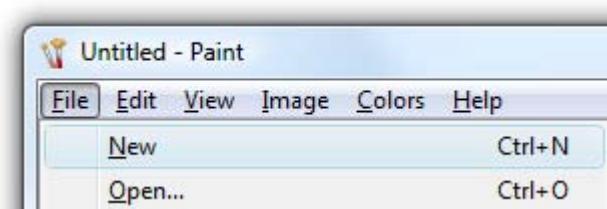
アクセスキーとショートカットキーには次に示すように、メニューに関して根本的な違いがいくつかあります。

アクセスキーとは次のようなものです。

- メニュー名に示された下線付きの文字。
- Altキーを押しながら英数字キーを押す。
- 主としてアクセシビリティを目的としている。
- すべてのメニューに割り当てられている。
- 記憶して使用するものではなく、UI内に直接示されている(該当文字に下線が付けられています)。
- メニュー内で一貫性を持たせて割り当てられていない(常に存在するわけではないため)。

一方、ショートカットキーとは次のようなものです。

- Ctrlキーを押しながら一連のファンクションキーを押す。
- 主として上級ユーザーがすばやくアクセスできるようにすることを目的としている。
- 最も使用頻度の高いメニュー項目にのみ割り当てられている。
- 記憶して使用するものあり、メニュー、ツールヒント、ヘルプ内にのみ記されている。
- プログラム内で一貫性を持たせて割り当てる必要がある(記憶して使用するものあり、直接UI内には記さないため)。



この例では、メニューにアクセスキーもショートカットキーも割り当てられています。

ヒント: アクセスキーの下線は、通常、既定で非表示にされており、Altキーを押すと表示されます。プログラムで割り当てられているアクセスキーを認識しやすくするために、アクセスキーを常に表示するように設定できます。コントロールパネルからコンピューターの簡単操作センターを開き、[キー ボードを使いやすくします]をクリックし、[ショートカットキーとアクセスキーに下線を表示します]チェックボックスをオンにします。

## ツールバー

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

提示方法

コントロールとコマンド

構成と順序

メニュー バーの非表示化

対話操作

アイコン

標準メニューと分割ボタン

パレット ウィンドウ

カスタマイズ

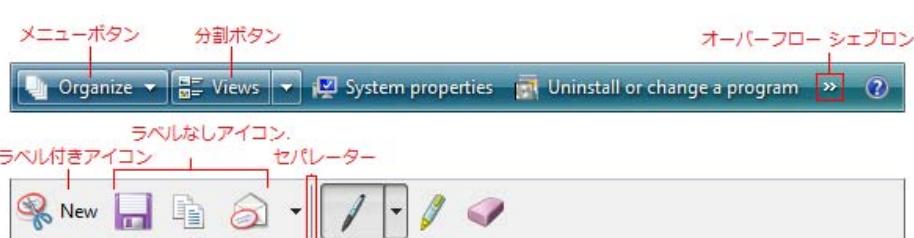
省略記号の使用

推奨されるサイズと間隔

ラベル

ドキュメント

"ツールバー" は、コマンドをグラフィカルに提示するもので、効率的にアクセスできるように最適化されています。



典型的なツールバーです。

ツールバーは、メニュー バーと併用するか、またはメニュー バーに代えて使用します。ツールバーは、操作が直接的で(マウス クリックによって表示されるのではなく、常に表示されている)、すぐに効果があり(追加の入力を必要としない)、最もよく使用されるコマンドが含まれる(包括的な一覧ではない)ため、メニュー バーよりも効率的です。メニュー バーとは異なり、コマンドを包括的に提供する必要やコマンドの内容がひとめでわかるようにする必要はありませんが、すばやく、便利で効率的に使用できるようにする必要があります。

ツールバーには、ユーザーがカスタマイズして追加/削除できるものや、サイズや場所、内容を変更できるものがあります。種類によっては、ドッキングを解除してパレット ウィンドウにすることができるものもあります。ツールバーの種類の詳細については、このトピックの「[使用パターン](#)」を参照してください。

注: メニュー、コマンド ボタン、アイコンに関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- ・ ウィンドウがメイン ウィンドウかどうか。ツールバーは、[メイン ウィンドウ](#)では効率的に機能しますが、[サブ ウィンドウ](#)では多くの場合邪魔になります。サブ ウィンドウの場合は、ツールバーではなく、[コマンド ボタン](#)、[メニュー ボタン](#)、[リンク](#)を使用してください。
- ・ 頻繁に使われるコマンドの数が少ないかどうか。ツールバーでは、メニュー バーのように多くのコマンドを扱うことができないため、頻繁に使われるコマンドの数が少ない場合にアクセスを効率化する手段として用いるのが最適です。
- ・ すぐに効果のあるコマンドがほとんどかどうか。つまり、追加の入力を必要としないコマンドがほとんどかどうか。効率性の観点から、ツールバーには直接的、即時的であることが求められます。該当しない場合は、メニュー バーの方が適しています。
- ・ コマンドのほとんどを直接提示できるかどうか。つまり、ユーザーが1回のクリックでコマンドを操作できるかどうか。メニュー ボタンを使っていくつかのコマンドを提示することができますが、コマンドの多くをこの方法で提示するとツールバーの効率性が損なわれます。この場合はメニュー バーの作成を検討します。
- ・ コマンドをアイコンでうまく表現することができるかどうか。メニュー コマンドはテキストを使って提示しますが、ツールバー コマンドは(テキスト ラベルを併用する場合もありますが)主にアイコンを使って提示します。コマンド アイコンの品質が十分でなく、ひとめで内容がわかるものではない場合は、メニュー バーの作成を検討します。

ツールバーがあつてメニュー バーがないプログラムで、ほとんどのコマンドにメニュー ボタンや[分割 ボタン](#)から間接的にアクセス可能である場合、このツールバーは、実質的にメニュー バーの役割を果たしています。この場合は、メニュー ガイドラインの[ツールバー メニュー](#)のパターンに従ってください。

### デザイン コンセプト

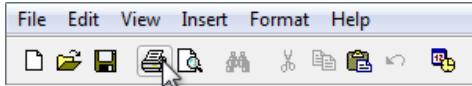
優れたメニュー バーは、利用可能なすべての最上位コマンドをカタログのように一覧表示できるものであり、優れたツールバーは、頻繁に使用するコマンドに簡単かつ便利にアクセスできます。ツールバーの目的はユーザーの生産性を高めることであつて、ユーザーを訓練することではありません。ツールバーからコマンドにアクセスする方法を知ると、そのコマンドをメニュー バーから使用することは

ほとんどなくなります。したがって、プログラムのメニュー バーとツール バーを直接対応させる必要はありません。

#### ツール バーとメニュー バー

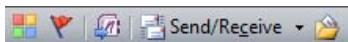
一般的に、ツール バーは以下の点においてメニュー バーと異なります。

- 使用頻度。メニュー バーはプログラムで利用可能なコマンドをすべて提供するものですが、ツール バーは最もよく使われるコマンドのみを提供するものです。
- 即時性。メニュー コマンドには追加の入力が必要なものがありますが、ツール バー コマンドはクリックしてすぐに効果があります。たとえば、メニュー バーの [印刷] コマンドでは最初に [印刷] ダイアログ ボックスが表示されますが、ツール バーの [印刷] ボタンはクリックするとすぐにドキュメントのコピーを既定のプリンターで 1 部印刷します。



この例では、ツール バーの [印刷] ボタンをクリックすると、すぐにドキュメントのコピーが既定のプリンターから 1 部印刷されます。

- 直接性。メニュー バー コマンドを呼び出すにはメニュー 内を移動する必要がありますが、ツール バー コマンドは 1 回のクリックで呼び出することができます。
- コマンドの数と密度。ツール バーでは、コマンドの数に比例して必要な画面領域は大きくなり、コマンドを使用しない場合でもそのまま領域を使用します。そのため、ツール バーでは領域を効率的に使用するようにする必要があります。これに対し、メニュー バー コマンドは通常は非表示になっており、階層構造によってコマンドをいくつでも追加できます。
- サイズと表示。メニュー バーはテキスト形式のコマンドを使用しますが（オプションでアイコンを付けることもできます）、ツール バーのコマンドは多くのコマンドを小さな領域にまとめて表示するためにアイコンを使用します（ツールヒント形式のラベルを付けます）。ツール バー ボタンに標準的なテキスト ラベルを追加することもできますが、かなりの領域が必要となります。



ラベル付きのツール バー ボタンは、ラベルなしのツール バー ボタンに比べて 3 倍以上の領域を使用します。

- わかりやすさ。ツールヒントだけで効率的にコマンドを探すことは難しいため、デザインの優れたツール バーにするには、コマンドの内容がひとめでわかるアイコンが必要です。ただし、あまり頻繁に使われないコマンドのアイコンが多少わかりにくいものであっても、問題ありません。

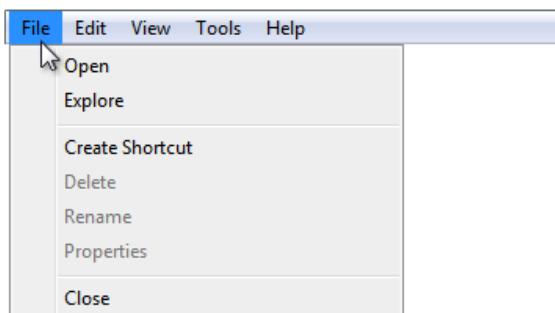


この例では、最もよく使用されるアイコンがどのような内容のコマンドであるかが、ひとめでわかるようになっています。

- 認識と区別のしやすさ。頻繁に使用するツール バー コマンドの場合、ユーザーはボタンの場所、形、色などの属性を記憶しています。ツール バーのデザインが優れていると、アイコンを正確に思い出すことができなくても、すばやくコマンドを見つけることができます。それに対して、頻繁に使用するメニュー バー コマンドの場合、ユーザーは場所は記憶していますが、コマンド ラベルを頼りにして選択を行います。



ツール バー コマンドでは、固有の場所、形、色によって、アイコンの認識と区別がしやすくなります。



メニュー バー コマンドの場合、結局はラベルに頼ることになります。

#### 効率性

ツールバーはその特色を活かすため、主に効率性を意識してデザインする必要があります。効率的でなければツールバーの意味がありません。

#### 最も重要な点

ツールバーは、主に効率性を意識してデザインします。ツールバーには、使用頻度が高く、すぐに効果があり、直接的で、すばやく認識できるコマンドを表示します。

#### メニュー バーの非表示化

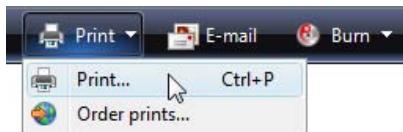
通常、ツールバーは、メニュー バーと組み合わせることによって最大限の効果が得られます。優れたツールバーは効率的であり、優れたメニュー バーは包括的です。メニュー バーとツールバーの両方を提供することで、競合することなくそれぞれの長所を活用できます。

しかし、シンプルなプログラムではこのことは当てはまりません。コマンドの数が少ないプログラムの場合、メニュー バーが、冗長で効率的なツールバーのようなものとなるため、メニュー バーとツールバーを両方付けることにはあまり意味がありません。

このため、Windows Vista® ではシンプルなプログラムの多くは、主にツールバーにコマンドが配置され、メニュー バーは既定で非表示になっています。Windows エクスプローラー、Windows® Internet Explorer®、Windows Media® Player、Windows フォト ギャラリーなどがこれに当たります。

これは大きな変更です。メニュー バーを削除すると、ツールバーにコマンドを包括的に配置する必要があるほか、以下のような変更を行わなければならないため、ツールバーの性質が根本的に変わります。

- 使用頻度。メニュー バーを削除すると、使用頻度にかかわらず、ウィンドウやコンテキスト メニューから直接利用できないすべてのコマンドを、ツールバーからアクセスできるようにする必要があります。
- 即時性。メニュー バーを削除すると、ツールバーが目に見える唯一のコマンドへのアクセス ポイントになります。このため、メニュー バーのすべての機能をツールバーが備えている必要があります。たとえば、メニュー バーがない場合、ツールバーの [印刷] コマンドは、すぐに印刷を行うのではなく、[印刷] ダイアログ ボックスが表示されるようにする必要があります(ただし、この場合は分割ボタンを使用するとうまく解決できます)。標準的な [印刷] 分割ボタンについては、「[標準メニューと分割ボタン](#)」を参照してください。



この例では、Windows フォト ギャラリーのツールバーの [印刷] ボタンに、[印刷] ダイアログ ボックスを表示する [印刷] コマンドが設定されています。

- 直接性。領域を節約して煩雑さを避けるために、使用頻度の少ないコマンドは、直接的ではなくなりますが、メニュー ボタンに移動することもできます。

メニュー バーを補完するものとして使用するツールバーと、メニュー バーを削除または非表示にして使用するツールバーとではデザインが異なります。また、ユーザーがあるコマンドを実行するのに非表示になっているメニュー バーを表示させる前提でデザインを決めることはできないため、メニュー バーを非表示にすることとメニュー バーを完全に削除することは同じであると考える必要があります(既定でメニュー バーを非表示にすることとメニュー バーを表示してコマンドを探すこととがどちらも同じ扱いになります)。

メニュー バーなしで使用できるツールバーをデザインすると、多くの場合、妥協しなければならない点がいくつか出てきます。しかし、効率性についてあまり妥協しすぎてもいけません。メニュー バーを非表示にすることによってツールバーの効率性が損なわれる場合は、メニュー バーは非表示にしません。

#### キーボード アクセシビリティ

ツールバーとメニュー バーでは、キーボードを使ったアクセス方法がまったく異なります。メニュー バーの入力フォーカスを取得するには Alt キーを押し、解除するには Esc キーを押します。メニュー バーに入力フォーカスを取得すると、上下左右の方向キー、Home キー、End キー、Tab キーを使って、メニュー バー内でフォーカスを移動できます。これに対し、ツールバーの入力フォーカスは、Tab キーを使って、ウィンドウ内の要素全体で切り替えて取得します。ツールバーのタブオーダーは最後なので、Shift + Tab キーで逆に切り替えられることを知らない場合、要素が多いページでツールバーをアクティブにするのに苦労する可能性があります。

ここに、アクセシビリティに関するジレンマがあります。つまり、ツールバーは、マウス ユーザーにとって使いやすいものの、キーボード ユーザーにとっては使いにくいということです。メニュー バーとツールバーが両方ある場合は問題ありませんが、メニュー バーが削除されたり、非表示になっている場合は問題となります。

アクセシビリティの観点から、ツールバーをメインに考える場合でもメニュー バーは完全に削除せずには保持することをお勧めします。メニュー バーを削除するか、単に非表示とするかを選択する必要がある場合は、非表示にすることを選択してください。

#### 使用パターン

ツールバーにはいくつかの使用パターンがあります。

**メインツールバー** メインツールバーは効率性と包括性のニーズを調整する必要があります。シンプルなプログラムで最も効果を発揮します。

メニューが非表示になっているか削除されている場合に、メニューバーなしでも使用できるようにデザインされたツールバーです。



**補助ツールバー** 補助ツールバーでは、妥協することなく効率性を追求することができます。

メニューと組み合わせて利用するにデザインされたツールバーです。



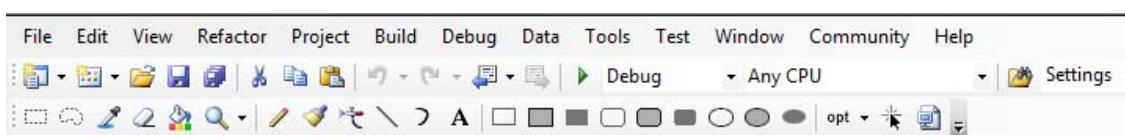
**ツールバーメニュー** ツールバーメニューは、主にメニューボタンおよび分割ボタンのコマンドで構成されるツールバーで、少数の直接コマンドが含まれる場合もあります。



*Windows Photo Gallery のツールバー メニュー*

**カスタマイズ可能なツールバー** カスタマイズ可能なツールバーには、ユーザーがツールバーを追加/削除できるものや、サイズや場所、内容を変更できるものがあります。

ユーザーによるカスタマイズが可能なツールバーです。



*Microsoft Visual Studio® のカスタマイズ可能なツールバー*

**パレットウィンドウ** パレットウィンドウは、ドッキングされていないツールバーです。

ドウ

コマンドの配列を提示するモードレスダイアログボックスです。



*Windows Paint のパレット ウィンドウ*

ツールバーには、以下のようなスタイルがあります。

**ラベルなしのアイコン** ラベル付きのボタンの数が多くなる場合、または頻繁に使用されるプログラムの場合は、このスタイルを使用します。このスタイルを使うと、複雑な機能を持つプログラムで複数行のツールバーを利用できるようになります。このスタイルは、カスタマイズ可能である必要のある唯一のスタイルになります。このスタイルでは、使用頻度の高いコマンドボタンにラベルを付けることもできます。



*WordPad のラベルなしのアイコンツールバー*

**ラベルなしの大好きなアイコン** このスタイルは、シンプルに認識できるアイコンを持ち、通常、小さなウィンドウで実行されるシンプルなユーティリティに使用します。





Windows Live Messenger と Windows Snipping Tool のラベルなしの大きなアイコンツールバー

ラベル付きアイコン  
このスタイルは、コマンドの数が少なく、あまり頻繁に使用されないプログラムの場合に使用します。  
1行のラベル付きの小さなアイコンです。  
このスタイルでは、常に1行のみとなります。



Windows エクスプローラーのラベル付きアイコンツールバー

部分ツールバー  
完全なツールバー  
このスタイルは、移動ボタン、検索ボックス、またはタブを含むウィンドウに使用して、ウィンドウの上部が重くならないようにします。  
バーが必要でない場合に、領域を節約するために使用される小さなアイコンの行の一部分です。



部分ツールバーは、移動ボタン、検索ボックス、またはタブと組み合わせることができます。

大きな部分ツールバー  
完全なツールバー  
このスタイルは、移動ボタンまたは検索ボックスを含むシンプルなユーティリティに使用して、ウィンドウの上部が重くならないようにします。



Windows Defender の大きな部分ツールバー

大きな部分ツールバー

完全なツールバー

このスタイルは、移動ボタン、検索ボックス、またはタブと組み合わせることができます。

大きな部分ツールバー

完全なツールバー

このスタイルは、移動ボタンまたは検索ボックスを含むシンプルなユーティリティに使用して、ウィンドウの上部が重くならないようにします。

モードアイコンボタン

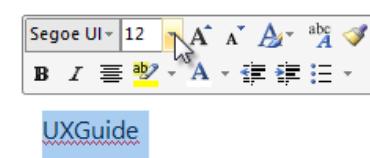
モードボタンをクリックすると、選択したモードが入力されます。



Windows Paint のモードボタンの例。

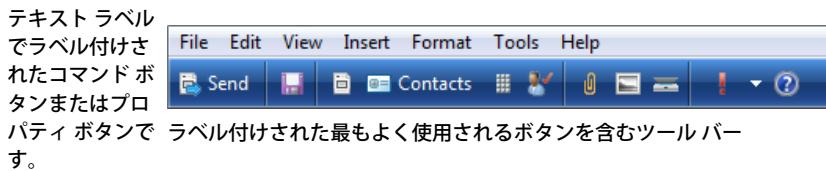
プロパティアイコンボタン

プロパティアイコンボタンは、現在選択されたオブジェクトがある場合、その状態が反映されます。ボタンをクリックすると、選択されたオブジェクトに変更が適用されます。



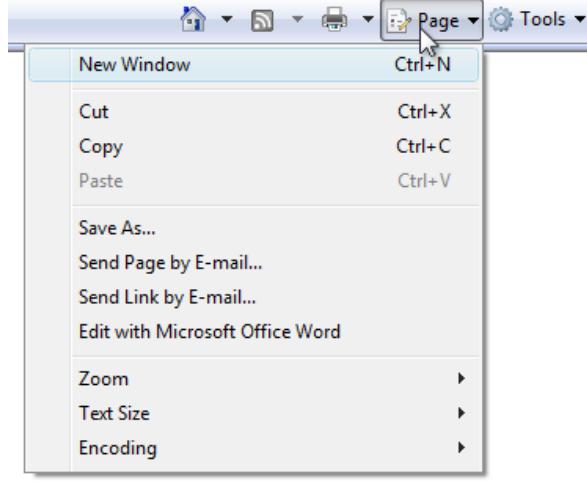
Microsoft Word のプロパティボタンの例。

ラベル付きアイコン ボタン これらのボタンは、アイコンが内容を十分表現できていない、使用頻度の高いツールバー ボタンに使用されます。また、ボタン数が少ないために、各ボタンにテキストラベルを付けることのできるツールバーにも使用されます。



テキストラベルでラベル付けされたコマンドボタンまたはプロパティボタンでラベル付けされた最もよく使用されるボタンを含むツールバーです。

メニュー ボタン 下向きの三角形は、クリックするとメニューが表示されることを表します。

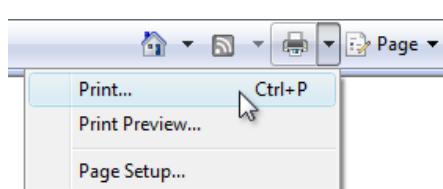


関連するいくつかのコマンドをまとめたメニュー ボタン



通常の分割ボタン

右端にある下向きの三角形をクリックすると、メニュー ボタンと同様にメニューが表示されることを表します。



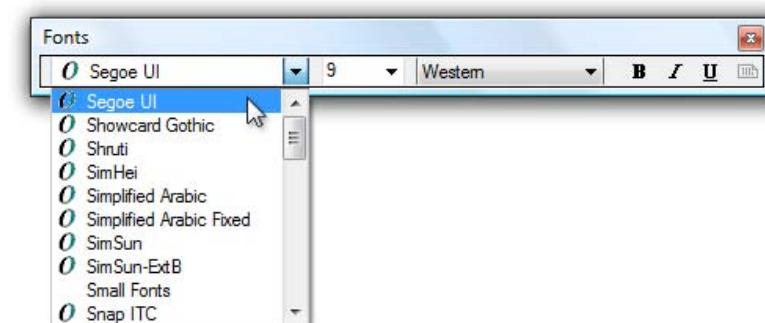
ドロップダウンされた状態の分割ボタン

この例では、印刷関連のコマンドを1つにまとめるために使われています。即時型の [印刷] コマンドが使用されることがほとんどなので、通常、他のコマンドは表示する必要はありません。

メニュー ボタンと違って、ボタンの左側部分をクリックすると、ラベルに示された操作が直ちに実行されます。分割ボタンは、次のコマンドが、直前のコマンドと同じであるような状況にも向いています。この場合、ラベルは直前のコマンドに応じて変わります。カラー ピッカーはその典型的な例です。



この例では、直前のコマンドに応じてラベルが変わっています。



この例では、フォント属性を表示/設定するためにドロップダウンリストが使用されています。

ツールバーのドロップダウンリストは、現在選択されたオブジェクトがある場合、その状態が反映されます。リストを変更すると、選択されたオブジェクトの状態が変更されます。

## ガイドライン

### 提示方法

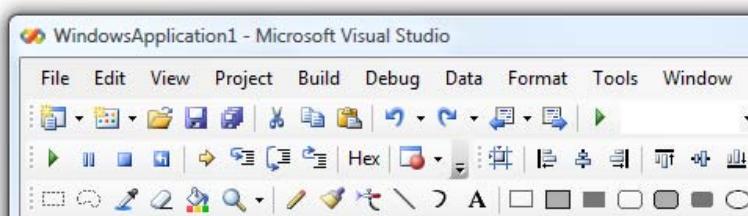
- コマンドの数と使用方法に基づいて適切なツールバーのスタイルを選択します。選択方法に関するガイダンスは、上記の[ツールバーのスタイル](#)の表を参照してください。プログラムの作業領域を大幅に狭めてしまうツールバー構成は避けてください。
- ツールバーはコンテンツ領域の上に配置します。メニュー バーやアドレスバーがある場合はその下に配置します。
- 領域を節約することを優先する場合は、以下に従います。
  - 一般的によく知られているアイコンやあまり使われないコマンドのラベルを省略します。
  - ウインドウ幅全体を使わず、部分ツールバーのみを使用します。
  - メニュー ボタンまたは分割ボタンを使って、関連するコマンドをまとめます。
  - オーバーフローセブロンを使用して、あまり使用されないコマンドを非表示にします。
  - 現在のコンテキストに該当するコマンドのみを表示します。



Windows Internet Explorer のツールバーでは、一般的なアイコンのラベルを省略して部分ツールバーを使用し、あまり使用されないコマンドをオーバーフローセブロンを使用して隠すことで、領域を節約しています。

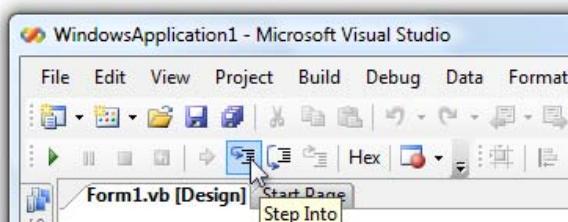
- ラベルなしのアイコンツールバー形式の場合は、既定のツールバー構成を 2 行以内にします。3 行以上のツールバーが便利な場合は、[カスタマイズ可能](#)なツールバーにします。3 行以上のツールバーはユーザーの作業を妨げ、プログラムの作業領域を大幅に狭める可能性があります。

間違った例:



既定で 3 行以上のツールバー構成は、かなり煩雑な外観になります。

- 現在のコンテキストに該当しないツールバー ボタンは、個々に無効にします。削除せず、無効にすることで、ツールバーの内容が一定に保たれ、探しやすくなります。
- クリックすると直接エラーが発生するツールバー ボタンは、個々に無効にします。これは、[直接性](#)を維持するために必要です。
- ラベルなしのアイコンツールバー形式の場合は、現在のコンテキストに該当しないツールバーをすべて削除します。該当するモードでのみ表示されるようにします。



この例では、プログラムの実行中のみ [デバッグ] ツールバーが表示されます。

- ツールバー ボタンは左揃えで表示します。[ヘルプ] アイコンがある場合は、右揃えに配置します。



すべてのツールバー ボタンは、[ヘルプ] アイコンを除き左揃えで表示します。

- ツールバー ボタンラベルが動的に変化するようにしないでください。これは予想外の動作であり、混乱を招きます。ただし、現在の状態が反映されるようにアイコンを変化させることは可能です。



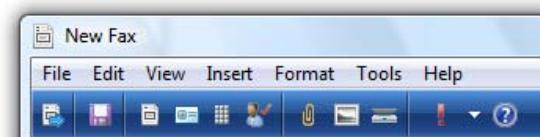
この例では、既定のコマンドであることを示すためにアイコンが変わっています。

### コントロールとコマンド

- よく使用されるコマンドを優先します。
  - メインツールバーでは、包括的なコマンドを表示します。メインツールバーは、メニュー バーほど包括的である必要はありませんが、他ではすぐに見つからないコマンドをすべて提示する必要があります。メインツールバーには、以下のコマンドを提示する必要はありません。
    - UI 自身に直接表示されているコマンド
    - 通常、コンテキストメニューからアクセスするコマンド
    - [切り取り]、[コピー]、[貼り付け]などの一般的なコマンド

- 補助ツールバーには、使用頻度の最も高いコマンドを提示します。ツールバー コマンドの上位機能としてメニュー バー コマンドを利用できるため、すべてのコマンドを提供する必要はありません。コマンドにすばやく便利にアクセスすることにのみ着目し、その他の要素は無視します。
- 直接コントロールを優先します。ツールバー ボタンは、次の優先順位で使用します。
  - アイコン ボタン。直接コントロールであり、最小限の領域で済みます。
  - ラベル付きアイコン ボタン。直接コントロールですが、アイコン ボタンよりも広い領域が必要です。
  - 分割ボタン。最も一般的なコマンドについては直接コントロールですが、派生コマンドを取り扱います。
  - メニュー ボタン。間接コントロールですが、多くのコマンドを提示します。
- 即時型のコマンドを優先します。即時型のコマンドと、柔軟性を持たせるために追加入力できるコマンドのいずれかを選択できる場合、次のこととに留意します。
  - メイン ツールバーの場合、柔軟性の高いコマンド ([印刷...] など) を使用します。
  - 補助ツールバーの場合、ツールバーでは即時型のコマンド ([印刷] など) を、メニュー バーでは柔軟性の高いコマンド ([印刷...] など) を使用します。
- 使用頻度の高いコマンドにはラベルを表示します(特に、あまり一般的ではないアイコンの場合)。

許容される例:



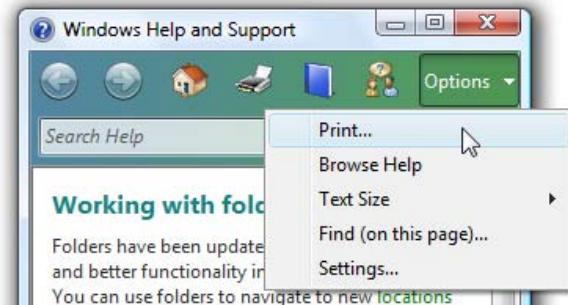
より良い例:



/Windows Fax とスキャンツールバーにはコマンド数が少なく、この優れたツールバーでは最も重要なコマンドにラベルが付いています。

- ツールバーに直接提示されているコマンドは、ツールバー メニューに提示しないでください。

間違った例:



この例では、[印刷] がツールバーに直接提示されているため、メニューに提示する必要はありません。

構成と順序

- ツールバー内のコマンドは関連グループに分類します。
- 最初によく使用されるグループを配置し、グループ内では、コマンドを論理的な順序で配置します。全体では、コマンドは、最もよく使用されるコマンドが最初に表示されるようにしながら、探しやすくするために論理的にも自然な順序になるようにします。こうすると、特にコマンドがオーバーフローした場合に、最も効率的になります。
- グループ間のコマンドの結びつきが弱い場合、グループの区切り線を使用します。こうすると、グループ分けがはっきりし、コマンドが探しやすくなります。



Windows Mail のグループ化されたツールバーの例。

- リスクのあるコマンドを使用頻度の高いコマンドの隣に配置しないようにします。両者を離して並べるか、または別のグループにします。また、リスクのあるコマンドをツールバーに配置せず、メニュー バーまたはコンテキストメニューのみに配置することを検討します。

許容される例:



より良い例:



より良い例では、[削除] コマンドと [印刷] コマンドが物理的に離れた位置に配置されています。

- オーバーフロー シェブロンを使って、表示されていないコマンドがあることを示します。ただしこの方法は、すべてのコマンドを表示するのに必要な領域を確保できない場合に限って使用してください。

間違った例:



このオーバーフロー シェブロンは、すべてのコマンドが表示されていないことを示していますが、レイアウトを改善すれば多くのコマンドを表示できます。

- 最もよく使用されるコマンドには、小さいウインドウ サイズでもツールバーから直接アクセスできる(つまり、コマンドをオーバーフローさせない)ようにします。必要であれば、コマンドの順序を変更したり、使用頻度の低いコマンドをメニュー ボタンや分割ボタンに移動したり、ツールバーから完全に削除したりします。それでも問題が解決しない場合、[ツールバーのスタイル](#)を選択し直してください。

#### メニュー バーの非表示化

通常、ツールバーとメニュー バーは、競合することなくそれぞれの長所を活用でき、ツールバーとメニュー バーは組み合わせることによって最大限の効果が得られます。

- ツールバーのデザイン上、メニュー バーを表示するとしつこく感じる場合は、既定でメニュー バーを非表示にします。
- キーボード ユーザーにとってはメニュー バーの方が使いやすいため、メニュー バーを完全に削除するのではなく、非表示にします。
- メニュー バーを復元する手段として、[表示] メニュー カテゴリ(メインツールバー)または[ツール] メニュー カテゴリ(補助ツールバー)で[メニュー バー] チェックマーク オプションを用意します。詳細については、「[標準メニューと分割ボタン](#)」を参照してください。
- Alt キーを押すとメニュー バーが表示され、最初のメニュー カテゴリに入力フォーカスが設定されます。

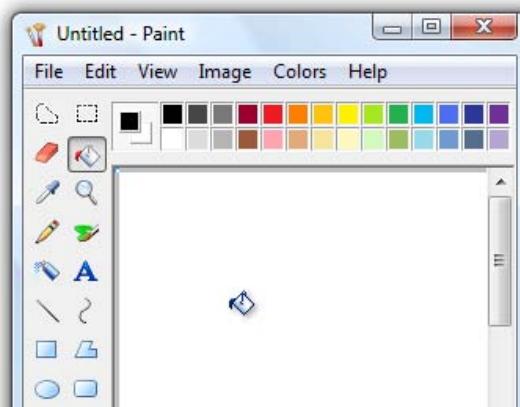
#### 対話操作

- マウス ポインターをポイントしたときに、アイコンがクリック可能であることを示す[アフォーダンス](#)をボタンに表示します。ツールヒントのタイムアウト後は、ツールヒントまたは情報ヒントを表示します。



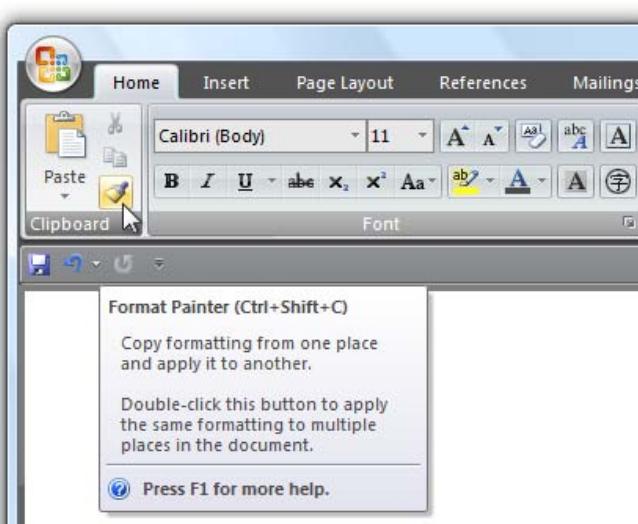
この例では、さまざまな表示状態を示しています。

- 左ボタンのシングルクリックでは、次のようにになります。
  - [コマンド ボタン](#)の場合、通常どおりコントロールを操作します。
  - [モード ボタン](#)の場合、コントロールを表示して、現在選択されているモードを反映します。モードによってマウス操作の動作が変わるのは、ポインターも変更します。



この例では、ポインターを変更することでマウスの操作モードを示しています。

- [プロパティ ボタン](#)および[ドロップダウン リスト](#)の場合、コントロールを表示して、現在選択されているオブジェクトがある場合、その状態を反映します。対話操作時には、コントロールの状態を更新し、選択されたオブジェクトに変更を適用します。何も選択されていない場合は、何もしません。
- 左ボタンのダブルクリックでは、左ボタンのシングルクリックと同じ操作を実行します。
  - 例外: ごくまれに、ツールバー コマンドの方がモーダルでより効率的に使用できることがあります。この場合、ダブルクリックによってモードを切り替えます。



この例では、[書式のコピー/貼り付け] コマンドをダブルクリックすると、指定した書式がその後のすべてのクリックで適用されるモードになります。このモードは、左ボタンのシングルクリックで解除できます。

- 右クリックでは、次のようになります。
  - カスタマイズ可能なツールバーの場合、ツールバーをカスタマイズするためのコンテキストメニューを表示します。マウスの右ボタンが押下げられたとき（マウスのボタンが元に戻ったときではない）にメニューを表示します。
  - その他のツールバーの場合は、何もしません。

#### アイコン

- ドロップダウンリストを除くすべてのツールバーのコントロールにアイコンを付けます。



ドロップダウンリストにはアイコンは必要ありませんが、その他のツールバーのコントロールにはアイコンが必要です。

- ツールバーのアイコンが、ツールバーの背景色に対して明瞭に見えるようにします。ツールバーのアイコンの評価は、常にコンテキスト内で、ハイコントラストモードで行います。
- 最もよく使用されるコマンドの場合は特に、目的を明確に伝えるアイコンのデザインを選択します。ツールヒントだけでは効率的にコマンドを探すことは難しいため、デザインの優れたツールバーには、コマンドの内容がひとめでわかるアイコンが必要です。ただし、あまり頻繁に使われないコマンドであれば、わかりにくいものでも問題ありません。
- 認識しやすく、区別しやすいアイコンを選択します（よく使用されるコマンドの場合は特に）。アイコンには特徴のある形や色を設定します。そうすることで、アイコンの記号を思い出すことができなくても、コマンドをすばやく探すことができます。
- ツールバーのアイコンを [Aero style のアイコンのガイドライン](#) に準拠させるようにします。

詳細と例については、「[アイコン](#)」を参照してください。

#### 標準メニューと分割ボタン

ツールバーにメニュー ボタンや分割ボタンを使用する場合、可能な限り、次の標準メニュー構造および関連するコマンドを使用してください。メニュー バーとは異なり、ツールバーのコマンドにはアクセスキーを使用しません。

#### メイン ツール バー

次のコマンドは、標準メニュー バーで使用されるコマンドを反映したものなので、[メイン ツール バー](#)でのみ使用する必要があります。次の一覧に、ボタンのラベル（および種類）を、表示すべき順序で、区切り記号、ショートカットキー、省略記号も併せて示します。メニュー バーを表示および非表示にするコマンドは [表示] メニューの中にあります。

##### ファイル

新規作成	Ctrl + N
開く…	Ctrl + O
閉じる	
<区切り記号>	
上書き保存	Ctrl + S
名前を付けて保存…	
<区切り記号>	

送信

<区切り記号>

印刷... Ctrl + P

印刷プレビュー

ページ設定

<区切り記号>

終了 Alt + F4 (通常はショートカットキーなし)

編集 (メニュー ボタン)

元に戻す Ctrl + Z

繰り返し Ctrl + Y

<区切り記号>

切り取り Ctrl + X

コピー Ctrl + C

貼り付け Ctrl + V

<区切り記号>

すべて選択 Ctrl + A

<区切り記号>

削除 Del (通常はショートカットキーなし)

名前の変更...

<区切り記号>

検索... Ctrl + F

次を検索 F3 (通常はコマンドなし)

置換... Ctrl + H

ジャンプ... Ctrl + G

印刷 (分割ボタン)

印刷... Ctrl + P

印刷プレビュー

<区切り記号>

ページ設定

表示 (メニュー ボタン)

メニューバー (表示されている場合はチェックあり)

詳細ウィンドウ (表示されている場合はチェックあり)

プレビュー ウィンドウ (表示されている場合はチェックあり)

ステータスバー (表示されている場合はチェックあり)

<区切り記号>

ズーム

拡大 Ctrl + +

縮小 Ctrl + -

<区切り記号>

テキスト サイズ (選択した設定にはマークあり)

最大

大

中

小

最小

<区切り記号>

全画面表示 F11

最新の情報に更新 F5

ツール (メニュー ボタン)

...

<区切り記号>

オプション

ヘルプ (分割ボタン、[ヘルプ] アイコンを使用します)

<プログラム名> ヘルプ F1

<区切り記号>

<プログラム名> のバージョン情報

補助ツールバー

次のコマンドは、標準メニュー バーを補うものです。次の一覧に、ボタンのラベル (および種類) を、表示すべき順序で、区切り記号、ショートカットキー、省略記号も併せて示します。メニュー バーの表示および非表示に関するコマンドは [ツール] メニューにあります。

印刷 (分割ボタン)

印刷... Ctrl + P

印刷プレビュー

<区切り記号>

ページ設定

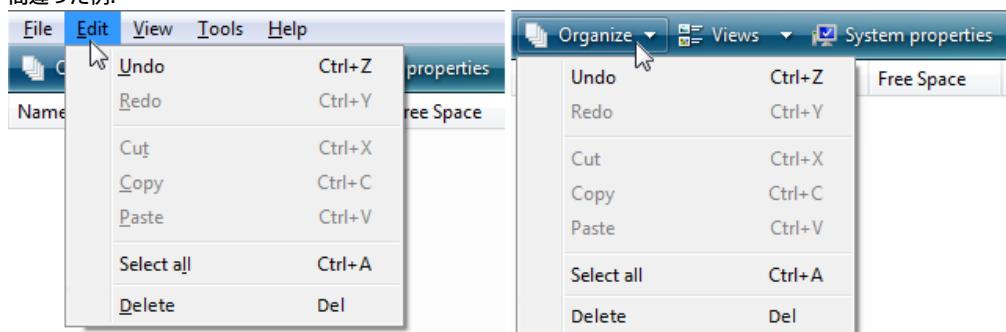
ツール (メニュー ボタン)  
メニュー バー (表示されている場合はチェックあり)  
詳細 ウィンドウ (表示されている場合はチェックあり)  
プレビュー ウィンドウ (表示されている場合はチェックあり)  
ステータス バー (表示されている場合はチェックあり)  
<区切り記号>  
印刷... (他の場所にない場合)  
検索...  
<区切り記号>  
全画面表示 F11  
最新の情報に更新 F5  
<区切り記号>  
ズーム  
拡大 Ctrl + +  
縮小 Ctrl + -  
<区切り記号>  
テキスト サイズ (選択した設定にはマークあり)  
最大  
大  
中  
小  
最小  
<区切り記号>  
オプション

整理 (メニュー ボタン)  
新しい フォルダー Ctrl + N  
<区切り記号>  
切り取り Ctrl + X  
コピー Ctrl + C  
貼り付け Ctrl + V  
<区切り記号>  
すべて 選択 Ctrl + A  
<区切り記号>  
削除 Del (通常はショートカットキーなし)  
名前の変更  
<区切り記号>  
オプション

ページ (メニュー ボタン)  
新しい ウィンドウを開く Ctrl + N  
<区切り記号>  
ズーム  
拡大 Ctrl + +  
縮小 Ctrl + -  
<区切り記号>  
テキスト サイズ (選択した設定にはマークあり)  
最大  
大  
中  
小  
最小

補助ツールバーのカテゴリ名は、より広範囲のコマンドを扱う必要があるため、標準メニューのカテゴリ名とは異なります。たとえば、[整理] カテゴリには編集に関連のないコマンドが含まれているため、[編集] ではなく [整理] というカテゴリ名を使用しています。メニュー バーとツール バーの一貫性を維持するために、標準メニューのカテゴリ名を使用しても誤解のおそれがない場合は、そのようにします。

#### 間違った例:



この例では、ツールバーに標準的な [編集] メニューのコマンドが含まれているため、一貫性を保つために [整理] ではなく [編集] というカテゴリ名を使用する必要があります。

#### パレット ウィンドウ

- パレット ウィンドウでは、短いタイトルバーを使用して、画面領域を最小限に抑えます。タイトルバーには [閉じる] ボタンを追加します。
- タイトルバーのテキストを、パレット ウィンドウを表示したコマンドに設定します。
- [センテンススタイルの大文字化](#)を使用し、末尾に句読点は付けません。
- ウィンドウの管理コマンド用にコンテキストメニューを用意します。このコンテキストメニューは、ユーザーがタイトルバーを右クリックしたときに表示されます。

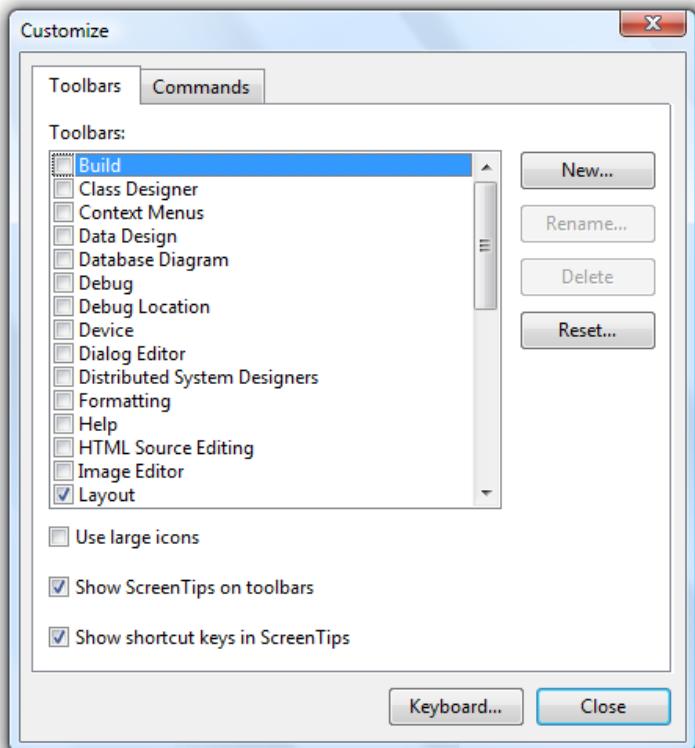


この例では、ユーザーがタイトルバーを右クリックすると、コンテキストメニューを表示することができます。

- 可能で便利であれば、パレット ウィンドウのサイズを変更できるようにします。ウィンドウ枠に重ねたときに表示されるサイズ変更ポインターによって、ウィンドウがサイズ変更可能であることを示します。
- パレット ウィンドウを再表示する場合、前回アクセスしたときの状態で表示します。閉じるときに、ウィンドウのサイズと場所を保存します。再表示する場合、保存したウィンドウのサイズと場所を復元します。また、これらの属性を、ユーザーごとに、プログラムインスタンスレベルで保持することも検討します。

#### カスタマイズ

- カスタマイズは、複数の行で構成されるツールバーに対して提供します。[ラベルのないアイコン](#) スタイルのみカスタマイズが必要です。コマンド数の少ないシンプルなツールバーには、カスタマイズは必要ありません。
- 適切な既定の構成を提供します。一般的なシナリオでは、ユーザーがツールバーをカスタマイズする必要がないようにします。ユーザーが不適切な初期構成を修正してカスタマイズを行うことをあてにしないでください。ほとんどのユーザーがツールバーをカスタマイズしないことを前提としてください。
- 次のコマンドを含むコンテキストメニューを用意します。
  - 利用可能なツールバーを表示するためのチェックボックスリスト
  - ツールバーのロック/ロック解除
  - カスタマイズ...
- カスタマイズ可能なツールバーを既定でロックします。これによって、誤って変更が行われないようにします。
- [カスタマイズ] コマンドで、オプションのダイアログボックスを表示します。このダイアログボックスで、表示するツールバーや各ツールバーに配置するコマンドを選択できます。



この例では、Visual Studio で提供されたオプションのダイアログ ボックスで、ツールバーのカスタマイズを行います。

- [カスタマイズ] のオプションのダイアログ ボックスに [リセット] コマンドを用意して、元のツールバー構成に戻すことができます。
- 次の点について、ドラッグ アンド ドロップによってツールバーをカスタマイズできるようにします。
  - ツールバーの順序や位置を設定します。
  - ツールバーの長さを設定します。小さすぎてコンテンツを表示できないツールバーは、[オーバーフロー・シェvron](#)を使って表示します。
  - 可能であれば、ツールバーをドッキング解除してパレット ウィンドウにしたり、パレット ウィンドウをドッキングしてツールバーにしたりします。

[カスタマイズ] のオプションのダイアログ ボックスでは、次の点についてカスタマイズできるようにします。

- ツールバーのコンテンツを設定します。
- ツールバーのコンテンツの順序を設定します。

こうすることで、ユーザーはより直接的かつ効率的に変更を行うことができます。

- すべてのツールバーのカスタマイズを保存します。これはユーザーごとに行います。

#### 省略記号の使用

ツールバー コマンドは即座に実行する操作のために使用されるものですが、その操作を実行するためには、さらに詳しい情報が必要な場合があります。省略記号は、コマンドが有効になるにはさらに詳しい情報が必要であることを示すために使用します。そのようなコマンドでは、省略記号をツールヒントやラベルの最後に付けます。



この例では、[印刷...] コマンドをクリックすると、情報を入力するための [印刷] ダイアログ ボックスが表示されます。

ただし、コマンドの効果がすぐに有効にならない場合は、省略記号は必要ありません。したがって、たとえば、共有の設定には追加情報が必要であっても、コマンドの効果がすぐに有効にならないため、省略記号は付いていません。

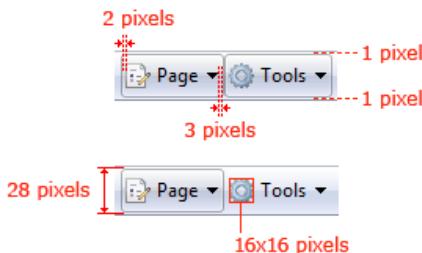


[共有の設定] コマンドは、効果がすぐに有効にならないため、省略記号は付いていません。

ツールバーは常に表示されており、領域が優先されるため、省略記号はできる限り使用しないようにします。

注: ツールバーで表示されるメニューには、[メニューの省略記号のガイドライン](#)を適用してください。

## 推奨されるサイズと間隔



標準ツールバーに推奨されるサイズと間隔です。

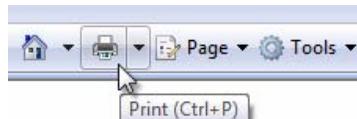
## ラベル

### 全般

- センテンススタイルの大文字化を使用します。
  - 例外: 古いアプリケーションでは、大文字/小文字の形式が混在しないように、[タイトルスタイルの大文字化](#)を採用することもできます。

### ラベルなしのアイコン ボタン

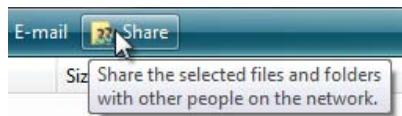
- コマンドにラベルを付けるには、[ツールヒント](#)を使用します。ツールヒントテキストには、ボタンにラベルが付いていればラベルの内容を記載しますが、ショートカットキーがある場合はそれを含めます。



アイコン ボタンのツールヒントの例。

### ラベル付きアイコン ボタン

- 簡潔なラベルを使用します。可能であれば1語に、最大でも4語（12文字程度）に納めます。
- ラベルはアイコンの右側に配置します。
- コマンドにラベルを付けるには、[情報ヒント](#)を使用します。ボタンにラベルが付いているため、情報ヒントの代わりにツールヒントを使用すると冗長になります。



ラベル付きアイコン ボタンの情報ヒントの例。

### ドロップダウンリスト

- リストに常に値が含まれている場合、現在の値をラベルに使用します。



この例では、現在選択されているフォント名がラベルの役割を果たしています。

- 編集可能なドロップダウンリストに値がない場合は、[プロンプト](#)を使用します。



この例では、プロンプトは、ドロップダウンリストのラベルに使用されています。

### メニュー ボタンと分割ボタン

- 動詞ベースのメニュー ボタン名を優先します。ただし、"作成する (Create)"、"表示する (Show, View)"、または "管理する (Manage)" の場合は、動詞を省略します。たとえば、[ツール] および [ページ] メニュー ボタンには動詞は含まれていません。
- メニューの内容を明確に正しく説明する具体的な1語を使用します。名前は、メニュー全体を説明するような一般的なものにする必要はありませんが、ユーザーがメニュー項目を見たときに困惑しないように、十分予測可能なものにする必要があります。
- 情報ヒントの説明は必ずしも必要ありませんが、役に立つ場合は入力します。

### メニュー項目

- 動詞、名詞、または名詞句で始まるメニュー項目名を使用します（英語の場合）。
- 動詞ベースのメニュー名を優先します。ただし、"作成する (Create)"、"表示する (Show, View)"、または "管理する (Manage)" の場合は、動詞を省略します。たとえば、次のコマンドでは動詞を使用しません。
  - バージョン情報

- 詳細設定
- 全画面表示
- 新規作成
- オプション
- プロパティ
- 具体的な動詞を使用します。"変更する"、"管理する"など、汎用的でわかりにくい動詞を使用しないようにします。
- 単一のオブジェクトに適用されるコマンドには名詞の単数形を使用します(英語の場合)。複数のオブジェクトの場合は、複数形を使用します。
- 相互補完的なコマンドのペアの場合は、補完的であることがよくわかる名前を選択します。たとえば、[追加]と[削除]、[表示]と[非表示]、[挿入]と[削除]のようにします。
- メニュー項目名は、技術ではなく、ユーザーの目的と作業に基づいて選択します。
- メニュー項目名は、次に示す目的に沿って使い分けます。
  - オプション: プログラム オプションを表示します。
  - カスタマイズ: 特に機械的な UI 構成に関するプログラム オプションを表示します。
  - 個人設定: 一般的に使用される個人設定の概要を表示します。
  - 基本設定: 使用しません。代わりに [オプション] を使用します。
  - プロパティ: オブジェクトのプロパティ ウィンドウを表示します。
  - 設定: メニュー ラベルには使用しません。代わりに [オプション] を使用します。
- サブメニューを表示するメニュー項目のラベルには、省略記号を含めません。サブメニューの矢印は、別の選択が必要であることを示しています。

## ドキュメント

ツールバーに言及するときは、以下のことに留意します。

- ツールバーが 1 つしかない場合は、"このツールバー" といいます。
- ツールバーが複数ある場合、ツールバーの名称の後ろに "ツールバー" という語を付けます。既定で有効になっており、ファイルを開いたり印刷する基本的な作業を行うボタンを含むメインツールバーは、"標準ツールバー" といいます。
- "ツールバー" のように、"ツール" と "バー" の間に半角スペースを入れます ("メニューバー" も同様)。
- ツールバーのボタンには、ツールヒント ラベルを使用します。大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、省略記号は含めません。
- ツールバー メニューのボタンは、そのラベルに "メニュー" という語を付けます。大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。
- ツールバー コントロールは、通常、"ツールバー ボタン" と呼びます。
- ユーザー操作を説明する場合は、ツールバー ボタンや読み取り専用のドロップダウン リストでは "クリック"、編集可能なドロップダウン リストでは "入力" を使用します。"選択 (choose, select, pick)" は使用しません。
- プログラミング文書を除き、メニュー ボタンの記述として "カスケード"、"フルダウント"、"ドロップダウン"、"ポップアップ" は使用しません。
- 使用できないアイテムは "利用不可の" と表現し、"薄く表示された"、"無効になった"、"灰色表示の" とはしません。"無効" は、プログラミング文書で使用します。
- ラベルは半角の角かっこ ([]) で囲み、可能な場合は太字にします。

例:

- ツールバーの [ページ] メニューで、[このページを電子メールで送信] をクリックします。
- ツールバーの [フォント] ボックスに、「Segoe UI」と入力します。
- [書式設定] ツールバーで [表示] をポイントし、[コメント] をクリックします。

## リボン

リボンは、最新のユーザーインターフェイスです。クリックの回数を最小限に抑え、試行錯誤の操作を減らし、ヘルプを参照することなく、ユーザーが効率的に直接コマンドを見つけ、理解し、使用できるようにします。

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

ガイドライン

タブ

コンテキストタブ

モーダルタブ

標準的なリボンタブ

グループ

標準的なリボングループ

コマンド

全般

提示方法

対話操作

ギャラリー

プレビュー

アイコン

拡張ツールヒント

アクセスキーおよびキーヒント

アプリケーションボタン

クイックアクセスツールバー

ダイアログボックス起動ツール

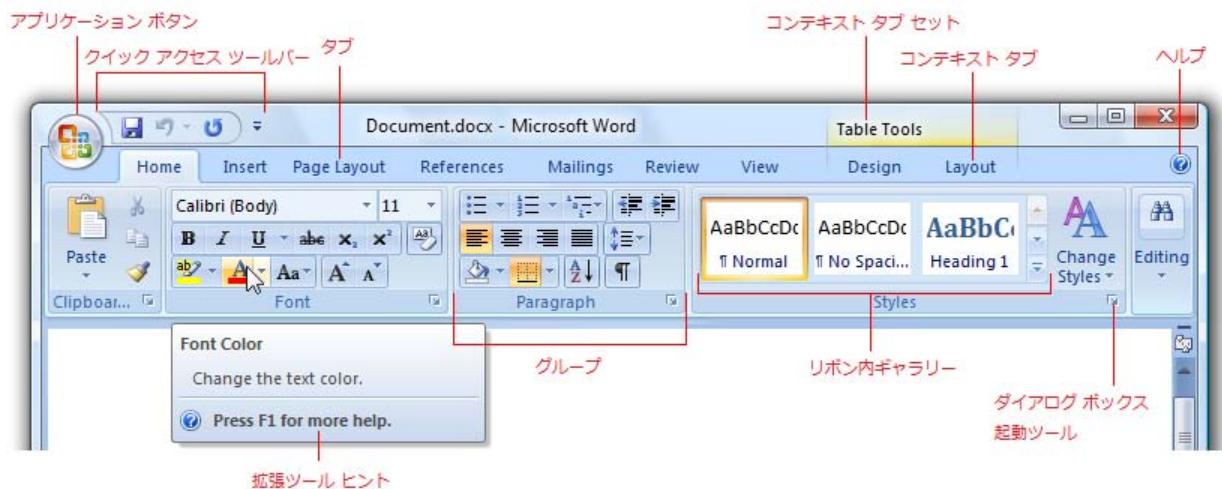
ラベル

ドキュメント

リボンのデザインプロセス

"リボン"とは、プログラムの機能をウィンドウ上部にある一連のタブに整理するコマンドバーです。

リボンを使用すると、機能を見つけやすくなり、プログラム全体をすぐに習得でき、プログラムの操作方法を把握しやすくなります。リボンは従来のメニューバーとツールバーの代わりに使用できます。



### 一般的なリボン

リボンタブは、ラベル付けされた、密接に関連するコマンドセットから成るグループで構成されています。タブとグループに加えて、リボンには以下の要素が含まれます。

- ・ "アプリケーションボタン"は、ファイル関連コマンドなど、ドキュメントや作業領域に対して何らかの操作を行うコマンドのメニューを提示します。
- ・ "クイックアクセスツールバー"は、よく使用するコマンドを表示する小さくてカスタマイズ可能なツールバーです。
- ・ "コアタブ"は、常に表示されているタブです。
- ・ "コンテキストタブ"は、特定のオブジェクトの種類が選択されたときのみ表示されます。常に表示されているタブは、"コアタブ"と呼ばれます。
- ・ "タブセット"は、オブジェクトの種類別のコンテキストタブの集まりです。オブジェクトには複数の種類がある(たとえば、画像付きの表内のヘッダーは3種類)ので、一度に複数のコンテキストタブセットが表示されることがあります。
- ・ "モーダルタブ"は、印刷プレビューなど特定の一時的なモードで表示されるコアタブです。
- ・ "ギャラリー"は、グラフィカルに提示されるコマンドまたはオプションの一覧です。結果ベースのギャラリーでは、コマンドそのものではなく、コマンドまたはオプションの効果が示されます。リボン内ギャラリーは、ポップアップウィンドウとは対照的にリボン内に表示されます。
- ・ "拡張ツールヒント"は、関連付けられたコマンドを簡潔に説明し、ショートカットキーを提供します。また、拡張ツールヒントにはグラフィックやヘルプへの参照が含まれている場合があります。拡張ツールヒントを使用すると、コマンドに関連するヘルプを参照する必要が少くなります。
- ・ "ダイアログボックス起動ツール"は、特定のグループの下部にあるボタンで、グループに関連する機能が含まれているダイアログボックスを開きます。

リボンはMicrosoft Office 2007で初めて導入されました。Officeでリボンの使用が必要な理由、およびリボンを使用して多くの問題を解決できる理由については、[リボンの開発ストーリーに関するページ](#)を参照してください。

従来のメニューやツールバーを現在使用しているプログラムにリボンを適用する方法の詳細については、[「リボンのデザインプロセス」](#)を参照してください。

注: メニュー、ツールバー、コマンドボタン、アイコンに関するガイドラインは、それぞれ別の項目として記載しています。

Office UI のライセンスについては、「[Office UI ライセンス](#)」を参照してください。

## 適切なユーザーインターフェイスかどうかの判断基準

リボンの使用が適切かどうかは、以下の点に基づいて判断します。

### プログラムの種類

- 設計の対象となるプログラムの種類は何か。プログラムの種類は、リボンの使用が適切かどうかを判断する有効な目安となります。リボンは、ドキュメントの作成プログラムおよびオーサリングプログラムだけでなく、ドキュメントビューアーやブラウザーでも効果的です。その他の種類のプログラムでもリボンを使用することができますが、その他のコマンドの提示方法の方が適していることがあります。一般的に、軽量プログラムには軽量のコマンド提示方法を使用する必要があります(プログラムの種類の一覧については、「[プログラム コマンドのパターン](#)」を参照してください)。

### 見つけやすさと習得に関する問題

- ユーザーがコマンドを探すときに手間取る。または、ユーザーが既にプログラム内にある機能を要求している。該当する場合は、ラベルだけで内容を把握でき、関連するコマンドがグループ化されているリボンを使用することで、コマンドを見つけやすくなります。また、メニューバーやツールバーよりもリボンを使用する方が拡張性に優れ、今後、機能が増えたときに対応できます。
- ユーザーがプログラムのコマンドを理解するときに問題がある。または、適切なコマンドを選択したり、コマンドのしくみについて把握するときに、試行錯誤が必要である。該当する場合は、リボンで、ギャラリーおよびリアルタイムのプレビューをベースとする結果指向のコマンドを使用すると、コマンドを理解しやすくなります。

### コマンドの特徴

- 複数の場所でコマンドが提示されている。または、プログラムが既に存在する場合、コマンドがメニューバー、ツールバー、タスク ウィンドウ、作業領域自体に提示されている。該当する場合は、リボンを使用して複数のコマンドを1つの場所にまとめ、コマンドを見つけやすくなります。
- コマンドが適用されるのがウィンドウ全体か、特定の作業ウィンドウのみか。リボンは、ウィンドウ全体または特定のオブジェクトに適用されるコマンドに最適です。個別の作業ウィンドウにはインプレース コマンドの方が適しています。
- コマンドのほとんどを直接提示できるかどうか。つまり、ユーザーが1回のクリックでコマンドを操作できるかどうか。よく使用されるコマンドにメニューおよびダイアログ ボックスからアクセスする場合、それらのコマンドをリファクタリングして直接アクセスできる。コマンドによっては、メニューおよびダイアログ ボックスを使用して提示できますが、この方法でほとんどのコマンドを提示するとリボンの効率性が損なわれ、メニューバーを選択する方が適切な場合があります。

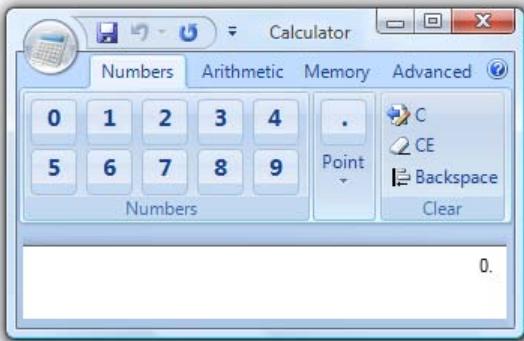
### コマンドの拡張性

- コマンドの数が少ない。最もよく使用されるコマンドを1つのシンプルなツールバーに簡単に表示できる。コア タブおよびコンテキスト タブを追加して、最もよく使用されるタスクを実行するために単体で使用できるシンプルな[ホーム] タブに構成できる場合は、リボンを使用する価値があります。該当しない場合は、リボンを使用しても、少数のコマンドのために動作が重くなり、メリットがない場合があります。
- コマンドの数が多い。リボンを使用すると、8つ以上のコア タブが必要になる。ユーザーが一般的なタスクを実行するときにタブを頻繁に変更する必要がある。該当する場合は、タブを変更する必要のないツールバー、または、タブを変更する必要がある場合でも、一度に複数のタブを開くことができるパレット ウィンドウを使用する方が効率的な場合があります。
- ほとんどの場合、ユーザーは少数のコマンドを使用する傾向にある。該当する場合は、これらのコマンドを[ホーム] タブ上に配置することで、ユーザーはリボンを効率的に使用できます。タブを頻繁に変更する場合、リボンの使用は効率的ではありません。
- プログラムのコンテンツ エリアを最大限に大きくするとプログラムにメリットがある。該当する場合は、リボンよりもメニューバーおよびツールバーを単体で使用する方が領域の面で効率的です。ただし、プログラムでツールバーが3行以上必要であったり、作業ウィンドウを使用する場合は、リボンを使用する方が領域の面で効率的です。
- ユーザーはプログラムの大きいウィンドウ内の特定のエリアで長時間作業する傾向にある。該当する場合は、ミニツールバー、パレット ウィンドウ、直接コマンドを隣接させると、ユーザーにとって使い勝手が良くなります。作業領域とリボンの間を何度も移動するのは効率的ではありません。
- 効率性と柔軟性を持たせるために、ユーザーがコマンドを提示するコンテンツ、位置、またはサイズに大幅な変更を加える必要がある。該当する場合は、カスタマイズと拡張に対応するツールバーやパレット ウィンドウの方が適しています。ツールバーの種類によっては、ドッキングを解除してパレット ウィンドウとして使用できるものもあります。パレット ウィンドウは、移動、サイズ変更、カスタマイズに対応しています。

最後に、次の最も重要な点に基づいて判断します。追加の領域という負荷と、コマンドを整理するタブが必要であるというコストに見合った、見つけやすさや習得のしやすさや生産性の向上が得られるかどうか。該当する場合は、リボンの使用が最適です。確信がない場合は、リボンベースのデザインをテストし、リボンの次に最適な代替案と比較してユーザビリティについて検討します。

リボンは、ユーザーの注意を引き付ける新しいコマンドの提示形式で、プログラムを刷新できる有効な手段です。優れた特長を持つリボンですが、すべてのプログラムに適しているわけではありません。

間違った例:



このようにはしないでください。

## デザインコンセプト

### 既存のプログラムにリボンを組み込む

単純に既存のプログラムの従来のメニュー バーとツール バーのデザインをリボン形式にリファクタリングしても、リボンの価値の大部分を無駄にしていることになります。多くの場合、ギャラリーおよびリアルタイムのプレビューの形式で、即時型で結果指向のコマンドを提示するためにリボンを使用するときに、その真価を最大限に引き出すことができます。結果指向のコマンドを使用すると、コマンドを理解しやすくなり、ユーザーの作業効率と生産性が大幅に向上します。既存のコマンドをリファクタリングするのではなく、プログラムでコマンドを実行する方法を完全にデザインし直すことをお勧めします。

効率的なリボンを簡単に作成できるとは考えないでください。また、リボンを使用すれば良いプログラムができる場合も考えないでください。効率的なリボンの作成には、多くの時間と労力が必要です。前述したコマンドの再デザインに必要な時間と労力を注ぐ意思があるかどうかは、リボンの使用が適切かどうかを判断する重要な要素になります。

既存のプログラムでコマンドをリボンに移行する方法の詳細については、「[リボンのデザイン プロセス](#)」を参照してください。

### リボンの特性

従来のメニュー バーおよびツール バーと比べて、リボンには以下の特徴があります。

- すべてのコマンドに対応する単一のユーザー インターフェイス (UI)。メニュー バーは包括的で習得しやすく、ツール バーは効率的で直接的ですが、画面領域をもう少し拡大して、メニュー バーとツール バーの機能をすべて持つ単一のコマンド UI を作成できれば便利です。単体の UI だけで構成されるリボンを使用すると、ユーザーは目的のコマンドがどの UI にあるのかを考えずに済みます。
- 一目瞭然。メニュー バー コマンドはラベルで処理内容がわかりますが、多くの場合非表示になっています。ツール バー ボタンは、画面領域を節約するために主にラベルの代わりにアイコンで表示されており (一部のツール バー ボタンではラベルとアイコンの両方を使用)、アイコンの意味が伝わりにくい場合はツールヒントに頼っています。通常、ユーザーがアイコンの意味を把握しているのは最もよく使用するコマンドだけです。ほとんどのコマンドをラベル付きのアイコンで提示することによって、リボン コマンドは見るだけで内容を把握できます。ツールヒントを使用するのは補足情報を作成するときだけです。コマンドを理解するために他の場所 (ヘルプなど) にアクセスする必要はほとんどなくなります。
- ラベル付けされたグループ。メニュー カテゴリはラベル付けされていますが、ドロップダウン メニュー内のグループはラベル付けされておらず、ラベル付けされていない区切り記号でのみ示されます。ツール バー内のグループもラベル付けされていない区切り記号で示されます。リボンでは、コマンドをラベル付けされたグループに整理することで、コマンドを見つけやすく、コマンドの目的を確認しやすくなります。
- モーダルで階層化されていない。メニュー バーは、コマンドの階層を作成して拡張できます。項目数が多いメニューでは、1つまたは複数のレベルのサブメニューを使用して、さらにコマンドを表示できます。リボン コマンドは、ツール バー コマンドよりも領域が必要なので、タブを使用して拡張します。タブの使用でリボンはモーダルになり、ユーザーは場合によってコマンドを見つけるためにモードを変更する必要があります。ただし、タブ内のほとんどのコマンドは直接実行されるか、単体の分割ボタンまたはメニュー ボタンを使用するもので、階層化はされていません。
- 直接的、即時型。1回のクリックで呼び出される (複数のメニューを開く必要がない) コマンドは直接的で、即座に効果が現れる (追加入力のためのダイアログ ボックスが表示されない) コマンドは即時型です。メニュー バー コマンドは、常に非直接的で、多くの場合、即時型ではありません。ツール バーと同様、ほとんどのリボン コマンドは、最もよく使用されるコマンドを1回のクリックで呼び出し、追加入力のためのダイアログを必要とせず、直接かつ即座に実行するように設計されています。
- 画面領域が必要。メニュー バーとツール バーは一般的に、画面領域を効率的に使用できるように設計されています。両方のメリットを得るために、リボンではメニュー バーにツール バーを3行追加した程度の領域を垂直方向に消費する場合があります。ツール バーが3行以上あるプログラムはほとんどないため、リボンを使用する場合は、一般的に従来のコマンド UI よりも多くの領域が必要です。
- アプリケーション ボタンおよびクイック アクセス ツール バーを配置。リボンには常に、アプリケーション ボタンとクイック アクセス ツール バーが表示されます。これにより、ユーザーはタブを変更せずにファイルに関連するコマンドや、よく使用するコマンドにアクセスできるので、プログラム間の一貫性が向上します。
- 最小限のカスタマイズ。メニュー バーの表示は固定であるのに対して、ツール バーの多くはカスタマイズ可能で、ユーザーは位置、サイズ、コンテンツを設定できます。リボン自体はカスタマイズできませんが、クイック アクセス ツール バーで最小限のカスタマイズ機能を提供しています。
- キーボードによる操作性の向上。メニュー バーはキーボードでの操作性に優れ、Alt キーを押すとメニュー バーに入力フォーカスが移動します。ツール バーは、ウィンドウのコンテンツとキーボード ナビゲーションを共有しているので、このようないくつかありません。このため、ユーザーは Tab キーを使用してツール バーに移動してから (ツール バーには最後のタブストップが割り当てられています)、方向キーを使用して特定のコマンドまで移動する必要があります。

これに対して、リボンではキーヒントによって、キーボードの操作性が強化されています。キーヒントを使用するには、通常、以下の3つの手順を実行します。

1. Alt キーを押して、キーヒント モードに入ります。

2. 文字キーを押して、タブ、アプリケーション ボタン、またはクイック アクセス ツール バーのコマンドを選択します。

3. タブ内で、1つまたは2つの文字キーを押して、コマンドを選択します。

この方法は視覚的で、柔軟性もあり、拡張したり、記憶しやすいアクセスキーを割り当てたりすることができます。

アクセスキーとショートカットキーを混同しないでください。アクセスキーとショートカットキーは両方ともキーボードでUIにアクセスできますが、それぞれ異なる目的とガイドラインがあります。詳細については、「[キーボード](#)」を参照してください。

#### リッチコマンドの特性

"リッチコマンド"とは、リボンで使用するコマンドの表示および対話操作のことを示しますが、必ずしもリボンコンテナーを使用するわけではありません。リッチコマンドには以下の特徴があります。

- ラベル付け。すべてのコマンドには、処理内容が一目でわかるラベルが付けられています。ただし例外として、アイコンがよく知られている場合、および領域を優先する場合に限り、ラベルは省略されます。

正しい例:



これらのコマンドはよく知られているので、ラベルは必要ありません。

間違った例:



これらのアイコンだけでは意味が伝わらないので、リッチコマンドにラベルが必要です。

- サイズ変更。サイズを均一化するのではなく、使用頻度と重要度に応じてコマンドのサイズを変更します。これにより、最もよく使用されるコマンドの視認性とクリックしやすさを向上させるだけでなく、タッチしやすくします。



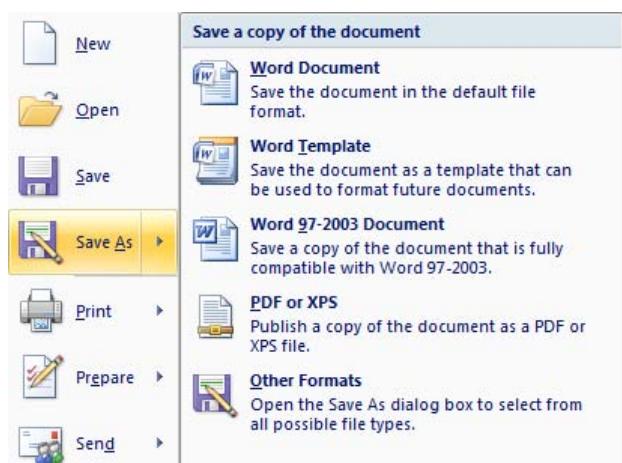
この例では、最もよく使われるボタンは他のボタンよりも大きくなっています。

- 動的なサイズ変更。リッチコマンドコントロールでは、自動的にサイズの変更を行い、使用可能な領域を最大限に活用します。一方、固定サイズでは、サイズが不足する場合に表示を省略したり、オーバーフローを使用します。



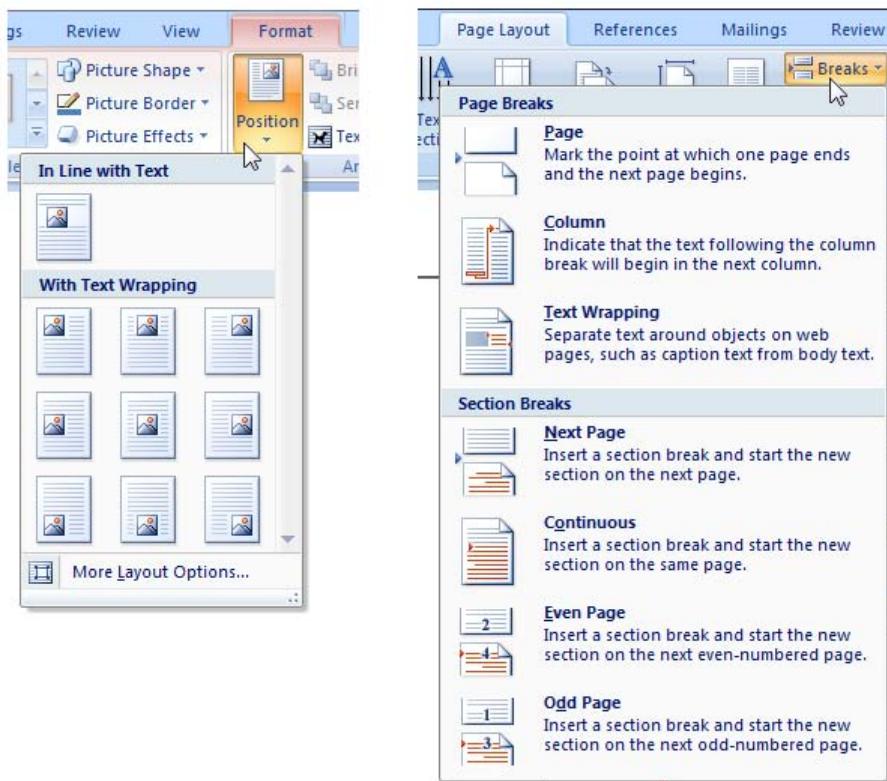
この例では、使用可能な領域に合わせてコマンドボタンのサイズが適切に変更されています。

- 分割ボタン。分割ボタンは、よく使用されるコマンドを直接実行できるようにしながら、必要に応じてコマンドのバリエーションセットをまとめに適した方法です。



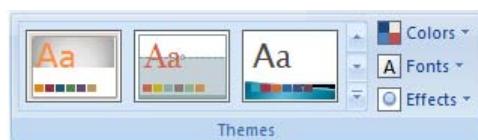
この例では、/名前を付けて保存/コマンドで分割ボタンを使用しています。メインボタンでは最もよく使用するコマンドを実行し、メニュー部分ではコマンドのバリエーションをメニューで表示しています。

- リッチドロップダウンメニューおよびギャラリー。ドロップダウンメニュー、ドロップダウンリスト、ギャラリーでは、通常、グラフィックや説明文を使用して選択肢の効果を伝えたり、違いを区別するために領域が必要です。大きな枠組みのオプションセットを整理するために、カテゴリが使用されます。



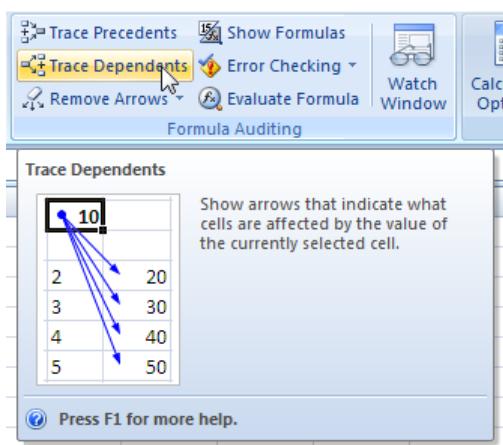
これらの例では、メニュー ボタンをクリックすると、効果を示す選択肢の一覧が表示されます。

- リアルタイムのプレビュー。ユーザーが書式設定オプションにマウス ポインターを合わせると、プログラムで実際のコンテンツを使用してその書式設定を選択した場合にどのように見えるかが表示されます。



リアルタイムのプレビューは、マウス ポインターを合わせた書式設定オプションの適用結果を示します。

- 拡張ツールヒント。拡張ツールヒントは、関連するコマンドを簡潔に説明したり、ショートカット キーを表示したりします。また、拡張ツールヒントではコマンドに関するヘルプはほとんど必要ありませんが、グラフィックやヘルプへの参照が含まれている場合があります。



拡張ツールヒントは、関連するコマンドを簡潔に説明しています。

リボンは、すべてのプログラムに適したものではありませんが、リッチ コマンドによってすべてのプログラムでメリットを得る可能性があります。

リボンに常備されているアプリケーション ボタンおよびクイック アクセス ツール バー

アプリケーション ボタンおよびクイック アクセス ツール バーはあらゆるコンテキストで有用なコマンドを表示することができるため、タブを変更する必要が少なくなります。3つのコンポーネントは論理的に独立していますが、リボンにはアプリケーション ボタンおよびクイック アクセス ツール バーが常備されている必要があります。コマンドはリボンまたはアプリケーション ボタンに配置できることを考えると、コマンドをどこに配置するか判断に迷う場合があります。その選択には根拠が必要です。

アプリケーション ボタンは、従来 [ファイル] メニューに配置されるコマンド（ファイルの作成、ファイルを開く、ファイルの保存、ドキュメントを印刷、送信して公開するコマンド）など、ファイルに対して何らかの操作を行うコマンドのメニューを表示するときに使用されます。

これに対して、リボン自体はウィンドウのコンテンツに影響するコマンドのためのものです。たとえば、コンテンツの読み込み、変更、使用や表示の変更に使用されるコマンドがあります。

リボンを使用する場合は、プログラムがドキュメントやファイルに関与しなくても、アプリケーションボタンを使用する必要があります。このような場合は、アプリケーションメニューを使用して、印刷、プログラムのオプション、プログラムの終了のためのコマンドを提示します。こうしたプログラムにはアプリケーションボタンは必要ないともいえますが、アプリケーションボタンを使用するとプログラム間の一貫性が保たれます。[元に戻す]や[保存]コマンド、プログラムのオプションは常に同じ場所に配置されているので、ユーザーはこれらの場所を探す必要はありません。

クリックアクセスツールバーは、リボンで1つのタブのみを使用する場合でも必要です。これについても、すべてのコマンドが既に単一のタブ上で提示されているので、こうしたプログラムにはクリックアクセスツールバーは必要ないともいえますが、カスタマイズ可能なクリックアクセスツールバーを使用すると、プログラム間の一貫性が保たれます。たとえば、ユーザーが[印刷]コマンドをよく使用する場合、リボンを使用するすべてのプログラムでこのコマンドを実行できるようにする必要があります。

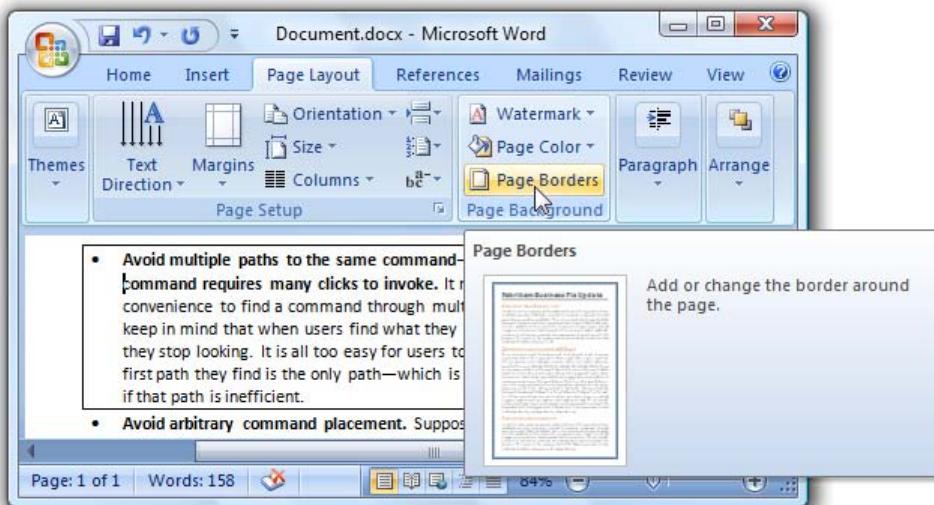
### 構成と見つけやすさ

リボンでタブとグループを使用することで、コマンドを整理して見つけやすくすることができます。ただし、適切に構成しないと、逆に見つけにくくなるという問題があります。配置されているコマンドと、コマンドが配置されている説明的なラベルを付けたタブ、グループの間で、明確で一意な対応付けが必要です。

リボンを使用してしばらくすると、ユーザーの頭の中でリボンのメンタルモデルが形成されます。そのメンタルモデルがユーザーにとって不可解であったり、非効率だったり、妥当でない場合は混乱やストレスを招きます。リボンを設計するうえで最も重要な目的は、コマンドをすばやく、迷わずに見つけやすくすることです。この目的を実現できない場合、リボンの設計は失敗します。目的を実現するには、入念な設計、ユーザー テスト、繰り返しによる改善が必要です。これは、容易に実現できるものではありません。

以下に、陥りやすい間違いを避ける方法をいくつか示します。

- 汎用的なタブ名およびグループ名は避けます。適切なタブ名またはグループ名にするには、できる限りタスクと目的に基づいた言葉を使用し、具体的なコンテンツを正確に言い表す必要があります。汎用的なタブ名やグループ名のほか、テクノロジーに基づいた名前を避けます。たとえば、ドキュメントを作成/オーサリングするプログラムのコマンドは、ほとんどすべてが、"編集"、"書式設定"、"ツール"、"オプション"、"詳細設定"、"詳細"とラベル付けされたタブに当てはまります。記憶に頼るようなラベルではなく、具体的で説明的なラベルを設定します。
- 特定の目的に限定されるタブ名およびグループ名は避けます。タブ名とグループ名は具体的にする必要がありますが、特定の目的に限定されていて、コンテンツが表示されたときに違和感がある名前も避ける必要があります。多くの場合、ユーザーは消去法で目的の名前を探しているので、名前が正しく解釈されずに、タブやグループが見過ごされることがないようにします。
- 複数の方法で同じコマンドにアクセスすることは避けます。特にその方法が予想外であったり、コマンドを呼び出すために何度もクリックする必要がある場合は注意します。複数の方法があると、コマンドを見つけるのに便利だと感じます。しかし、ユーザーは必要なコマンドが見つかったらそれ以上は探さないことに注意してください。一般的にユーザーは、最初に見つけた方法が唯一の方法であると考えます。これは、方法が非効率的であったり、予想外であったりすると深刻な問題になります。さらに、重複したコマンドがあると、他の目的のコマンドを見つけることが難しくなります。



この例では、[ページ罫線] コマンドを使用して段落の罫線を変更できますが、[ホーム] タブ上にも直接的な方法があります。段落の罫線に関するコマンドを探しているユーザーがこの予想外の方法を発見した場合、それが唯一の方法であると考える可能性があります。

- 根拠のないコマンドの配置は避けます。適切なタブとグループを設計したはずが、いくつかのコマンドがどこにも当てはまらないことに気付くことがあります。この場合、自分で考えているほどタブとグループの設計が適切ではない可能性があるので、継続して改良していく必要があります。こうしたコマンドを適切ではない場所に配置して、この問題を解決したことにしてください。こうすると、ユーザーはすべてのタブを確認してコマンドを探すことになり、また見つけたコマンドの場所はすぐにわからなくなります。
- マーケティングに基づいた配置は避けます。新しいバージョンのプログラムがあり、マーケティング担当チームがその新しい機能のプロモーションに熱心に取り組んでいるとします。この新しい機能を[ホーム] タブ上に配置することを考えるかもしれません、そうすることで全体の見つけやすさが損なわれるのであれば、その過ちの代償は大きなものとなります。このことが製品の将来のバージョンに与える影響や、頻繁に構成を変更することによるストレスの大きさについて考えてみる必要があります。

## タブ

最初のステップで、「標準的なリボンタブ」について確認することをお勧めします。プログラムのコマンドが標準的なタブに既に対応している場合は、この標準に基づいてタブを構成します。一方、プログラムのコマンドが対応していない場合は、無理に対応させないでください。より適切な構造を見つけて、完成するまでユーザー テストを何度も実行します。

標準でないタブの場合は、以下の点について考慮します。

- 各タブ名がコンテンツを説明している。適度に具体的で、適切な名前を選択します。ユーザーがコンテンツを見て違和感を持たないようにしてください。
- 各タブ名が目的を反映している。コマンドに関連する目的またはタスクを考慮して名前を設定します。
- 各タブ名はその他のすべてのタブ名と明確に区別できる。

[ホーム] タブは、上記の考慮事項から除外されます。[ホーム] タブは必須ではありませんが、ほとんどのプログラムで必要です。[ホーム] タブは最初のタブで、最もよく使用されるコマンドが含まれています。よく使用されるコマンドがあり、他のどのタブにも該当しない場合は、[ホーム] タブが最適な配置場所になります。

適切で、説明的なタブ名を決められない場合、タブのデザインが適切でない可能性があります。リボンの構成が適切でない場合は、タブのデザインを見直す必要があります。

## グループ

コマンドをグループに分類して、コマンドを関連するセットに整理します。グループ ラベルはコマンドに共通する目的を説明するものです。

グループとその提示方法を決定するにあたって、以下のようなさまざまな判断要素があります。

- 標準的なグループ化。コマンドはプログラムによって大きく異なりますが、多くのプログラム間で共通する標準的なグループがあります。これらのコマンドを同じ名前で同じような場所に提示すると見つけやすくなります。

正しい例: 間違った例:



間違った例では、2つの標準的なグループのコマンドが1つの標準的でないグループに結合されています。

- 細分化。ある程度の体系化は必要ですが、体系化しすぎるとコマンドを見つけにくくなります。グループ名が汎用的な場合、細分化が十分でない可能性があります。グループごとにコマンドが1つまたは2つのみの場合は、細分化しすぎている可能性があります(ただし、グループ内にその他のコマンドがないリボン内ギャラリーは問題ありません)。

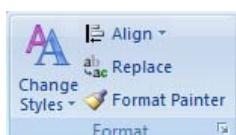
正しい例: 間違った例:



間違った例では、コマンドが1つまたは2つのみのグループがあり、体系化しすぎています。

- 名前。適切なグループ名を付けることで、グループに含まれるコマンドの目的を説明できます。グループ名でコマンドの目的を説明できていない場合は、名前またはグループ化について再検討します。

間違った例:

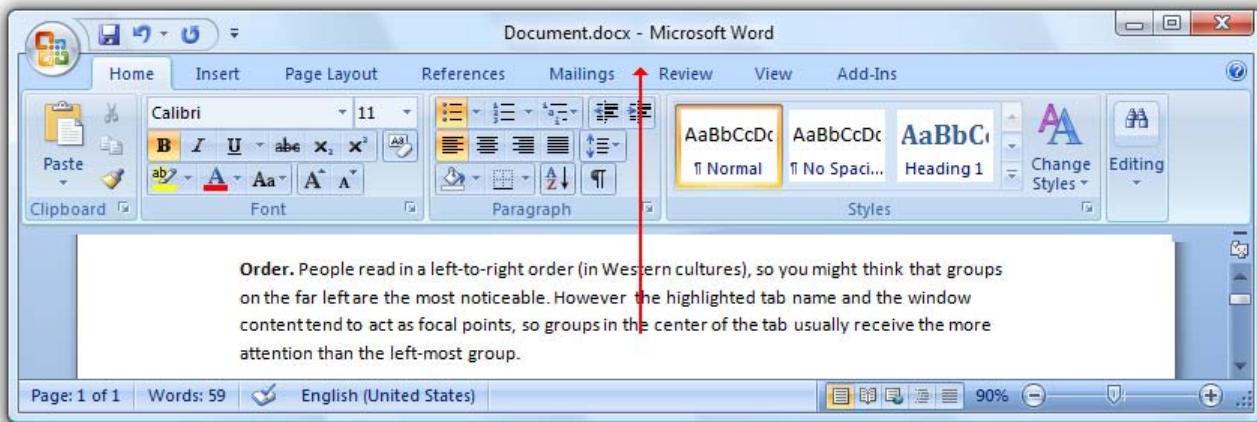


正しい例:

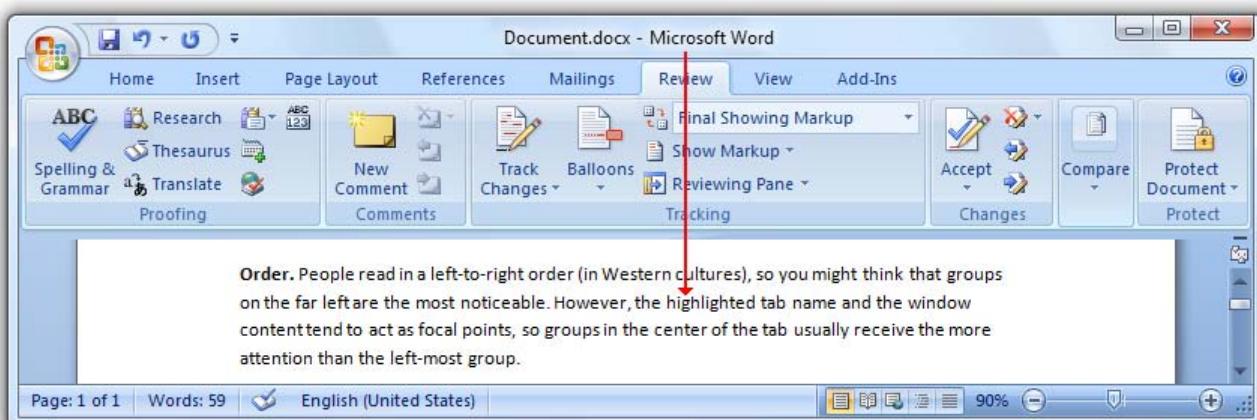


間違った例では、グループ名があいまいすぎて役に立ちません。この場合、これらのコマンドを具体的なグループに整理し直す必要があります。

- 順序。人は左から右に向って読むので(西欧文化圏の場合)、左端のグループが最も目立つと考えるかもしれません。ところが、強調表示されたタブ名やウインドウ コンテンツは、[フォーカル ポイント](#)としての機能を果たす傾向があり、通常、タブの中央のグループの方が左端のグループよりも目立ちます。最もよく使用するグループを一番目立つ場所に配置し、論理的な流れに沿ってグループがタブに配置されていることを確認します。



この例では、[フォント] グループおよび[段落] グループの方が、ドキュメントから目線を上に移動する  
と最初に目に付くので、[クリップボード] グループよりも目立っています。



この例では、[変更履歴] グループが一番目立っています。これは、[校閲] タブがフォーカル ポイントとしての機能を果たしているからです。

- 均一性。コマンドの提示がすべて同じように見えると、コマンドの区別が付きにくくなります。異なる形状や色のアイコン、書式設定に変化があるグループ、異なるサイズのコマンドを使用すると、ユーザーはコマンド グループを簡単に区別できます。リボンを小さいサイズに縮小するときのみ、コマンドのサイズ変更を均一化する必要があります。

正しい例:



間違った例では、コマンドはすべて同じサイズなので、コマンドの外観が互いに類似しています。

適切で、説明的な名前を決められない場合、グループのデザインが適切でない可能性があります。

#### プレビュー

さまざまな種類のプレビューを使用して、コマンドの実行結果を表示できます。適切なプレビューを使用すると、プログラムの効率を向上させ、試行錯誤による習得方法に頼る必要が少なくなります。また、リアルタイムのプレビューによって実際に試してみると、創造性が促進されます。

次のような、いくつかの種類のプレビューが利用可能です。

- 実際の効果を示す静的アイコンおよびグラフィック。コマンドの効果を実際に示す静的なイメージです。これらは、ギャラリー、ドロップダウンメニュー、拡張ツールヒントに使用できます。



この例では、[フォント] ドロップダウンリストで、フォントそのものを使用してフォント名を表示しています。



この例では、実際のイメージに基づいたサムネイルを使用して各種の透かしを表示しています。

- 動的なアイコンおよびグラフィック。現在の状態を反映して変更されるアイコンおよびグラフィックです。このようなアイコンは、特にギャラリーで有用なほか、既定の効果が直前の操作に変更される分割ボタンにも役に立ちます。

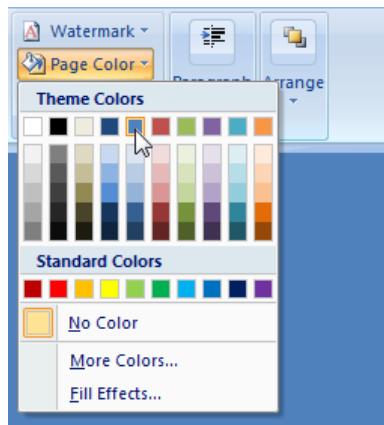


この例では、Microsoft Word で現在のスタイルを反映するためにスタイルギャラリーが変更されています。



この例では、Word で現在の効果を示すために [蛍光ペンの色] および [フォントの色] コマンドが変更されています。

- リアルタイムのプレビュー。ユーザーが書式設定オプションにマウス ポインターを合わせると、リアルタイムのプレビューにより、その書式設定を選択した場合にどのように見えるかが表示されます。リアルタイムのプレビューによって、ユーザーは実際のコンテキストに基づいて効率よく、迷うことなく選択できます。

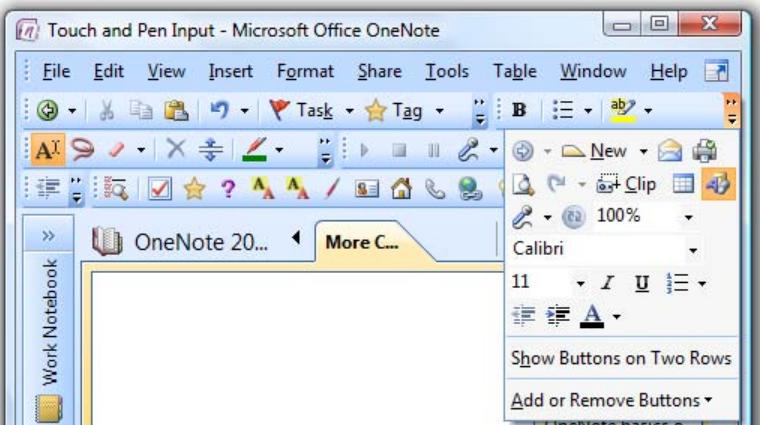


この例では、[ページの色] コマンドで色のオプションをポイントしたときの効果を示すリアルタイムのプレビューが実行されています。

リアルタイムのプレビューは、ユーザーの生産性を実際に向上させることができる優れた機能ですが、シンプルなスタティック プレビューでも役立ちます。

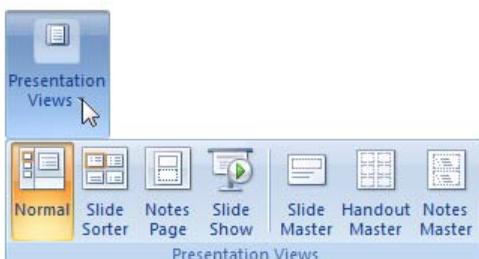
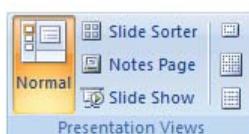
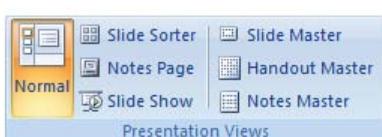
#### リボンの拡大縮小

ツールバーの拡大縮小は単純です。ウィンドウの幅が狭くツールバーを表示できない場合、画面に収まる要素はツールバーに表示し、他のものはすべてオーバーフロー ボタンを使用してアクセスできるようにします。



ツールバーは、オーバーフロー ボタンを使用して拡大縮小を行います。

リッチ コマンドの目的は、使用可能な領域を最大限に活用することです。したがって、リボンの拡大縮小には詳細なデザイン作業が必要になります。リボンの既定サイズはないので、特定の幅を想定してリボンをデザインしないようにします。さまざまな幅のレイアウトをデザインし、どの幅のレイアウトもほとんどのユーザーが目に見えるものとして認識する必要があります。拡大縮小はリボン デザインの基本的な部分であり、最終ステップではありません。



リボンには既定サイズはありません。一番小さいサイズは、単一のポップアップ グループ アイコンです。

タブをデザインする際は、グループごとに異なるレイアウト(3つまで)と、一緒に使用できる組み合わせも指定します。リボンは、現在のウィンドウ サイズに収まる最大の、有効な組み合わせを表示します。

#### 7つの重要な点

1. プログラムの種類に適したコマンドソリューションを選択します。リボンを使用する場合は、プログラムの操作を簡単にし、効率を高め、使いやすくなるようにして、逆効果にならないようにします。リボンの使用が適切でない場合は、代わりにリッチ コマンドの使用を検討します。
2. 効率的なリボンを簡単に作成できるとは考えないでください。リボンの作成は、既存のメニューとツールバーを単純に移植するだけではありません。また、リボンを使用すれば良いプログラムができる場合とも考えないでください。コマンドの再デザインに必要な時間と労力を注ぐ意思があるかどうかは、リボンの使用が適切かどうかを判断する重要な要素になります。
3. コマンドを見つけやすくします。配置されているコマンドと、説明的なラベルを付けたタブが明確で一意に対応しているタブのデザインを選択します。必要なコマンドが配置されているタブを、すばやく、迷わずに判断でき、間違ったタブを選択することがほとんどないようにする必要があります。
4. 効果が一目でわかるコマンドにします。ユーザーがコマンドのラベル、アイコン、ツールヒント、プレビューからコマンドの効果を理解できるようにする必要があります。ユーザーがコマンドのしくみについて知るために試行錯誤したり、ヘルプのトピックを参照することがないようにします。
5. 効率的なコマンドにするには、以下の点に留意します。
  - ユーザーが [ホーム] タブでほとんどの作業を行えるようにする。

- ユーザーが一般的なタスクを実行している間はタブを変更する必要がないようにする。
  - ウインドウが最大化され、ユーザーが正しいタブを選択している場合、最もよく使用するコマンドが一番視覚的に強調され、1回のクリックでコマンドを呼び出せるようにする。ユーザーがタブ上にあるその他のすべてのコマンドを多くても4回のクリックで実行できるようにする。
  - ユーザーがダイアログボックスを開いてコマンドを入力したり、一般的なタスクの属性を変更することができないようにする。
6. ユーザーがコマンドやオプションを迷わず選択できるようにし、試行錯誤する必要性を最小限に抑えます。適切な場合は常に、結果指向のコマンドを使用します。多くの場合、ギャラリーおよびリアルタイムのプレビュー形式を使用します。
7. 最大のウインドウ サイズから最小のウインドウ サイズまで、リボンが適切に拡大縮小されることを確認します。

## ガイドライン

### 全般

- リボンをウインドウ内でメニュー バーおよびツール バーと組み合わせないようにします。リボンはメニュー バーおよびツール バーの代わりに使用する必要があります。ただし、パレット ウインドウや、[進む] ボタンおよび[戻る] ボタン、アドレス バーなどのナビゲーション要素とリボンを組み合わせることができます。
- 常に、リボンをアプリケーション ボタンおよびクリック アクセス ツール バーと組み合わせます。
- プログラム起動時に左端のタブ(通常は[ホーム]タブ)を選択します。最後に選択したタブをプログラムインスタンス間で保持しないでください。
- プログラムの初回起動時にリボンを通常の状態(最小化されていない状態)で表示します。ユーザーは既定の設定を変更しないことが多いので、プログラム起動時にリボンが最小化されていると、すべてのコマンドに対する効率が下がる可能性があります。また、リボンを最初に最小化して表示すると、ユーザーの混乱を招きます。
- リボンの状態をプログラムインスタンス間で保持します。たとえば、ユーザーがリボンを最小化した場合、次回プログラムが実行されるときにリボンを最小化した状態で表示する必要があります。ただし、繰り返しになりますが、最後に選択したタブをこのように保持しないでください。

### タブ

- 実用的である場合は常に、標準的なタブを使用するようにします。標準的なタブを使用すると、特に複数のプログラムを使用するときに見つけやすさが大幅に向上します。後の「[標準的なリボンタブ](#)」を参照してください。
- 適切な場合は、最初のタブを[ホーム]とラベル付けします。[ホーム]タブには最もよく使用されるコマンドが含まれている必要があります。よく使用されるコマンドがあり、他のどのタブにも該当しない場合は、[ホーム]タブが最適な配置場所になります。
- 以下の場合、新しいタブを追加します。
  - コマンドが特定のタスクと密接に関連していて、タブのラベルで正確に説明できる。タブを追加することで、コマンドを見つけにくくするのではなく、見つけやすくする必要があります。
  - コマンドがその他のタブのタスクとほとんど関連していない。タブを追加することで、一般的なタスクを実行しているときに必要以上にタブを切り替えることがないようにします。
  - 表示する別の場所を追加する必要があるほど、タブに含まれるコマンドが多い。コマンドの数が少ないタブを追加しないでください。次の場合は例外です。
    - コマンドが特定のタスクと密接に関連していて、タブを追加することで、複雑になりすぎた[ホーム]タブが大幅に簡素化される場合は、配置するコマンドの数が少なくともタブを追加することを検討します。

一般的に、タブの数は少ないほど良いので、上記の目的を実現するうえで役に立たないタブは削除します。

- 残りのタブについては、タブ間の論理的な順番を維持したうえで、最もよく使用されるタブを最初に配置します。
- ユーザーがすばやく、迷わずにコマンドを見つけることができるようタブのデザインを最適化します。その他の考慮事項はすべて補助的なものです。
- [ヘルプ]タブを配置しないようにします。その代わりに、プログラム全体のヘルプと拡張ツールヒントを使用してアシスタンスを提供します。
- 使用するコアタブは最大7つにします。コアタブが8つ以上あると、どのタブにどのコマンドがあるかを判断しにくくなります。コマンドが多いアプリケーションの場合、コアタブが7つあっても問題ありませんが、ほとんどのプログラムでは、タブの数を4つ以下に収めることを目標にする必要があります。

タブのラベル付けのガイドラインについては、「[タブラベル](#)」を参照してください。

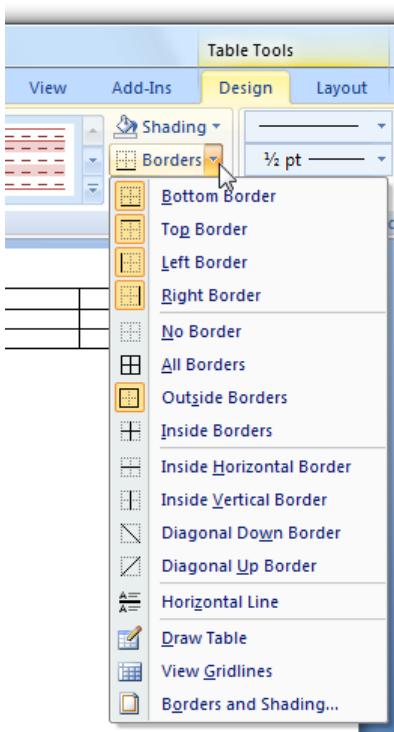
### コンテキストタブ

- ユーザーが特定のオブジェクトの種類を選択したときのみ、関連するコマンドのコレクションを表示するには、コンテキストタブを使用します。数が少なく、よく使用するコマンドの場合は通常のタブを使用し、適用されない場合はコマンドを無効にすると、便利で一貫性が得られます。



[切り取り]や[コピー]などの一般的なコマンドは、コンテキストタブを使用するのではなく無効にするようにします。

- 特定のオブジェクトの種類に特有のコマンドのみを含めます。オブジェクトを選択してからでなくとも使用できるコマンドの場合は、コンテキストタブ上だけにコマンドを配置しないでください。
- 特定の種類のオブジェクトでの作業によく使用されるコマンドを含めます。よく使用する一般的なコンテキスト依存コマンドをコンテキストメニューおよびミニツールバーに配置して、一般的なタスクを実行している間はタブの切り替えを避けるようにします。また、一般的なコマンドを重複してコンテキストタブに配置したときに、頻繁なタブの切り替えを避けることができる場合は、そのようにします。ただし、過度なコマンドの追加は推奨しません。ユーザーがオブジェクトで作業しているときに必要なコマンドをすべて含めることはしないでください。

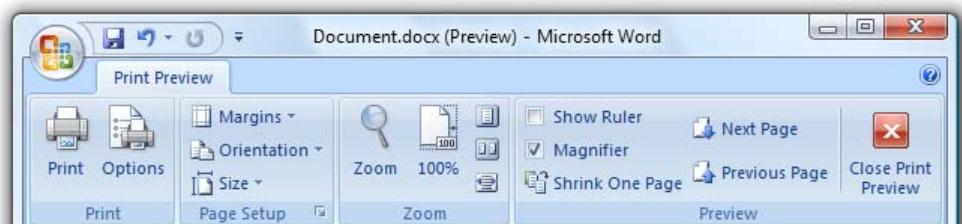


この例では、[罫線] コマンドは [デザイン] タブに含まれており、一般的なタスクを実行している間は頻繁にタブを切り替える必要がありません。

- 現在表示されているコンテキストタブとは異なるコンテキストタブの色を選択します。これを実現するために、同じタブセットを後で別の色を使用して表示できます。ただし、可能な限り、呼び出すたびに一貫した色の割り当てを使用します。
  - 以下の場合は、コンテキストタブを自動的に選択します。
    - ユーザーがオブジェクトを挿入した場合。この場合は、セットの最初のコンテキストタブを選択します。
    - ユーザーがオブジェクトをダブルクリックした場合。この場合は、セットの最初のコンテキストタブを選択します。
    - ユーザーがコンテキストタブを選択し、オブジェクトをクリックして解除してからすぐに同じ種類のオブジェクトをクリックした場合。この場合は、前回選択されたコンテキストタブに戻ります。
- こうすることによってタブを見つけやすくなり、表示の一貫性が保たれ、タブを切り替える必要が少なくなります。ただし、その他の状況では、コンテキストタブを自動的に選択せずにユーザーに操作を任せます。
- アクティブなタブであるコンテキストタブを削除する場合は、[ホーム] タブまたは最初のタブをアクティブなタブにします。こうすると、最も表示の一貫性が保たれます。

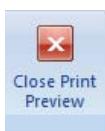
#### モーダルタブ

- 特定の "一時的な" モードを適用したコマンドのコレクションを表示し、コアタブを適用しないようにするには、モーダルタブを使用します。コアタブの一部を適用する場合は、代わりにコンテキストタブを使用し、適用されないコマンドを無効にします。モーダルタブは限定的なので、他に良い手段がない場合のみ使用するようにします。



印刷プレビューは一般的に使用されるモーダルタブです。

- モーダルタブを閉じるために、タブ上の最後のコマンドとして [<モード> を閉じる] コマンドを配置します。[閉じる] アイコンを使用して、コマンドを見つけやすくなります。閉じる対象について混乱が生じないように、コマンドにモードを適用します。



この例では、[閉じる] コマンドをモードで明示的にラベル付けしているので、閉じる対象は明らかです。

- モーダルタブを閉じるために、ウィンドウのタイトルバー上の [閉じる] ボタンを、プログラムではなくモードを閉じるように再定義します。ユーザーテストによって、多くのユーザーがこの操作を想定していることがわかりました。

#### 標準的なリボンタブ

実用的である場合は常に、プログラムのコマンドをこれらの標準的なタブに対応させ、標準的な表示順序で配置するようにします。

## 通常のタブ

- ホーム。最もよく使用されるコマンドが含まれています。使用する場合は常に最初のタブとなります。
- 挿入。コンテンツおよびオブジェクトをドキュメントに挿入するコマンドが含まれています。使用する場合は常に2番目のタブとなります。
- ページレイアウト。テーマ、ページ設定、ページの背景、インデント、間隔、配置など、ページレイアウトに影響するコマンドが含まれています。領域にゆとりがある場合は、インデントグループと間隔グループを[ホーム]タブに配置することもできます。使用する場合は常に3番目のタブとなります。
- 校閲。コメントの挿入、変更履歴の記録、およびバージョンの比較を実行するコマンドが含まれています。
- 表示。表示モード、オプションの表示/非表示、ズーム、ウィンドウの管理、マクロなど、ドキュメントの表示に影響するコマンド(従来の[ウィンドウ]メニューのカテゴリにあるコマンド)が含まれています。使用する場合は、[開発]タブを表示しない限り、通常のタブの最後に表示します。
- 開発。開発者のみが使用するコマンドが含まれています。使用する場合は、既定で非表示にし、表示する場合は通常のタブの最後になります。

ほとんどのプログラムでは、[校閲]タブおよび[開発]タブは必要ありません。

## コンテキストタブ

- 書式設定。選択したオブジェクトの種類の書式設定の変更に関するコマンドが含まれています。通常、オブジェクトの部分に適用されます。
- デザイン。選択したオブジェクトの種類にスタイルを適用するコマンド(多くの場合はギャラリー内)が含まれています。通常、オブジェクト全体に適用されます。
- レイアウト。表やグラフなど、複雑なオブジェクトの構造を変更するコマンドが含まれています。

書式設定、デザイン、レイアウトに関するコンテキスト依存コマンドがあり、複数のタブが必要になるほどコマンドの数が多くない場合は、[書式設定]タブに配置します。

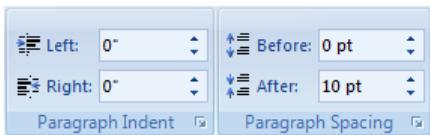
## グループ

- 実用的である場合は常に、標準的なグループを使用するようにします。コマンドを同じ名前で同じような場所に表示すると、見つけやすさが大幅に向上します。後の「[標準的なリボングループ](#)」を参照してください。
- 以下の場合、新しいグループを追加します。
  - コマンドが密接に関連していて、グループのラベルで正確に説明できる。グループを追加することで、コマンドを見つけにくくするのではなく、見つけやすくなる必要があります。
  - コマンドとその他のグループのコマンドとの関連性が低い。1つのタブのコマンドはすべて密接に関連している必要がありますが、一部のコマンドの関連性は他よりも密接です。
  - 表示する別の場所を追加する必要があるほど、グループに含まれるコマンドが多い。グループのコマンドの数を通常は3~5つに収めることを目指します。コマンドが1つまたは2つのみのグループを配置することは避けます。ただし、グループ内にその他のコマンドがないリボン内ギャラリーを配置するのは問題ありません。コマンドが1つだけ含まれているグループを多く配置すると、過剰な構造になったり、コマンドの結びつきがなくなったりします。

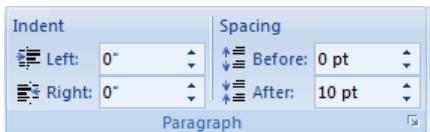
unnecessary にグループを追加して構成が過剰にならないようにします。

- 以下の場合、グループを分割することを検討します。
  - ラベルを追加すると大きなメリットが得られるコマンドがグループにある場合。たとえば、コマンドの明確化が必要な場合や、ラベルでテキストが繰り返されている場合はグループを分割します。

正しい例:



より良い例:



より良い例では、ラベルを追加することでグループのコマンドがわかりやすくなり、個別の長いグループ名よりも、1つの短いグループ名の方が適しています。

- グループにさまざまなサイズのコマンドが多数あり、整理が必要な場合。



この例では、さまざまなサイズのコマンドが多数あります。

- グループをほぼ均一な2~3個のグループに分割できる場合。
- 個別のグループを使用するよりも分割グループを使用すると内容がわかりやすくなる、または、不自然さが軽減される場合。
- 最もよく使用するグループを一番目立つ場所に配置し、論理的な順序でグループがタブ全体に配置されていることを確認します。
- ユーザーがすばやく、迷わずにコマンドを見つけることができるようグループのデザインを最適化します。その他の考慮事項はすべて補助的なものです。
- 1つのボタンのみが含まれているグループをポップアップグループアイコンに縮小しないようにします。縮小する場合は、1つのボタンとして残します。

ます。

- 使用するグループは最大 7 つにします。グループが 8 つ以上あると、どのタブにどのコマンドがあるかを判断しにくくなります。

グループ ラベルのガイドラインについては、「[グループ ラベル](#)」を参照してください。

#### 標準的なリボン グループ

実用的である場合は常に、プログラムのコマンドをこれらの標準的なグループに対応させ、標準的な表示順序で関連するタブ内に配置するようにします。

#### [メイン] タブ

- クリップボード
- フォント
- 段落
- 編集

#### [挿入] タブ

- 表
- 図

#### [ページ レイアウト] タブ

- テーマ
- ページ設定
- 配置

#### [校閲] タブ

- 文章校正
- コメント

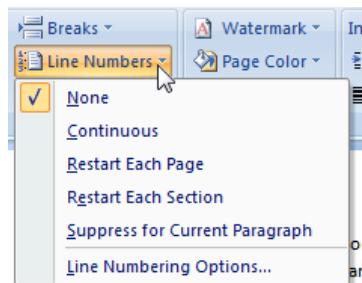
#### [表示] タブ

- 文書の表示
- 表示/非表示
- ズーム
- ウィンドウ

#### コマンド

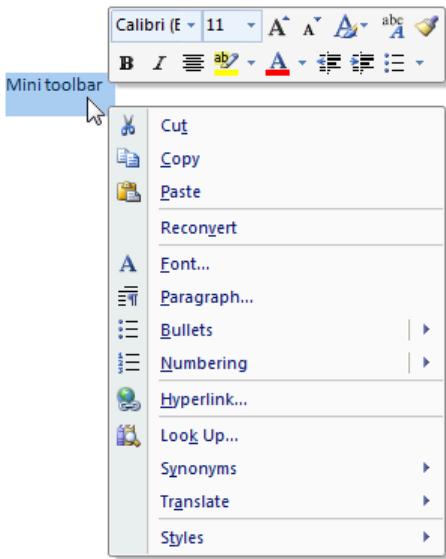
##### 全般

- よく使用するコマンドをすべて表示して、リボンの持つ見つけやすさと拡張性を活用します。適切な場合は、よく使用するコマンド(特に見つけにくいコマンド)をダイアログ ボックスからリボンに移動します。理想的には、ユーザーがダイアログ ボックスを使用しなくても一般的なタスクを実行できるようにします。



この例では、行番号の設定はこれまでプロパティ シートに埋もれていきました。この設定をリボンに配置することで、見つけやすくなっています。

- リボンの拡張性を利用して不必要に複雑な構成にすることは避けます。常に控えめを心がけます。できるからという理由だけでコマンドをリボンに追加しないでください。コマンド全体の操作性をシンプルに保ちます。提示を簡素化する方法を次に示します。
  - インプレース コマンドやコンテキスト依存コマンドには、コンテキストメニューおよび[ミニ ツールバー](#)を使用します。



この例では、コンテキストメニューおよびミニ ツールバーはコンテキスト依存コマンドをインプレースで表示しています。

- ほとんど使用しないコマンドはダイアログ ボックスに移動(保持)します。これらのコマンドにアクセスするには、ダイアログ ボックス起動ツールを使用します。ダイアログ ボックスはリボンでも使用できます。一般的なタスクを実行している間は、ダイアログ ボックスを使用する必要が少なくなるようにしてください。
- 重複し、ほとんど使用しない機能を削除します。

#### 提示方法

- 各コマンドを1つのタブだけに提示します。複数の方法で同じコマンドにアクセスすることは避けます。特に、コマンドを呼び出すために何度もクリックする必要がある場合は注意します。複数の方法があると、コマンドを見つけるのに便利だと感じます。しかし、ユーザーは必要なコマンドが見つかったらそれ以上は探さないことに注意してください。一般的にユーザーは、最初に見つけた方法が唯一の方法であると考えます。これは、方法が非効率的な場合、深刻な問題になります。
  - 例外: コンテキスト タブでは、[ホーム] タブおよび [挿入] タブのコマンドのいくつかを重複して使用することができます。ただし、一般的なコンテキスト タスクでタブを変更する必要がなくなる場合に限ります。
- グループ内では、よく使用されるコマンドを優先的に配置したうえで、コマンドを論理的な順序で配置します。全体的に、コマンドは見つけやすいように論理的な流れに沿って配置する必要があります。最もよく使用されるコマンドは最初に表示します。一般的に、32×32ピクセルアイコンのコマンドを16×16ピクセルアイコンのコマンドの前に表示して、各グループを探しやすくなります。
- リスクのあるコマンドを使用頻度の高いコマンドの隣に配置しないようにします。効果が広範囲に及び、簡単に戻せなかつたり、すぐにその効果に気付かない場合、コマンドはリスクのあるものと見なされます。
- 相互に排他的なオプションのセットなど、密接に関連するコマンドを示す場合は、区切り記号を使用します。
- 密接に関連しており、よく知られているラベルを必要としないコマンドのセットには、ツールバー スタイルのグループを使用することを検討します。こうすることで、見つけやすさや習得のしやすさに影響することなく、多くのコマンドをコンパクトな領域に提示できます。よく知られているコマンドは使用頻度が高く、すぐに認識できるので、[ホーム] タブに配置される傾向にあります。



この例では、密接に関連しており、よく知られているコマンドにツールバー スタイルのグループを使用しています。

- 最もよく使用されるコマンドおよび重要なラベルが付けられているコマンドには、32×32ピクセルのアイコンを使用します。グループのサイズを縮小する場合、これらのコマンドを16×16ピクセルのアイコンに変換するのは最後にします。
- 根拠のないコマンドの配置は避けます。ユーザーが必要なコマンドを見つけるためにすべてのタブを確認して時間を無駄にすることがないように、タブおよびグループのデザインについて慎重に考えてください。
- マーケティングに基づいた配置は避けます。新しい機能のプロモーションに関するマーケティングの目的は、時間の経過と共に変化する傾向があります。このことが製品の将来のバージョンに与える影響や、頻繁に構成を変更することによるストレスの大きさについて考えてみる必要があります。

#### 対話操作

- 現在のコンテキストに適用されないコマンド、または直接エラーを引き起こすコマンドを無効にします。有用な場合は、[拡張ツールヒント](#)を使用して、コマンドが無効になっている理由を説明します。こうしたコマンドは非表示にしないでください。非表示にすることにより、リボンのレイアウトが変更され、リボンの提示に一貫性がなくなります。
- コマンドのラベルを動的に更新しないようにします。繰り返しになりますが、ラベルを動的に更新するとタブのレイアウトが変更され、外観に一貫性がなくなります。このため、定型ラベルに合うようにコマンドをデザインします。

正しい例:

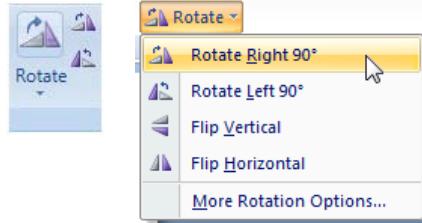
間違った例:



[メモの挿入]と[メモの削除]は同時に有効になることはありませんが、一貫性のあるリボンの提示には、両方のコマンドを表示する必要があります。

- 直接コントロールを優先します。1回のクリックで呼び出される(複数のメニューを開く必要がない)コマンドは直接的です。ただし、リボン内ギャラリーを除いて、直接コントロールはリアルタイムのプレビューをサポートしていないので、リアルタイムのプレビューが必要かどうかを判断要素になります。
  - 関連する書式設定オプションのセットにコマンドが含まれ、リアルタイムのプレビューが重要で実用的な場合(特に、プレビューがないとユーザーが不適切なオプションを選択する可能性が高い場合)、リアルタイムのプレビューを使用して、オプションの効果を示します。
    - そのコマンドがよく使用される場合は、リボン内ギャラリーを使用して直接性を保持します。
    - そのコマンドが頻繁に使用されない場合は、ドロップダウンギャラリーを使用します。

正しい例: より良い例:



正しい例は直接的であるのに対して、より良い例はリアルタイムのプレビューをサポートしています。

- 上記以外の場合、直接的な効果を得るには、リボンコントロールを以下の優先順位で使用します(その他の考慮事項はすべて同等とします)。
  - コマンドボタン、チェックボックス、ラジオボタン、インプレースギャラリー。これらは常に直接的です。
  - 分割ボタン。最も一般的なコマンドに対して直接的ですが、コマンドのバリエーションに対しては直接的ではありません。
  - メニュー ボタン。これらは直接的ではありませんが、見つけやすいコマンドを多数提示します。
  - テキストボックス(スピントリール付き)。テキスト入力は一般的に他の種類のコントロールよりも手間がかかります。

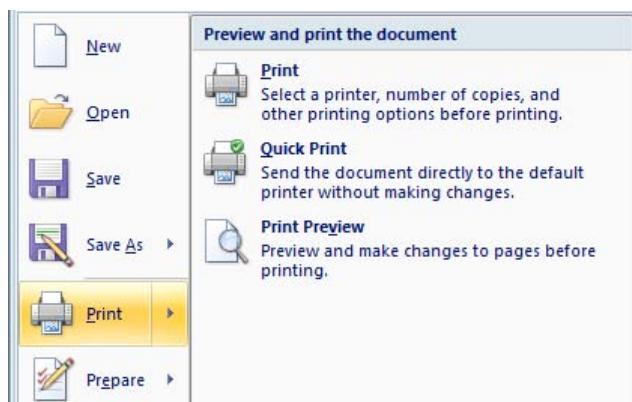
リボンをフルサイズで表示するときに、リボンが主にメニュー ボタンで構成されている場合は、メニュー バーを使用することができます。

間違った例:



このリボンにはメニュー ボタンが多いので、メニュー バーを使用した方がよいと考えられます。

- 即時型のコマンドを優先します。即座に効果が現れる(追加入力のためのダイアログボックスが表示されない)コマンドは即時型です。コマンドに入力が必要な場合は、分割ボタンを使用して、ボタンの部分に即時型のコマンドを配置し、サブメニューで入力が必要なコマンドを配置することを検討します。



この例では、ボタンをクリックすると、すぐに既定のプリンターでコピーが1部印刷され、サブメニューから選択すると[印刷オプション]ダイアログボックスが表示されます。

コマンドラベルのガイドラインについては、「[コマンドラベル](#)」を参照してください。特定のコマンドコントロールのガイドラインについては、関連する「[コントロールのガイドライン](#)」を参照してください。

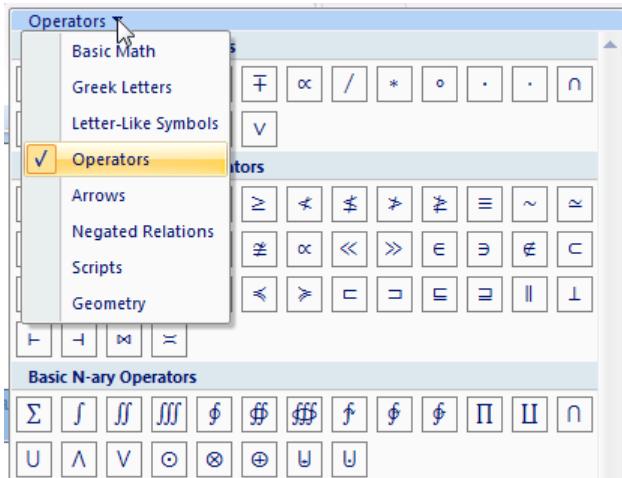
## ギャラリー

- 以下の場合に、[ギャラリー](#)を使用します。
  - 明確に定義された、関連する選択肢のセットがあり、ユーザーは通常その選択肢から選択する場合。バリエーションの数は無制限ですが、選択される可能性が高い選択肢が適切に含まれている必要があります。選択肢が密接に関連していない場合は、個別のギャラリーの使用を検討します。

- 書式設定に関する機能など、選択肢が視覚的に適切に表されている場合。サムネイルを使用すると、参照、理解、選択が簡単になります。選択肢にラベルを付けることもできますが、選択が視覚的に行われ、選択肢を理解するためのテキストラベルが必要ないようにします。
- 選択肢により、1回のクリックで直ちに実行される結果が示される場合。ユーザーの意図を明確にするための追加のダイアログボックスや、表示された結果を実現するための一連のステップが必要とならないようにします。ユーザーが選択を調整する必要がある場合は、後で調整できるようにします。

ギャラリーを使用して、グループ内で通常のコマンドを多数表示しないでください。

- 以下の場合に、リボン内ギャラリーを使用します。
  - 選択肢が頻繁に使用されている場合。こうした選択肢は領域を必要とし、他のコマンドの領域を消費する潜在的価値があります。
  - 標準的な用途で、提示された選択肢のグループ化またはフィルタリングを行う必要がない場合。
  - リボンの高さの範囲内(48ピクセル)で選択肢を効率的に表示できる場合。
- 目的を果たすことができる標準的なギャラリー サムネイルの一番小さいサイズを選択します。
  - リボン内ギャラリーの場合、 $16 \times 16$ 、 $48 \times 48$ 、または $64 \times 48$ ピクセルのサムネイルを使用します。
  - ドロップダウンギャラリーの場合、 $16 \times 16$ 、 $32 \times 32$ 、 $48 \times 48$ 、 $64 \times 48$ 、 $72 \times 96$ 、 $96 \times 72$ 、 $96 \times 96$ 、または $128 \times 128$ ピクセルのサムネイルを使用します。
  - ギャラリー項目のサムネイル サイズはすべて同じにする必要があります。
- リボン内ギャラリーの場合は、以下の点に留意します。
  - 少なくとも3つ(領域がある場合はそれ以上)の選択肢を表示します。通常のウィンドウ サイズで、少なくとも3つの選択肢を表示する十分な領域がない場合は、代わりにドロップダウンギャラリーを使用します。
  - 使用可能な領域を活用するには、リボン内ギャラリーを拡張します。追加の領域を使用して、より多くの項目を表示し、1回のクリックで選択しやすいようにします。
- ドロップダウンギャラリーの場合は、以下の点に留意します。
  - コンボ ボックス、ドロップダウンリスト、分割ボタン、またはメニュー ボタンからギャラリーを表示します。
  - ユーザーがメイン ウィンドウをクリックしてドロップダウンギャラリーを閉じる場合は、メイン ウィンドウのコンテンツを選択したり変更したりせずにギャラリーを閉じます。
  - ギャラリーに選択肢が多数あり、一部の選択肢をほとんど使用しない場合は、よく使用する選択肢を中心的に扱い、既定のギャラリーを簡素化します。残りのコマンドについては、ギャラリー ドロップダウンの下部に適切なコマンドを配置します。
    - コマンドに多くのバリエーションの一覧が表示される場合は、「その他の<個別の機能名>オプション...」と名前を付けます。
    - コマンドでダイアログ ボックスを提示して、ユーザーが独自のカスタム オプションを作成できる場合は、「カスタム <機能名>...」と名前を付けます。
  - 選択肢をグループに整理すると参照が効率的になる場合は、そのようにします。
  - ギャラリーに選択肢が多数ある場合は、ユーザーが選択肢を効率的に見つけることができるようフィルターの追加を検討します。混乱を避けるために、最初はフィルターを設定せずにギャラリーを表示します。ただし、ほとんどのギャラリーでは多くの選択肢を配置することは望ましくないので、フィルターを使用する必要はなく、グループの使用で十分です。



この例では、グループとフィルターの両方を使用することでギャラリーが見やすくなっています。

## プレビュー

- プレビューを使用すると、ユーザーがコマンドを実行する前にコマンドの効果を示すことができます。有用なプレビューを使用すると、プログラムの効率と習得のしやすさを向上させ、試行錯誤の習得方法に頼る必要が少くなります。異なる種類のコマンド プレビューについては、このトピックの「デザイン コンセプト」に記載されている「[プレビュー](#)」を参照してください。
- リアルタイムのプレビューの場合、500ミリ秒以内にプレビューが適用され、現在の状態に復元できることを確認します。これには、書式設定の変更を即座に、割り込み可能な方法で適用できる機能が必要です。ユーザーがリアルタイムのプレビューのメリットを最大限に活用して、さまざまなオプションをすばやく評価できるようにする必要があります。
- プレビューでのテキストの使用は避けます。テキストを使用する場合は、プレビューイメージのローカライズが必要になります。

## アイコン

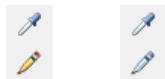
- ドロップダウンリスト、チェック ボックス、ラジオ ボタンを除き、すべてのリボン コントロールにアイコンを使用します。ほとんどのコマンドには $32 \times 32$ ピクセルと $16 \times 16$ ピクセルの両方のアイコンが必要です(クイック アクセス ツールバーでは、 $16 \times 16$ ピクセルのアイコンのみ使用されます)。ギャラリーでは、通常、 $16 \times 16$ 、 $48 \times 48$ 、または $64 \times 48$ ピクセルのアイコンが使用されます。



ドロップダウンリスト、チェックボックス、ラジオボタンにはアイコンは必要ありませんが、他のすべてのリボンコントロールでは必要です。

- コマンドごとに固有のアイコンを使用します。別のコマンドに同じアイコンを使用しないようにします。
- リボンの背景色に対して、リボンアイコンがはっきりと表示されるようにします。リボンアイコンの評価は、常にコンテキスト内で、ハイコントラストモードで行います。
- 特に、よく使用されるコマンドの場合は、効果を明確に伝えるアイコンデザインを選択します。優れたデザインのリボンには内容が一目でわかるアイコンがあり、ユーザーは効率的にコマンドを見つけて理解することができます。
- 認識しやすく、区別しやすいアイコンを選択します(よく使用されるコマンドの場合は特に)。アイコンに特徴的な形状と色が使用されていることを確認します。そうすることで、ユーザーがアイコンのシンボルを記憶していくなくても簡単にコマンドを見つけることができます。

正しい例: 間違った例:



間違った例では、アイコンの色が類似しているので、区別がつきにくくなっています。

- グループで最も目立つコマンドの  $16 \times 16$  ピクセルのアイコンを  $32 \times 32$  ピクセルのビジュアルコンテナー内に配置して、ポップアップグループアイコンを作成することを検討します。ポップアップグループ用に別のアイコンを作成する必要はありません。



この例では、最も目立つコマンドの  $16 \times 16$  ピクセルのアイコンからポップアップグループアイコンが作成されています。

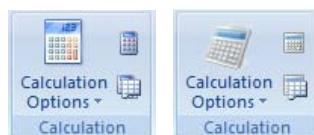
- 有益な場合は、現在の状態を反映するようにアイコンを変更します。この方法は、特に既定の効果が変更される分割ボタンで有用です。



この例では、Microsoft Word で現在の効果を示すために「蛍光ペンの色」および「フォントの色」コマンドが変更されています。

- リボンのアイコンを Aero style のアイコンのガイドラインに準拠させるようにします。ただし、リボンのアイコンは遠近法表示ではなく平面に表示します。

正しい例: 間違った例:



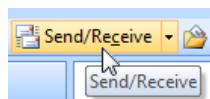
間違った例では、 $32 \times 32$  ピクセルのアイコンが遠近法で表示されています。

詳細と例については、「[アイコン](#)」を参照してください。

#### 拡張ツールヒント

- リボンコマンドすべてに拡張ツールヒントを実装し、コマンド名、ショートカットキー、説明、オプションの補足情報を表示する必要があります。ツールヒントで、ラベルと同じ内容を単純に繰り返さないようにします。

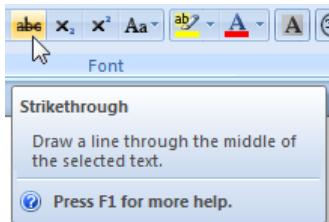
間違った例:



この例では、ツールヒントはコマンドラベルと同じ内容を単純に繰り返しているだけです。

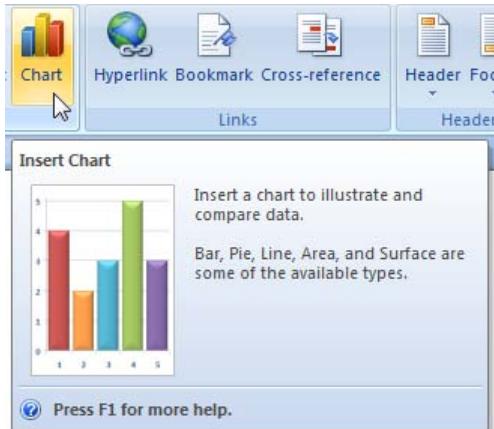
- 実用的な場合は、簡潔な説明を使用して、コマンドを適切に表現します。詳細な説明が本当に必要な場合のみ、ヘルプへのリンクを配置します。

間違った例:



この例では、コマンドにヘルプは必要ありません。

- 有用な場合は、プレビューを使用してコマンドの効果を示します。

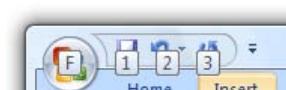


この例では、ツールヒントのイメージによりコマンドの効果を示しています。

ラベルのガイドラインについては、「[拡張ツールヒント ラベル](#)」を参照してください。

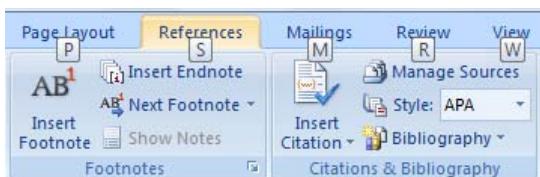
#### アクセス キーおよびキーヒント

- 注: キーヒントは、リボン上に直接表示されるコマンドのアクセス キーを表示するために使用されるメカニズムです(ドロップダウンメニュー コマンドのアクセス キーは下線付き文字で示されます)。キーヒントは以下の点で、メニュー アクセス キーとは異なります。
  - 2 文字のアクセス キーを使用できます。たとえば、FP を使用して、[書式のコピー/貼り付け] コマンドにアクセスできます。
  - アクセス キーの割り当ては、下線の代わりにヒントを使用して表示されるので、文字の幅やディセンダーは割り当てを行う際の判断要素になりません。
- すべてのリボンタブおよびリボンコマンドにアクセス キーを割り当てます。考えられる唯一の例外は、古いアドインのコマンドです。
- アプリケーション ボタンおよびクリック アクセス ツールバーの場合:
  - アプリケーション ボタンには F を割り当てます。この割り当てが使用される理由は、アプリケーション ボタンが従来の [ファイル] メニューに類似しているためです。
  - クリック アクセス ツールバーおよび最近使ったファイルの一覧には、数字でアクセス キーを割り当てます。



アプリケーション ボタンおよびクリック アクセス ツールバーのキーヒント

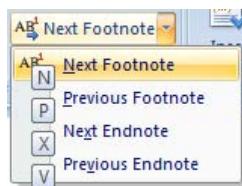
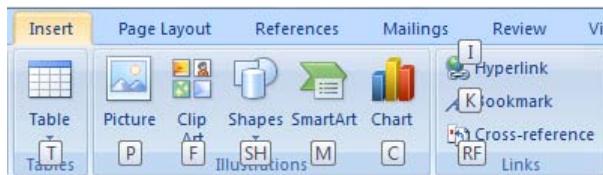
- タブの場合:
  - ホームには H を割り当てます。
  - 最もよく使用されるタブから始めて、ラベルの最初の文字を割り当てます。
  - 最初の文字を割り当てることができないタブについては、ラベルで特徴のある子音または母音を選択します。
  - メニュー バーのサポートに使用されるプログラムの場合は、実用性を最大限に考慮し、アクセス キーの互換性を維持するようにします。古いメニューのカタゴリとは異なる意味をアクセス キーに割り当てないようにします。たとえば、従来のメニュー バーのバージョンのプログラムに [編集] メニューがあった場合、それに相当するタブにはアクセス キーとして "E" を使用するようにします。同等のタブが存在しない場合は、混乱を避けるため、タブへのアクセス キーに E を割り当てないようにします。



タブのキーヒント

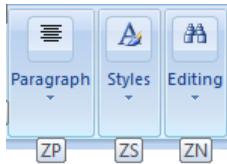
- リボン コマンド、メニュー、サブメニューの場合:
  - タブ内で、一意のアクセス キーの組み合わせを割り当てます。別のタブ内であれば、同じアクセス キーの組み合わせを使用できます。
  - 可能な限り、よく使用するコマンドに標準的なアクセス キーを割り当てます。「[標準的なアクセス キーの表](#)」を参照してください。
  - その他のコマンドの場合:
    - 最もよく使用されるコマンドの場合、ラベルの最初の文字か 2 番目の文字(できれば最初の文字)を選択します。
    - あまり頻繁に使用しないコマンドの場合、"Exit (終了)" の "x" など、ラベルで特徴のある子音または母音を選択します。

- ほとんど使用しないコマンドおよびダイアログ ボックス起動ツールの場合、必要に応じて 2 文字を使用します。
- メニューおよびサブメニューの場合は、1 文字を使用して、コマンドの完了に必要なキー入力の回数を減らします。
- J、Y、または Z で始まるアクセスキーは使用しません。これらは、コンテキストタブ、割り当てのないキーヒント、およびポップアップ グループに使用されます。



リボンおよびメニューのキーヒント

- ポップアップ グループの場合:
  - Z で始まる 2 文字のアクセスキーを使用します。
  - 最もよく使用されるグループから始めて、ラベルの最初の文字に 2 番目のアクセスキー文字を割り当てます。
  - 残りのグループについては、ラベルで特徴のある子音または母音を選択します。

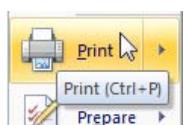


ポップアップ グループのキーヒント

ショートカットキーのガイドラインについては、「[キーボード](#)」を参照してください。

#### アプリケーション ボタン

- ファイルに対して何らかの操作を行うコマンドのメニューを提示するには、アプリケーション ボタンを使用します。例としては、従来、[ファイル] メニューに配置されるコマンド(ファイルの作成、ファイルを開く、ファイルの保存、ドキュメントの印刷、送信して公開するコマンド)などが挙げられます。
- リボンを使用する際は常にアプリケーション ボタンを配置します。プログラムでファイルを使用しない場合は、アプリケーション ボタンを使用してプログラムのオプションおよび[終了] コマンドにアクセスできるようにします。アプリケーション ボタンには常にコマンドメニューを表示します。アプリケーション ボタンは単なる飾りではありません。
- 適切な場合は、以下の標準的なアプリケーション メニュー コマンドを使用します。
  - 新規作成
  - 開く
  - 保存
  - 名前を付けて保存...
  - <ファイル形式の一覧>
  - <区切り記号>
  - 印刷...
  - クリック印刷
  - 印刷プレビュー
  - 閉じる
  - <フッター>
  - オプション
  - 終了
- アプリケーション メニューに属するコマンドは、そのメニューでのみ使用します。コマンドを重複して他のタブには配置しません。
- 各メニューの項目には以下を表示します。
  - コマンド名が含まれているラベル。
  - 32 × 32 ピクセルのアイコン。
  - 簡単な説明。多くても 2 行のテキストを使用して説明を表示できることを確認します。
- ツールヒントを使用して、ショートカットキーを説明します。通常のメニューとは異なり、アプリケーション メニューはラベルを使用してショートカットキーを説明しません。



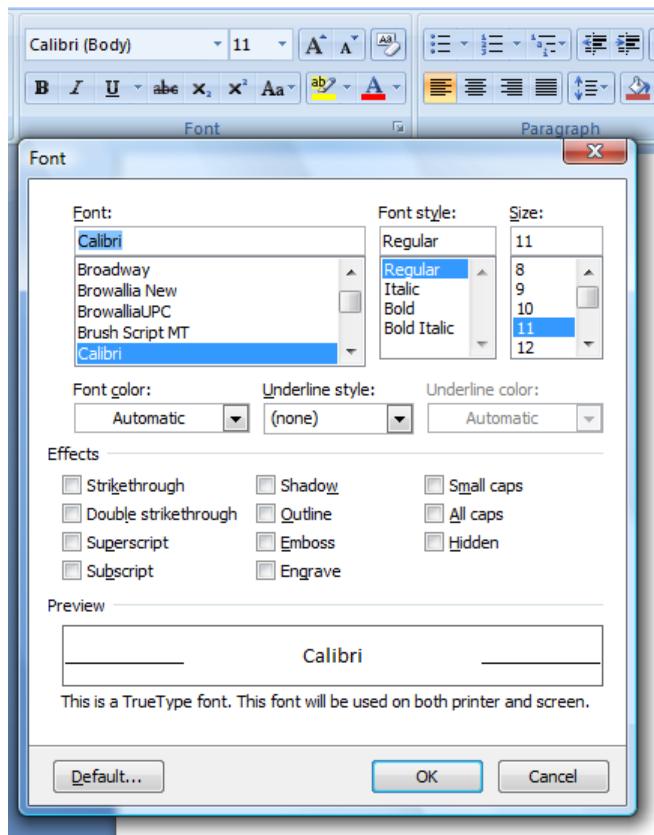
この例では、ツールヒントを使用してショートカットキーを説明しています。

#### クリック アクセス ツールバー

- よく使用するコマンドへのアクセスを提供するには、クイック アクセス ツールバーを使用します。アプリケーション ボタンまたはリボンのコマンドを配置できます。
- リボンを使用する際は常にクイック アクセス ツールバーを配置します。リボンにタブが 1 つの場合でもそうします。これにより、プログラム間の一貫性が保たれます。
- クイック アクセス ツールバーには、アプリケーション メニューのよく使用するコマンドを事前に配置します。[保存] および [元に戻す] がプログラムでサポートされる場合はこれらのコマンドを配置し、[開く] および [印刷] がサポートされ、よく使用される場合はこれらのコマンドを配置します。
- [クイック アクセス ツールバーのカスタマイズ] メニューの場合、最もよく使用される即時型のコマンドを最大 12 まで配置できます。即時型のコマンドは実行する前に追加の入力を必要としないので、クイック アクセス ツールバーに最適です。任意の即時型のコマンドを配置できますが、[ホーム] タブにないコマンドを優先します(ユーザーは [ホーム] タブにあるコマンドを使用することが多いため)。
- [クイック アクセス ツールバーのカスタマイズ] メニューで関連するコマンドの組み合わせがある場合、使用頻度に関係なく両方とも表示します。一般的な組み合わせは、[開く]/[閉じる]、[戻る]/[進む]、[元に戻す]/[やり直し] です。
- [クイック アクセス ツールバーのカスタマイズ] ダイアログには、任意のコマンドを追加できる方法を用意します。最もよく使用されるコマンドを表示する [基本設定] コマンドのフィルターを設定し、既定でこのフィルターを選択します。

#### ダイアログ ボックス起動ツール

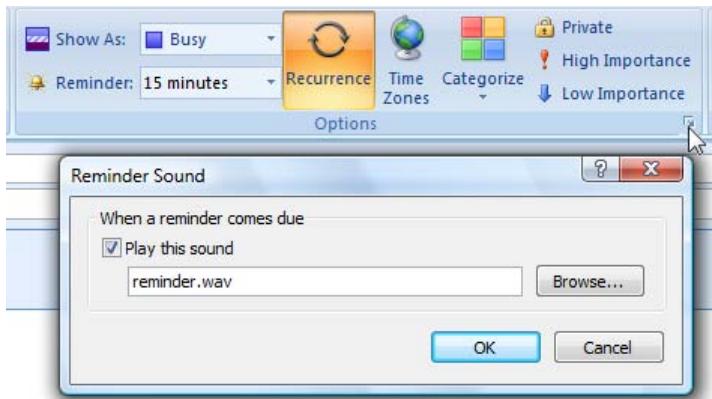
- 頻繁に使用しないコマンドおよび設定が含まれている関連するダイアログ ボックスがある場合は、ダイアログ ボックス起動ツールを使用してグループを表示します。ダイアログ ボックスにはグループのすべてのコマンドとその他のコマンド(グループと完全に異なるコマンドのセットやグループと同じコマンドではないもの)が含まれている必要があります。



この例では、[フォント] ダイアログ ボックスは [フォント] グループ内のコマンドの部分集合を表示しています。

- コマンドを直接実行するためにダイアログ ボックス起動ツールを使用しないようにします。ダイアログ ボックス起動ツールではダイアログ ボックスを表示する必要があります。
- よく使用するコマンドおよび設定にアクセスするためにダイアログ ボックス起動ツールを使用しないようにします。リボンに直接配置されているコマンドに比べて、ダイアログ ボックスのコマンドおよび設定は比較的見つけにくくなっています。
- ダイアログ ボックスの名前をグループの名前と一致させます。完全に一致している必要はありませんが、ユーザーが表示結果に違和感を持たないように、類似した名前にする必要があります。

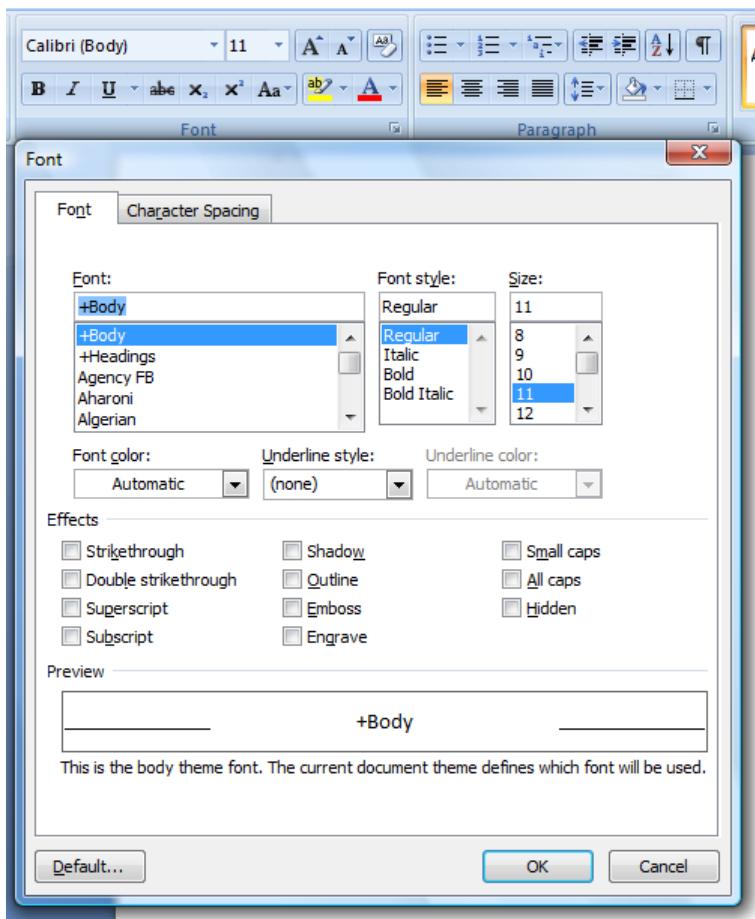
間違った例:



アラーム音はアラームのオプションですが、ダイアログ ボックス起動ツールを使用してアラーム音を設定することは想定されていません。

- グループに関連するコマンドおよび設定のみ表示します。ダイアログ ボックスに他のものを表示すると、ユーザーはこの方法以外に他のコマンドと設定にアクセスできないと結論付ける可能性があります。

間違った例:



この例では、[フォント] ダイアログ ボックスに、関連するタブと関連性のない [文字幅と間隔] 設定が表示されています。

## ラベル

### フォント

- リボンの日本語ラベルには **Meiryo UI** フォントを使用します。

abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ  
あいうえおかきくけこさしすせそアイウエオカキケコサシスセソ  
亞哩娃阿哀愛挨始達葵酉榎惡握渥旭葦芦鰯梓庄斡扱宛姐虹飴綾鮎  
0123456789 ()「」！？、。、。

### タブ

- すべてのタブにラベル付けします。
- 実用的である場合は常に、**標準的なリボン タブ**を使用するようにします。
- できるだけ簡潔な 1 語のラベルを使用します。複数単語のラベルも使用できますが、より多くの領域を必要とし、ローカライズが難しくなります。

コンテンツを明確かつ正確に説明する適切なタブ名を選択します。適度に具体的な名前を選択します。タブ名は、コンテンツが表示されたときにユーザーが違和感を持たないように、予測可能なものにする必要があります。最もよく使用されるコマンドに使用されるので、[ホーム] タブは一般的な名前になっています。

#### 間違った例:

基本設定

詳細設定

上記のタブ名は、適切な方法でコンテンツを説明していません。このようなタブ名では、ユーザーは目的のコマンドが基本設定か、詳細設定であるかを判断する必要があります。

- 目的を反映するタブ名を選択します。タブに関連付けられている目的またはタスクを考慮します。
- その他のすべてのタブ名と明確に区別できるタブ名を選択します。
- タブには名詞または動詞を使用します。タブ名では、同じ文法構造の表現にする必要がないので、名詞または動詞に関係なく最適なラベルを選択します。
- 動名詞は使用しません(英語の場合)。代わりに動名詞の派生元の動詞を使用します。

#### 正しい例:

描画 (Draw)

校閲 (Review)

#### 間違った例:

描画すること (Drawing)

校閲すること (Reviewing)

- 特に隣接するタブでは、先頭の文字が同じタブ名を避けます。リボンを縮小すると、これらのタブ名は切り詰められて同じテキストが表示されます。

#### 間違った例:

Format (書式設定)

Formulas (数式)

- できるだけ単数形の名前を使用します(英語の場合)。ただし、単数形の名前が不自然な場合は複数形の名前を使用できます。
- タイトルスタイルの大文字化を使用します。
- 末尾に句読点は付けません。

### コンテキスト タブおよびタブ セット

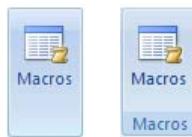
- コンテキスト タブ セットのラベルの末尾に "ツール" を付けます。これにより、コンテキスト タブの目的を識別しやすくなります。
- タイトルスタイルの大文字化を使用します。
- 末尾に句読点は付けません。

### グループ

- すべてのグループにラベルを付けます。

例外: グループ内のコマンドが 1 つの場合、およびグループとコマンドのラベルが同じとなる場合は、グループ ラベルを省略します。

正しい例: 間違った例:



グループ ラベルがグループ内の唯一のコマンドと重複している場合は省略します。

- 実用的である場合は常に、標準的なリボン グループを使用するようにします。
- できるだけ簡潔な 1 語のラベルを使用します。複数単語のラベルも使用できますが、より多くの領域を必要とし、ローカライズが難しくなります。
- コンテンツを明確かつ正確に説明する適切なグループ名を選択します。汎用的ではなく、具体的な名前にする必要があります。

#### 間違った例:

コマンド

タスク

ツール

操作

オブジェクト

基本設定

詳細設定

設定

オプション

個人設定

詳細

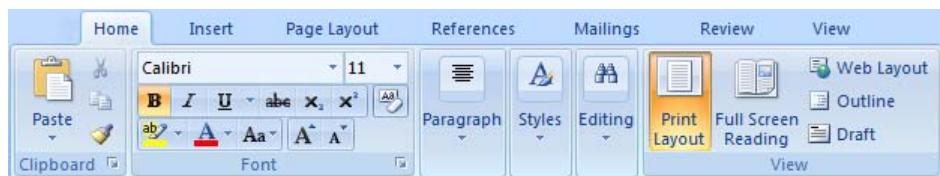
その他

上記のグループ名は、適切にコンテンツを説明していません。すべてのコマンドが上記のグループに当てはまる可能性があります。

- 目的を反映するグループ名を選択します。グループ内のコマンドに関連付けられている目的またはタスクを考慮します。
- 動名詞の使用は避けます(英語の場合)(～ ing で終了する名前)。ただし、動名詞の派生元の動詞を使用すると紛らわしくなる場合は、動名詞を使用できます。たとえば、"Edit" や "Proof" の代わりに "Editing" や "Proofing" を使用します。

- タブ名と同じグループ名は使いません。グループが配置されているタブ名をグループ名として使用しても情報は増えません。別のタブの名前を使用すると紛らわしくなります。

間違った例:



この例では、別のタブの名前をグループに付けたことで紛らわしくなっています。

- できるだけ単数形の名前を使用します(英語の場合)。ただし、単数形の名前が不自然な場合は複数形の名前を使用できます。

正しい例:

フォント

段落

図

切り替え効果

- [センテンススタイルの大文字化](#)を使用します。
- 末尾に句読点は付けません。

コマンド

- すべてのコマンドにラベルを付けます。明示的なテキストラベルを付けると、ユーザーは簡単にコマンドを見つけて理解することができます。
  - 例外: アイコンがよく知られている場合や、領域を優先する場合は、コマンドのラベルを省略できます。ほとんどの場合、ラベルのないコマンドは[ホーム]タブに配置します。この場合は、名前プロパティを適切なテキストラベルに割り当てます。この方法に従うと、スクリーンリーダーなどの支援テクノロジー製品で、画像に関する代替情報を提供できるようになります。

正しい例:



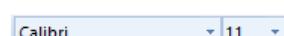
これらのコマンドはよく知られているので、ラベルは必要ありません。

間違った例:



これらのコマンドには、リッチコマンド用のラベルが必要です。

- コマンドボタンには、処理内容がわかる簡潔なラベルを使用します。可能であれば1ワードに、最大でも4語(12文字程度)に納めます。
- ドロップダウンリストの場合、リストに常に値があるのであれば、現在の値をラベルに使用します。



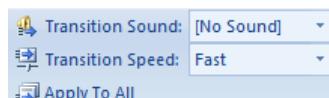
この例では、現在選択されているフォント名がラベルの役割を果たしています。

[編集可能なドロップダウンリスト](#)に値がない場合は、[プロンプト](#)を使用します。



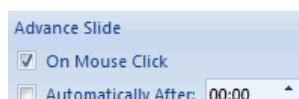
この例では、編集可能なドロップダウンリストのラベルにプロンプトが使用されています。

- 内容がわかりにくかったり、頻繁に使用されないドロップダウンリストには、明示的なラベルが必要です。ラベルの末尾にコロンを付けます。



この例では、頻繁に使用されないドロップダウンリストに明示的なラベルが使用され、わかりやすくなっています。

- テキストボックスには、明示的なラベルを使用します。ラベルの末尾にコロンを付けます。



この例では、テキストボックスコントロールに明示的なラベルが付けられています。

- センテンススタイルの大文字化を使用します。これはWindowsの[トーン](#)に従っています。
- ラベルは動詞の命令形で始めます(英語の場合)。次の場合は例外です。
  - 動詞がタブ名またはグループ名と同じ場合は省略します。
  - 他のラベルから動詞を簡単に推論できる場合は、"表示"、"作成"、"挿入"、"書式設定"といった共通する動詞を省略します。

- ・末尾に句読点は付けません。
- ・スペースを節約するために、リボン コマンドのラベルに省略記号は付けません。ただし、アプリケーション ボタンおよびドロップダウン メニューのコマンドでは省略記号が使用されます。

#### 拡張ツールヒント

- ・適用できる場合は、タイトルを使用して、コマンド名とショートカットキーを表示します。
- ・タイトルの末尾に句読点は付けません。
- ・説明は動詞から始めます(英語の場合)。説明により、特定の機能がユーザーが探しているものかどうかを判断できるようにします。説明は、"～する場合はこの機能を使用します。" というように、完結した文にします。

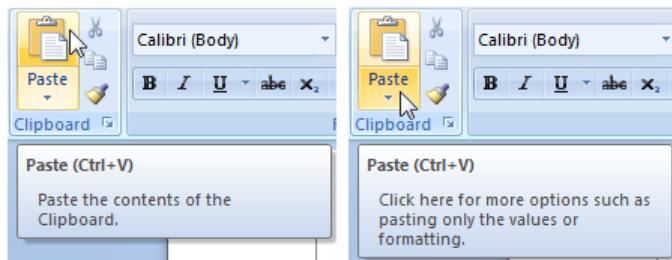
正しい例:

Insert or draw a table into the document. (表を文書に挿入または描画します。)

間違った例:

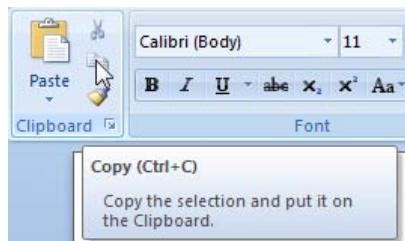
Inserts or draws a table into the document. (表を文書に挿入または描画。)

- ・説明は短くします。要点だけを伝えます。説明が長いと、読む意欲が失われます。
- ・分割ボタンには、異なるツールヒントを使用して分割ボタンメニューを説明します。



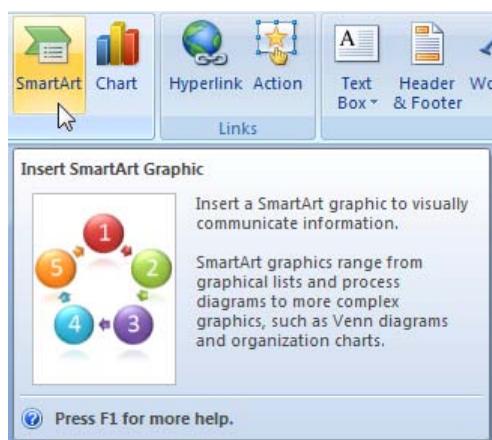
この例では、分割ボタンメニューにはメイン ボタンとは異なるツールヒントが表示されています。

- ・オプションの補足説明を使用して、コントロールの使用方法を説明します。このテキストには、コントロール自体が状態を示していない場合に、コントロールの状態に関する情報(無効になっている理由など)を含めることができます。このテキストは短くし、詳細な説明にはヘルプ トピックを使用します。



この例では、コマンドが無効になっている理由をツールヒントで説明しています。

- ・説明および補足説明には、文を使用し、末尾に句点を付けます。

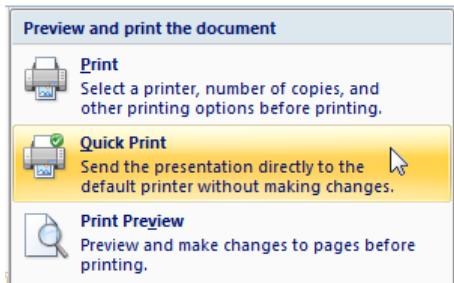


補足説明が含まれている拡張ツールヒントです。

- ・センテンス スタイルの大文字化を使用します。

#### アプリケーション ボタン

- ・即時型のコマンドであることを示す場合は、"クイック" を使用します。



この例では、"クリック"でコマンドが即時型であることを示しています。

- コマンドに追加情報が必要であることを示すには、[省略記号](#)を使用します。
- センテンススタイルの大文字化を使用します。

## ドキュメント

リボンに言及するときは、以下のことに留意します。

- リボンやそのコンポーネントは、"リボン"、"タブ"、"グループ"、"コントロール"と表現します。これらの用語は大文字化しません(英語の場合)。
- 円形のボタンは"アプリケーションボタン"、それに含まれるメニューは"アプリケーションメニュー"と表現します。
- ツールバーは"クリックアクセスツールバー"と表現します。
- タブは、そのラベルと"タブ"という語で示します。大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。
- コマンドはラベルで示します。ラベルのないコマンドはツールヒント名で示します。大文字と小文字の区別を含め、ラベルのテキストを正確に引用します。ただし、省略記号は含めません。"ボタン"や"コマンド"という語は含めません。
- ユーザー操作を説明する場合は、タブおよびコントロールに対して、"クリック"を使用します。編集可能なドロップダウンリストには、"入力"を使用します。"選択(choose、select、pick)"は使用しません。
- 使用できないアイテムは"利用不可の"と表現し、"薄く表示された"、"無効になった"、"灰色表示の"とはしません。プログラミング文書では、"無効"を使用します。
- ラベルは半角の角かっこ([ ])で囲み、可能な場合は太字にします。

例:

- [ホーム]タブで、[形式を選択して貼り付け]をクリックします。
- [ホーム]タブの[フォント]ボックスで、「Segoe UI」と入力します。
- [校閲]タブで、[変更履歴とコメントの表示]をクリックし、[校閲者]をクリックします。
- [書式設定]タブで、[図ツール]の[図の圧縮]をクリックします。

## リボンのデザイン プロセス

機能をリボンに移動すると、ボタンをタブに整理する以上のことことが期待できます。

### リボン

プログラムにとってリボンのコマンド ユーザー インターフェイスが適切であると判断したら、次のステップは、変更のための論理的で順序立てられたプロセスを採用することです。メニュー バーとツールバーからリボンへの移行は、プログラム インターフェイスを大幅に変更することになります。各機能を吟味し、機能の意義と価値を伝えることができるリボンの最適な使用方法を決定する必要があります。

タブとグループの観点から検討を開始すると、機能の差が見えてきます。また、以前のバージョンではダイアログ ボックスにのみ存在していたコマンドを、リボンに追加する必要が出てきます。使用頻度が高いダイアログ ボックスでユーザーが何を実行しているかを確認し、それらのコマンドをリボンに移動します。可能な限り、ユーザーがダイアログ ボックスを開かなくてもタスクを完了できるようにします。

新しいコマンド ラベルだけでなく、すべてのコマンド ラベルをテストし、変更します。これには長時間かかることを見込んでおきます。通常の製品サイクルでは、元のコマンド名の変更を正当化することは簡単なことではありません。コマンドをリボンに移動する場合は、このような変更を行うことができます。

従来の機能を微調整し、リボン モデルへの適合性を高める作業も必要です。たとえば、メニューまたはツールバーに動的に表示されるコマンドがあるとします。リボンを使用する場合、この機能を変更する必要があります。そうしないと、コマンドの表示によってタブとグループのレイアウトが大きく変化することになります。リボンのガイドラインを適用し、コマンドを非表示にするのではなく無効にする必要があります。

---

### ダイアログ ボックスのコマンドを表示する

プログラムのタブとグループに配置するものが大まかに把握できたら、その構成に注目し、不足しているものがないかどうかを確かめます。

たとえば Microsoft® Word では、利用状況データにより、[フォント] ダイアログ ボックスがプログラムで最も使用頻度の高いダイアログ ボックスの 1 つであることがわかっています。また別のデータでは、上付き、下付き、取り消し線の各機能が手動で書式設定ツールバーに追加されることが多いことが明らかになっています。Microsoft PowerPoint® では [フォントの拡大] および [フォントの縮小] ボタンがよく使用されるため、これらのコマンドは既定のツールバーに表示されています。

その結果、Word リボンの [フォント] グループには、Word の既定の書式設定ツールバーにある元のコマンドに加えて、新たに 8 つのコマンドがインポートされています。こうすると、[フォント] ダイアログ ボックスを開く必要性が低くなります。これは、以前のバージョンと比較してユーザーがカスタマイズする必要がある部分が大幅に減ったという意味でもあります。この場合の作業は、開発者にとって容易であることがわかっています。すべての機能が、ツールバーに追加可能なボタンとして既にデザインされているためです。

多くの場合、ツールバーに適したバージョンの機能がまだ存在していません。たとえば、Microsoft Excel 2003 では、条件付き書式の影響を受けるセルをすべて選択する場合、[ジャンプ] ダイアログ ボックスを開き、[セル選択] ボタンをクリックし、[条件付き書式] を選択して [OK] をクリックすることによって、結果を確認する必要があります。このダイアログ ボックスが、この機能の存在する唯一の場所であり、ユーザーがメニューまたはツールバーに追加できる同等のコマンドはありません。そのようなコマンドを Office 2007 リボンに追加するとなると、ツールバー ボタンに似たバージョンを作成する必要があります。

ダイアログ ボックスの機能をリボンに移動する方法がすぐにはわからないことがあります。典型的なダイアログ ボックスでは、ユーザーは [OK] をクリックするまでダイアログ ボックス内で作業する必要があります。[OK] をクリックした時点で選択項目がすべて反映され、ユーザーはプログラム ウィンドウに戻ります。リボン上のコントロールを操作する場合、変更内容はすぐに反映されます。

つまり、コントロールはすべて独立して動作する必要があり、他のコントロールに依存できません。たとえば、フォントの種類を選択してからフォントサイズを選択し、最後に[OK]をクリックすることによって、両方を同時に適用することはできません。リボンには[OK]や[キャンセル]ボタンはありません。これは、コントロールが互いに影響を与えないという意味ではありません。たとえば、[段落]グループでは、[右揃え]を選択すると、[左揃え]の選択が解除されます。

## コマンドにラベルを付ける

適切なラベルがあるとコマンドの理解に大きく役立ちますが、ほとんどのツールバー コマンドにはラベルが付いていません。幅が 640 (または 800) ピクセルの画面にはスペースに余裕がないため、ツールバーにできるだけ多くのコマンドを配置しようとすれば、通常、ラベルを省略することになります。また、ラベルが翻訳され文字数が極端に多くなると、他のコマンドを画面外に押し出す可能性があります。

リボンの主なメリットの 1 つが、すべての項目にラベルを付けることができる十分なスペースを提供できることです。タブとグループのレイアウトを作成するとき、最も一般的な画面解像度では、リボン内のほとんどのコマンドをラベルではっきりと示す必要があるという目標を忘れないようにします。コマンドをもう 2、3 個、タブに追加するために、ラベルを省略したくなることがあります。多くのユーザーはアイコンをすべて記憶しているわけではないことや、ツールバーとは異なり、ラベルを確認できるメニュー バーが別に存在していないことを認識する必要があります。

よく知られているいくつかのコマンドに限って、既定でラベルを付けないでおくことができます。たとえば [太字] や [切り取り] のようなコマンドが挙げられます。これらは、ほとんど間違いなく [ホーム] タブに配置されています。

コマンドに追加の入力が必要であることを示すのに、省略記号 (...) は使用しません。リボン上では、テキストの切り詰めが行われていることを示す場合にのみ省略記号を使用します。ただし、ドロップダウントラックメニュー内の省略記号には依然として従来の意味があります。リボンでは、自動的にラベルの末尾から省略記号が取り除かれます。タブにもう 2、3 個のボタンを追加しようとするとき、そのスペースがすべて確保されていれば便利でしょう。

その他のガイドラインについては、「[ラベル](#)」を参照してください。

## ユーザー インターフェイスに一貫性を持たせる

リボンの主なメリットの 1 つが、プログラムのすべてのコマンドに单一の固定された場所を提供できることです。この一貫性によってユーザーはプログラムを把握しやすくなります。一方、UI が動的に変化するとさまざまな混乱を引き起こします。計画の早い段階に、こうした問題を確認しておきます。

一貫性を維持するには、適用されないコマンドを非表示にするのではなく、無効にします。コマンドを非表示にすると、グループのレイアウトが変化してタブ全体に影響を及ぼし、場合によってはまったく違うものになることがあります。

ラベルは動的に変化させないようにします。繰り返しになりますが、実行中にコマンドのラベルを変えると、タブ全体のレイアウトに大きく影響します。既存の機能とラベルを微調整し、コマンドをリボンに適したものにする必要があります。

既存のコマンドのデザインを変更する必要が生じることもあります。たとえば、記録されていないテキストが選択された場合にのみ表示される [メモの挿入] コマンドと、記録済みのテキストが選択された場合にのみ表示される [メモの削除] コマンドがあるとします。リボンでは、両コマンドが常に表示され、1 度にどちらか一方だけが有効になるようにする必要があります。

## 使いやすいリボンにする

多くのユーザーは変化を好みません。その変化のために手間がかかったり、学習し直す必要がある場合はなおさらです。以前のプログラムでは簡単であったタスクが新たなプログラムでは難しくなったり発見しにくくなると、ユーザーを苛立たせることになります。

使いやすいリボンにするには、次のようにします。

- 以前と変わることなく、コマンドを発見しやすいようにします。リボンの統合性と明示的なラベル付けを利用すると、ほとんどのコマンドが以前よりも見つけやすくなると考えられます。

- 以前と変わることなく、タスクを実行しやすいようにします。リボンで結果指向のコマンドやその他の種類のプレビューを使用することによって、多くのタスクが以前よりも使いやすくなると考えられます。
- 元のプログラムで最も頻度に使用されたキーボードショートカットキーとアクセスキーが引き続き同じように使用できるように、特に力を注ぎます。従来のメニュー カテゴリのアクセスキーに新たな意味を割り当てるることは避けます。たとえば、従来のメニュー バーのバージョンのプログラムに [編集] メニューがあった場合、それに相当するタブにはアクセスキーとして "E" を使用するようにします。上級ユーザーの場合はキーボードを効率よく使用するという要望が強いので、こうして一貫性を保つと高い評価が得られます。逆に、一貫性が欠けていると、大きな欠点として捉えられます。

これで完了ではありません。デザイン プロセス全体にわたるユーザー テストを実施し、タスクが実行できるかどうかだけでなく、以前よりも明らかに操作性が向上していることを確かめます。

## まとめ

以下に、機能コマンドをリボンに移動する際の手順を簡単に示します。実際には、製品サイクルの過程で実施される何回もの繰り返しのなかで、これらの手順を何度も実行することになります。

1. プログラム内のすべてのコマンドを集めて一覧表を作成します。この一覧に基づいて一定期間作業するため、最初に一覧表作成の時間を取ってください。
  - この一覧表には、メニュー バー、ツール バー、コンテキスト メニュー、およびカスタム UI のコマンドをすべて収めておきます。
  - コマンドごとに、以下のものが必要です。
    - ツール バー コントロール ID (TCID) 情報
    - ラベル
    - 利用状況データ
2. 以下の標準プログラム タブに属するコマンドを除外します。
  - ホーム
  - 挿入
  - ページ レイアウト
  - 校閲
  - 表示
  - 開発
  - 標準コンテキスト タブ
3. コンテキスト タブに属するコマンドを除外します。
4. 以下の標準グループに属するコマンドを除外します。
  - クリップボード
  - フォント
  - 段落
  - 編集
  - 表
  - 図
  - テーマ
  - ページ設定
  - 配置
  - 文章校正
  - コメント
  - 文書の表示
  - 表示/非表示
  - ズーム
  - ウィンドウ
  - マクロ

5. 残りのコマンドを、いくつかの関連する機能から成るグループに整理します。

- プログラム固有のコマンド群をグループにします。
- ユーザーを対象とする "分類" テストの実施について、ユーザビリティ分析担当者と検討します。

6. グループを、5～10個のタスクベースのタブに整理します。

- タブのひな形作りを開始し、試作した UI を使用してみます。
- ユーザー分析担当者と協力し、ユーザーを対象にコマンドのカードソート テストを実施します。
- 不足がないかどうか調べます。ダイアログ ボックスから移動する必要があった機能が存在していることを確かめます。
- ある機能または機能セットの意義と目的を的確に表すための、ギャラリーを使用する場所を決めます。
- ユーザーがタブを見た際に、その用途がわかるようにします。それぞれのタブに明確な目的が存在するようにします。
- タブ セット全体について考えます。ユーザーがタブの並びを見たときに、プログラムの目的を把握できるようにします。
- プログラムの今後のバージョンについて考えます。選択したタブが、数世代先のバージョンにまで使用できるかどうかを考え、プログラムの全体的な方向性に適合していることを確認します。

7. 新しい機能を作成し、タブに追加します。

- 新しい機能を作成するまでは、タブ デザインに大きな空間が空くことになります。
- さらに悪いことに、何かを除外すると、その場所が空いたままになります。最初にこのことを考えておきます。機能の除外は常に起こるため、まずこの問題に取り組む必要があります。

8. 機能の構成をテストします。

- このテストは、ベータ版のリリースの際だけでなく、開発サイクル全体を通じて実行します。
- 試作品を自分で使ってみます。
- 他のユーザーにも使用してもらい、フィードバックを受けます。
- できる限り、キーボードのアクセスキーとショートカットキーに、以前のバージョンのプログラムと互換性があるようにデザインします。
- 最初の反応だけでなく、十分に使用された後にもフィードバックを受けるようにします。

9. 上記の手順を何度も繰り返します。

- 開発者、テスト担当者、担当マネージャーなど、すべての担当者が、コマンドの構成を定期的に繰り返し実施するつもりでいるようにします。
- レイアウトはすべて単純な XML で記述されているため、簡単に変更することができます。
- ただし、現在の計画を検証することは簡単ではありません。開発者が先走り、テスト担当者を置き去りにすることのないようにします。

## プログラム コマンドのパターン

### リボン

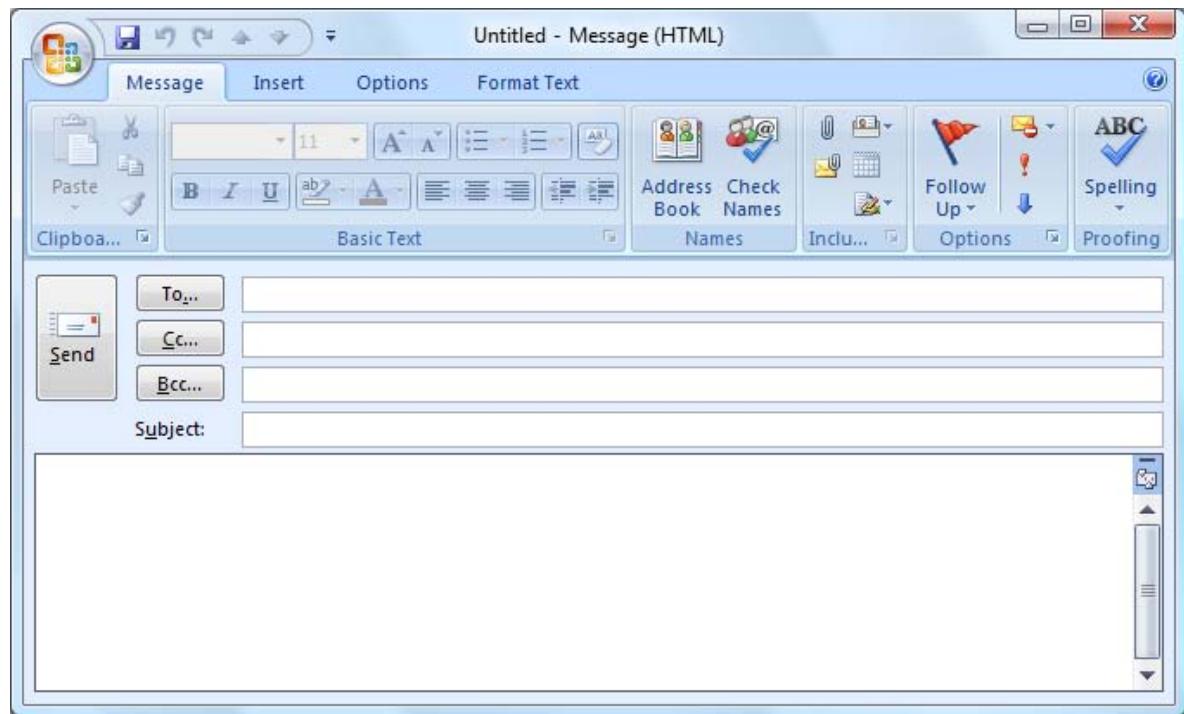
次のように、プログラムの種類によって、適切なコマンドの提示方法を判断することができます。

**簡単なドキュメント作成** ユーザーの目的: 基本的なドキュメント作成タスクに重点を置くこと。わかりやすさと単純さが重要です。

例: ワードパッド、ペイント、Windows Journal など。

**簡単なドキュメントの作成および表示に使用される。** 推奨されるコマンドの提示方法: これらのプログラムはシンプルですが、通常、コマンドが多すぎて1つのツールバーに収まりません。一般的にリボンが適切ですが、メニューバーとシンプルなツールバーの組み合わせも効果的です。多くの場合、結果指向のコマンドを表示できるかどうかが判断基準になります。

すべてのユーザーを対象としています。



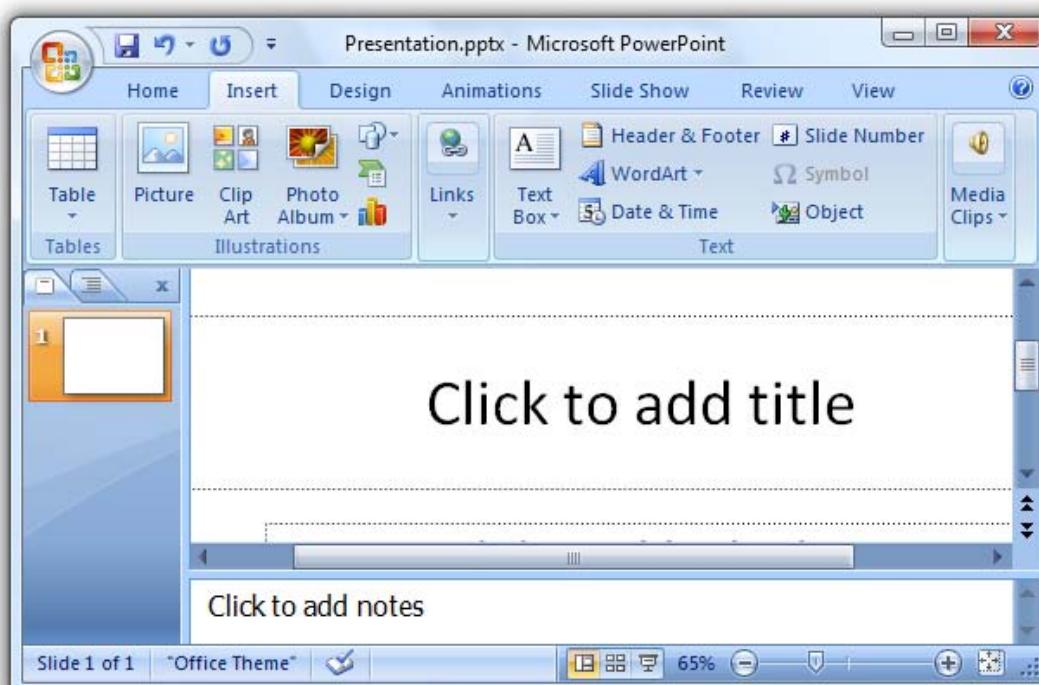
シンプルなドキュメント作成プログラムには、一般的にリボンが最適です。

**中級のドキュメント作成およびオーサリング** ユーザーの目的: 無理なく、広範囲のタスクを実行できること。ただし、わかりやすさと単純さが重要なことに変わりはありません。

例: Microsoft Office、Windows メーラーなど。

推奨されるコマンドの提示方法: 特に、結果指向のコマンドを提示する場合、リボンが理想的です。

やや複雑なドキュメントの作成および表示に使用されます。中級ユーザーを対象としています。

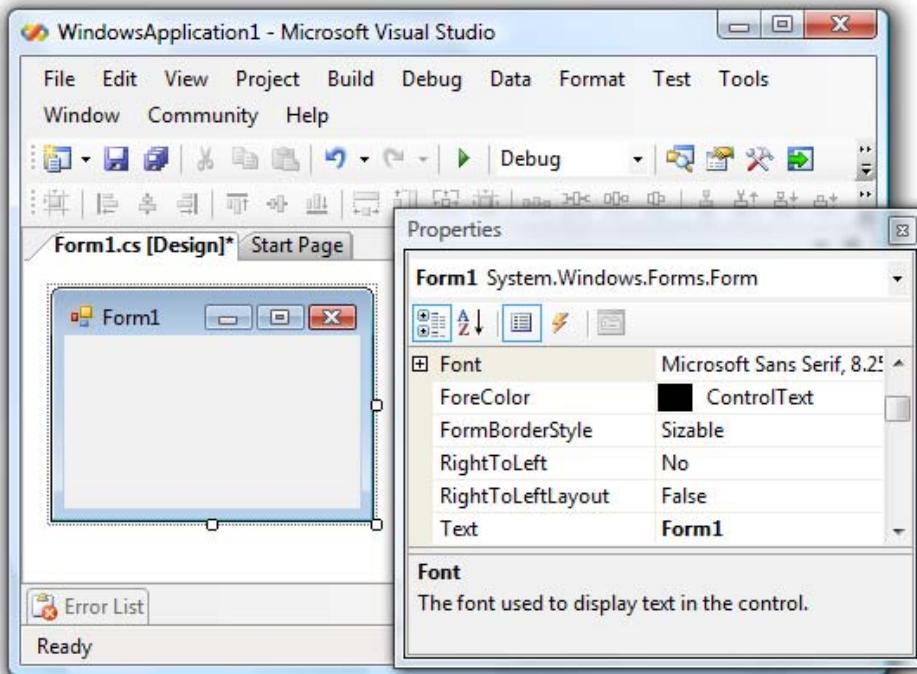


中級のドキュメント作成プログラムには、リボンが理想的です。

高度なドキュメント作成およびオーサリング  
高度なドキュメントの作成および表示に使用されます。熟練した上級ユーザーを対象とします。

ユーザーの目的: 効率性。大規模で複雑なプロジェクトを迅速に遂行すること。認識のしやすさと習得のしやすさが備わっていれば理想的ですが、必須ではありません。  
例: Microsoft Visual Studio® など。

推奨されるコマンドの提示方法: 効率的、構成可能で、スケーラブルな UI。ドッキングを解除して、パレット ウィンドウとして使用できる複数のツールバーを備えたメニュー バー。縦型の作業ウィンドウが適切な場合もあります。



高度なドキュメント作成プログラムには、ドッキングを解除できるメニュー バーが一般的に最適です。

ドキュメント ビューアーまたは ブラウザーで作成したコンテンツの検索、読み込み、表示、または再生に使用されます。すべてのユーザーを対象とします。

ユーザーの目的: コンテンツに集中すること。少数のシンプルなコマンドを検索、参照、移動、実行します。  
例: Windows® Internet Explorer®、Windows Media® Player、Windows フォト ギャラリーなど。

推奨されるコマンドの提示方法: ユーザーはコンテンツに集中する必要があることに加えて、コマンドは一般的にシンプルで少数であることを考慮すると、インライン コマンド、メニュー バー(既定で非表示にしてもよい)、シンプルなツールバーの組み合わせが最適です。ただし、結果指向のコマンドを使用するとプログラムにメリットがある場合や、ドキュメントに対話型オブジェクトが含まれている場合は、リボンの方が適していることがあります。



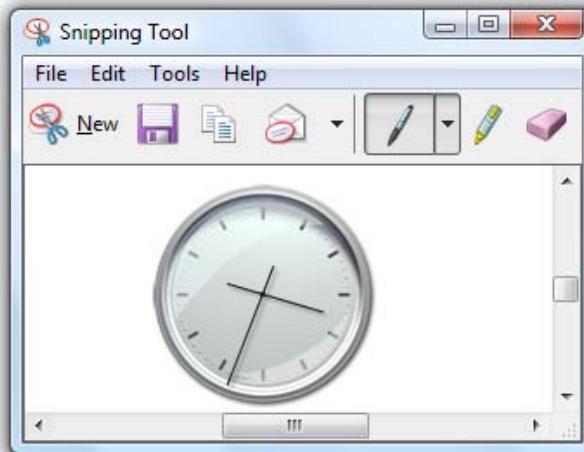
ドキュメントビューアーには、直接的なコマンド、メニュー バー、およびシンプルなツール バーの組み合わせが一般的に最適です。

**ユーティリティ** ユーザーの目的: すばやく簡単にタスクを実行できること。一部のタスクには馴染みのないものもあります。

タスクの実行に 例: 電卓、メモ帳、ガジェット、Windows Live™ Messenger、Windows FAX とスキャン、Windows Snipping Tool など。

すべてのユーザーを対象としています。 推奨されるコマンドの提示方法: シンプルなユーティリティには、直接的なコマンドと設定(コマンドボタン、ラジオ ボタン、チェック ボックス、ドロップダウンリスト、スライダーなど)が最適です。より高度なユーティリティには、メニュー バーとシンプルなツール バーの組み合わせを使用します。

コマンドはシンプルで少數なので、リボンを使用する必要はありません。



メニュー バーとシンプルなツール バーを備えているユーティリティもあります。

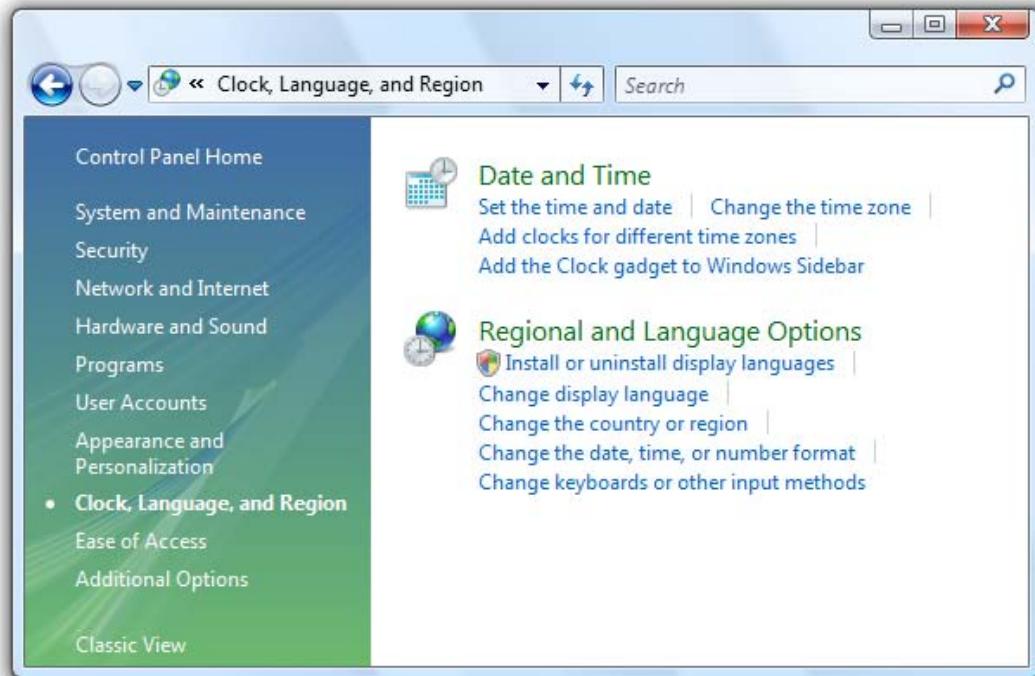
**構成プログラム** ユーザーの目的: すばやく簡単に馴染みのないタスクを実行できること。

ハードウェアおよびソフトウェアの構成に使用 例: コントロールパネルのページやプロパティ シートなど。

推奨されるコマンドの提示方法: 直接的なコマンドが最適です。

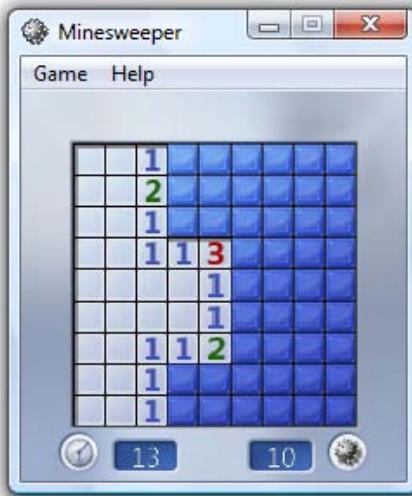
アの構成に使用

されます。すべてのユーザーを対象としています。



コントロールパネルのページには、シンプルで直接的なコマンドを提示する必要があります。

ゲーム ユーザーの目的: すぐにゲームを開始し、ゲームに集中すること。  
ゲームの実行に 例: Halo®、ソリティア、フリーセル、インクボールなど。  
使用されます。 推奨されるコマンドの提示方法: コマンドがシンプルで、頻繁に使用される場合は、直接的なコマンド  
すべてのユーザー を使用します。 頻繁に使用されないコマンドが複数ある場合は、シンプルなメニュー バー内に配置し  
対象とし ます。  
ています。



ほとんどのゲームには、シンプルで直接的なコマンドの提示方法を使用する必要があります。

## テキスト

ここでは、Windows® ベースのアプリケーションでテキストを使用するためのガイドラインについて説明します。

- [ユーザーインターフェイスのテキスト](#)。テキストは UI 内のどこにでも出現します。ここでは、無秩序な状態に秩序をもたらす方法について説明します。
- [スタイルとトーン](#)。ユーザーが UI にどう反応するかは、表現の方法とは関係があっても、表現の内容とはあまり関係のないことがよくあります。

特定のテキストのガイドラインについては、[コントロールとウィンドウ](#)のテキストまたはラベルに関するセクションにも示しています。

# ユーザー インターフェイスのテキスト

使用パターン  
デザインコンセプト  
ガイドライン  
全般  
テキストのフォント、サイズ、色  
その他のテキストの特性  
句読点  
大文字化  
グローバリゼーションとローカライズ  
タイトルバーのテキスト  
メイン指示テキスト  
補足指示テキスト  
コントロール ラベル  
補足説明  
コミット ボタンのラベル

"ユーザー インターフェイスのテキスト" は UI 画面上に表示されます。このテキストには、コントロール ラベルと静的テキストが含まれます。

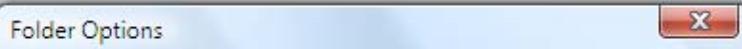
- ・ コントロール ラベルはコントロールを識別するもので、コントロール上またはコントロールの横に配置されます。
- ・ 静的テキスト (対話型コントロールの一部でないのでそう呼びます) は、詳細な指示または説明を提供し、ユーザーが情報を得たうえで判断を下せるようにするものです。

注: [スタイルとトーン](#)、[フォント](#)、[コモン コントロール](#)に関するガイドラインは、それぞれ別の項目として記載しています。

## 使用パターン

UI テキストにはいくつかの使用パターンがあります。

タイトルバーの  
テキスト  
ウィンドウまたは  
ダイアログ  
ボックスの呼び  
出し元を識別す  
るために、タイ  
トルバーのテキ  
ストを使用しま  
す。



この例では、タイトルバーのテキストによってウィンドウが識別されています。

メイン指示テキ  
スト  
ウィンドウまたは  
ページです  
べきことを簡潔  
に説明するため  
に、目につきや  
すいメイン指示  
テキストを使用  
します。

Do you want to get the latest online content when you search Help?

この例では、メイン指示テキストは、ユーザー自身のメリットや興味という観点から、ユーザーの注意を直接質問に引き付けています。

補足指示テキ  
スト  
ウィンドウまたは  
ページの理解  
や使用に役立つ  
追加の情報を提  
示するために、  
必要に応じて、

詳細情報、コンテキスト、および用語の定義を提示できます。補足指示テキストは、メイン指示テキストを繰り返すことなくさらに詳しく述べるものです。

補足指示テキストを使用します。



この例では、補足指示テキストは、メイン指示テキストに提示された情報に応えて、実行可能な操作が2つあることを示しています。

コントロール ラベル  
コントロール上に直接配置、またはコントロールの横に配置されるラベルです。



この例では、コントロール ラベルにより、ユーザーが選択または変更可能なデスクトップ時計の設定が特定されています。

#### 補足説明

コントロール ラベルの補足説明です(一般に、コマンドリンク、ラジオ ボタン、チェック ボックスに使用されます)。

##### On (recommended)

This setting blocks all outside sources from connecting to this computer, except for those unblocked on the Exceptions tab.

##### Block all incoming connections

Select this option when you connect to less secure networks. All exceptions will be ignored and you will not be notified when Windows Firewall blocks programs.

##### Off

Avoid using this setting. Turning off Windows Firewall will make this computer more vulnerable to hackers or malicious software.

この例では、補足説明によって選択肢の内容が明らかにされています。

## デザイン コンセプト

ソフトウェア開発者は多くの場合、テキストは製品ドキュメントと技術サポートに任せるものであると考えています。"まず開発者がコードを記述してから、誰かを雇って開発品の説明をしてもらう"という考え方です。しかし実際は、重要なテキストはUIが構想されコード化されるときなど、プロセスの早い段階で記述されます。このテキストが最終的に、他のどんな種類の技術に関する記述よりも頻繁に現れ、多くのユーザーの目に留まることになります。

効果的なUIには、わかりやすいテキストが不可欠です。デザインプロセスに不可欠の部分であるUIテキストに関しては、プロのライターや編集者がソフトウェア開発者と協力する必要があります。早期からテキストに取り組むのは、テキストの問題からデザインの問題が明らかになることが多いのです。チームにデザインの説明に関する問題が生じた場合、多くの場合、変更を必要とするのはデザインであり、説明ではありません。

### UI テキストを考慮した設計モデル

UI テキストや、UI 画面上の UI テキストの配置について考える際には、以下の事実を考慮に入れます。

- 人が何かに注目し、読み取りに集中するときは、左から右、上から下に向って読みます(西欧文化圏の場合)。
- ソフトウェアを使用するときは、ユーザーはUI自体ではなく作業に没頭します。したがって、ユーザーは、UI テ

キストを読まずに流し読みします。

- ユーザーがウィンドウに目を通すとき、テキストを読んでいるように見えて、実際はテキストをフィルタリングしていることがあります。必要性を認めない限り、多くの場合、ユーザーは UI テキストを正確に理解しません。
- ウィンドウ内では、さまざまの UI 要素がそれぞれ異なる注目度を得ます。コントロールのラベル、特に作業中のタスクの完了に関連して表示されるコントロール ラベルは最初に読まれる傾向にありますが、静的テキストはユーザーが必要と判断したときだけ読まれます。

一般的な設計モデルの場合、ユーザーは左から右、上から下へとテキストを注意深く読むとは考えないようにします。むしろ、ユーザーは初めにウィンドウ全体をざっと見渡し、主に次のような順序で UI テキストを読み取ると想定します。

1. 中央の対話型コントロール
2. コミット ボタン
3. 中央以外の対話型コントロール
4. メイン指示テキスト
5. 補足説明
6. ウィンドウ タイトル
7. UI 内のその他の静的テキスト
8. 脚注

また、ユーザーは何をすべきかを了解できたら即座に読むことをやめ、操作に移ると想定する必要があります。

#### 重複をなくす

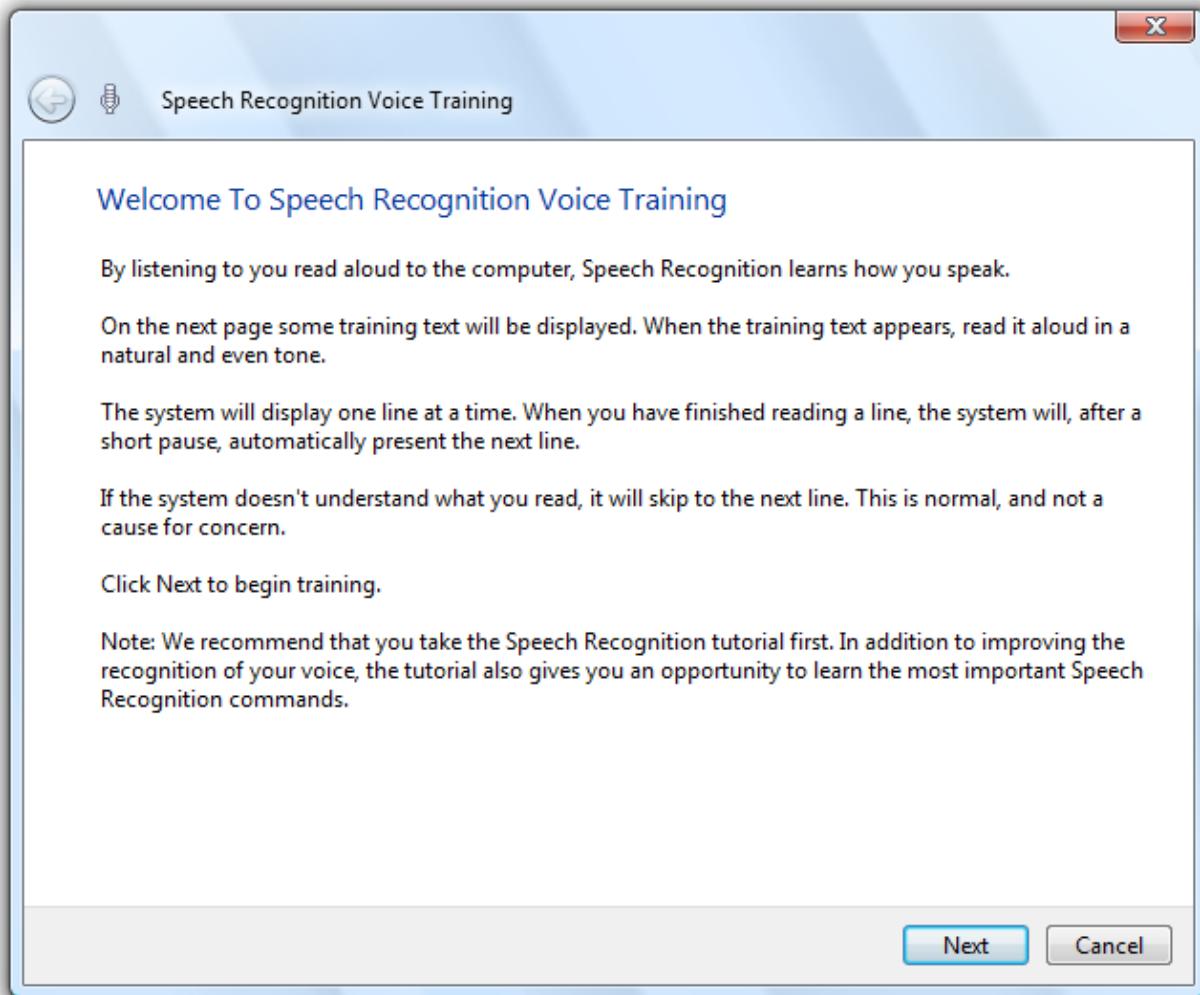
重複したテキストがあると貴重な画面スペースが浪費されるだけでなく、伝えようとしている重要な考え方や操作の効果が弱まります。また、読み手の時間を無駄にします。流し読みが当然とされるコンテキストでは、とりわけ無駄になります。Windows® では、ユーザーに必要な操作を適切かつ簡潔に、一度に説明することに重点を置いています。

各ウィンドウを見直し、コントロールの内外で重複した語や文を省きます。重要なテキストは省かず、必要な場合はいつでも明示します。ただし冗長にならないようにし、自明のことは説明しません。

#### 語数が多くならないようにする

テキストに重複がない場合でも、細部まで説明しようとすると冗長になります。テキストが多すぎると読む意欲が失われて読み飛ばしが多くなり、むしろ伝える内容が減るという皮肉な結果になります。UI テキストでは、重要な情報を簡潔に伝えます。一部のユーザーや一部のシナリオに対して追加の情報が必要な場合は、詳細な[ヘルプコンテンツ](#)へのリンク、または用語を明らかにするための用語集エントリへのリンクを提供します。

#### 間違った例:



この例では、多くのテキストが含まれ、簡単に目を通すことができません。デザイナーの意図とは反対に、多くのテキストが含まれているためにユーザーが何も読まずに「次へ」をクリックする可能性が高くなります。

読む気がなくなるテキストにしないようにするには、1語1語に気をつけてテキストを練ります。意味が加わらないものを取り去り、平易で簡潔なテキストを使用します。

#### 逆ピラミッド型にする

学術的な文書では一般に"ピラミッド"型が採用されます。基盤としての事実を築き、その事実を使用して結論を導いて、ピラミッド型の構造が形成されます。一方、マスコミ業界では"逆ピラミッド"型が採用されます。結論から開始して、読み手が知っているはずの基礎になる部分は取り除かれます。

その後、読み手が関心を持ちそうな部分が徐々に詳しく提示され、読み手はおそらくそれを流し読みします。この方法のメリットは、すぐに要点がわかるのであり、重要な情報は把握できているので、読み手は好きなところで読むのをやめることができます。

UIテキストでは、逆ピラミッド型の構造を採用する必要があります。最初に重要な情報を述べ、ユーザーが任意の場所で読むのをやめることができるようにして、さらに、ヘル普リンクを使用してピラミッドの残りの部分を提示できます。



この例では、重要な情報はメイン指示テキストのクエリ内に配置され、追加の有用な情報は補足指示テキストに配置されます。詳細情報はヘルプリンクをクリックすると提供されます。

#### 5つの重要な点

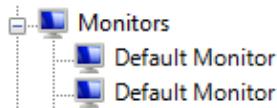
1. 早期からテキストに取り組みます。テキストの問題からデザインの問題が明らかになることが多いのです。
2. 流し読みに適したテキストをデザインします。
3. 重複するテキストを削除します。
4. わかりやすいテキストを使用し、語数が多くならないようにします。
5. 必要であれば、詳細な情報を提供するためにヘルプコンテンツへのリンクを用意します。

## ガイドライン

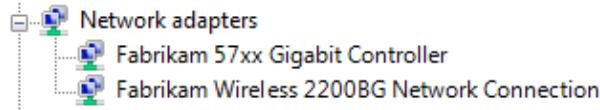
### 全般

- 冗長なテキストは削除します。ウィンドウタイトル、メイン指示テキスト、補足指示テキスト、コンテンツエリア、コマンドリンク、コミットボタンに冗長なテキストがないことを確認します。通常、メイン指示テキストと対話型コントロールのテキストはそのまま残し、他の部分にある冗長なテキストを削除します。
- UI テキストが、大きなブロックにならないようにします。それには、以下のようにします。
  - テキストをより短い文や段落に分けます。
  - 必要に応じて、有用ではあるが必須ではない情報に[ヘルプリンク](#)を用意します。
- オブジェクトの名前とラベルは、オブジェクトの内容を明確に伝え、区別できるものを選択します。オブジェクトの本来の意味や他のオブジェクトとの違いをユーザーが考えなくて済むものにしてください。

### 間違った例:



### より良い例:



間違った例では、オブジェクト名が差異化されていませんが、より良い例では、製品名によって明らかな違いが示されています。

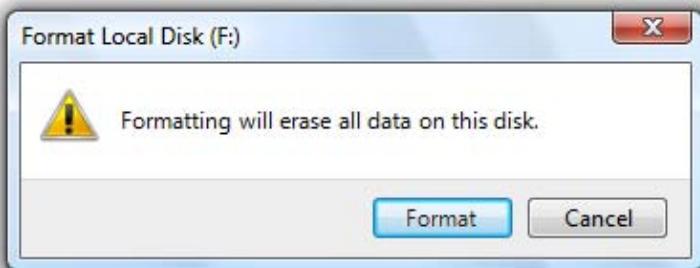
- ある操作に関する特定のテキストをユーザーが必ず読むようにするには、そのテキストを対話型コントロールに配置します。

### 許容される例:



この例では、確認事項が説明されたテキストを、ユーザーが読まない可能性があります。

より良い例:



この例では、ユーザーは少なくともディスクのフォーマットを実行しようとしていることを確認できます。

- 文と文との間には空白を 1 つ配置します (英語の場合)。2 つではありません。

テキストのフォント、サイズ、色

- 以下のフォントと色は、Windows の既定値です。

パターン	テーマ シンボル	フォント、色
Title bar text	CaptionFont	9 ポイント黒 (#000000) Segoe UI
Main instructions	MainInstruction	12 ポイント青 (#003399) Segoe UI
Secondary instructions	Instruction	9 ポイント黒 (#000000) Segoe UI
Normal text	BodyText	9 ポイント黒 (#000000) Segoe UI
Emphasized text	BodyText	9 ポイント黒 (#000000) Segoe UI、太字または斜体
Editable text	BodyText	9 ポイント黒 (#000000) Segoe UI、ボックス囲み
Disabled text	Disabled	9 ポイント濃い灰色 (#323232) Segoe UI
Link	HyperLinkText	9 ポイント青 (#0066CC) Segoe UI
Links (Hover)	Hot	9 ポイント薄い青 (#3399FF) Segoe UI
Document text	(なし)	9 ポイント黒 (#000000) Calibri
Document headings	(なし)	17 ポイント黒 (#000000) Calibri

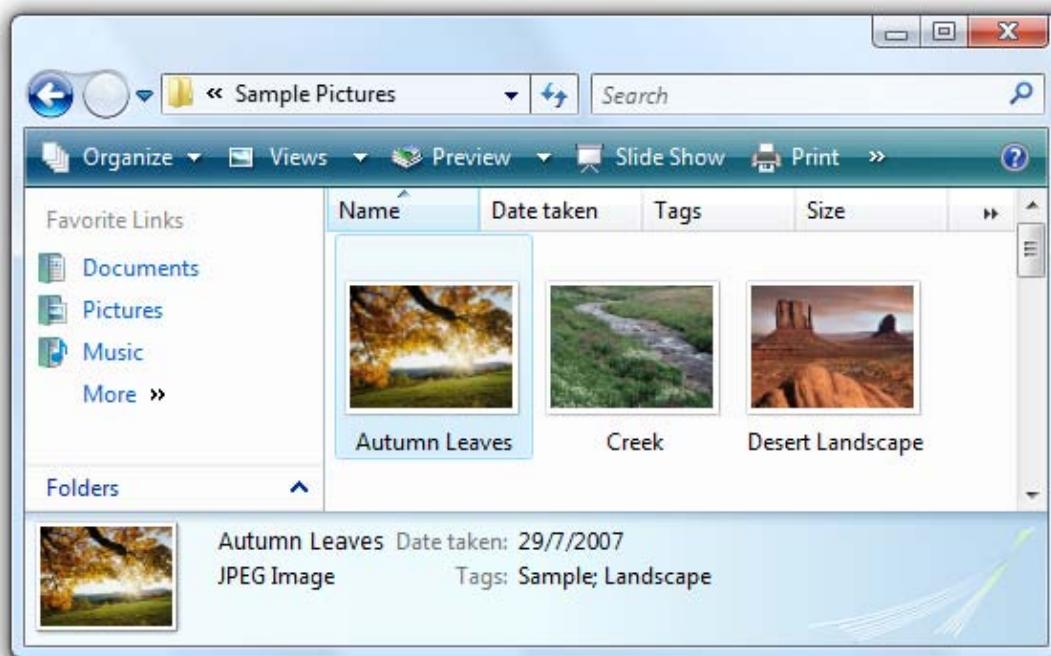
- 青色のテキストは、リンクとメイン指示テキストにのみ使用します。
- 緑色のテキストは、検索結果の URL にのみ使用します。

詳細と例については、「[フォント](#)」および「[色](#)」を参照してください。

## その他のテキストの特性

### 太字

- 太字は、必読のテキストに注目を集めるために、控えめに使用します。たとえば、ユーザーがラジオ ボタン オプションの一覧に目を通す場合、ラベルが太字で表示されていると、各オプションの補足テキストと見分けがつきやすいので、好感を持つと考えられます。乱用すると太字の効果が減少するので注意します。
- ラベル付きのデータでは、太字を使用してデータ全体にとって重要性の高い情報を強調します。
  - 一般的なデータ(数値や日付など、ラベルがないとほとんど意味を成さないデータ)の大半については、太字のラベルとプレーン データを使用し、ユーザーがデータの種類を簡単に確認して理解できるようにします。
  - 多くの、一見して内容がわかるデータについては、プレーン ラベルと太字のデータを使用し、ユーザーがデータそのものに注目できるようにします。
  - また別の方法として、太字によって重要性の高い情報を強調するのではなく、濃いグレーのテキストを使用し、重要性の低い情報を強調しないようにすることができます。



この例では、太字でデータを強調するのではなく、ラベルに濃いグレーを使用して強調されないようにしています。

- すべてのフォントで太字がサポートされるわけではないため、太字がなくてもテキストを理解できるようにします。

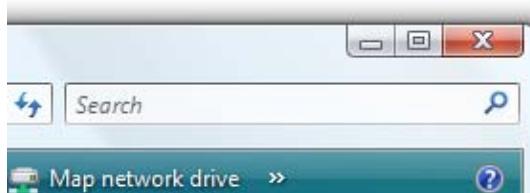
### 斜体

- テキストを文字どおりに参照させる場合に使用します(英語の場合)。この用途で引用符を使用しないでください。

正しい例:

The terms *document* and *file* are often used interchangeably. ("ドキュメント"と"ファイル"は、同義で使用されていることがよくあります。)

- テキスト ボックスや編集可能なドロップダウンリストの中のプロンプトにも使用します。



この例では、検索ボックスのプロンプトが斜体テキストとして書式設定されています。

- 特定の語を控えめに強調するのに使用して、理解しやすくします。

- すべてのフォントで斜体がサポートされるわけではないため、斜体がなくてもテキストを理解できるようにします。

## 太字斜体

- UI テキストでは使用しません。

## 下線

- リンク以外には使用しません。
- 強調を目的としては使用しません。代わりに、斜体を使用します (英語の場合)。

## 句読点

## 終止符

- コントロール ラベルまたはメイン指示テキストの末尾には配置しません。
- 補足指示テキスト、補足説明、またはその他の完全な文を構成している静的テキストの末尾に配置します。

## 疑問符

- 疑問文の末尾に配置します。終止符とは異なり、疑問符はあらゆる種類のテキストに使用します。

## 感嘆符

- 業務用アプリケーションでは使用しないようにします。
  - 例外: ダウンロード完了を示す場合 ("Done!") や、Web コンテンツに注意を引き付ける場合 ("New!") に、感嘆符が使用されることもあります。

## コンマ

- 3つ以上の項目を列挙する場合に、最後から2番目の項目の後ろに必ず配置します。

## コロン

- 外部コントロール ラベルの末尾にコロンを使用します。これは、アクセシビリティに特に重要なものです。一部の支援技術では、コントロール ラベルを識別する際にコロンを探します。
- 項目を列挙する直前に使用します。

## 省略記号

- 省略記号は不完全であることを意味します。次の UI テキストで、省略記号を使用します。
  - コマンド: コマンドに追加の情報が必要であることを示します。省略記号は、操作によって別ウィンドウが表示される場合に使用するのではなく、追加の情報が必要な場合にのみ使用します。詳細については、「[コマンドボタン](#)」を参照してください。
  - データ: テキストの一部が表示されていないことを示します。
  - ラベル: タスクが進行中であることを示します ("検索しています..." など)。

ヒント: ウィンドウやページに未使用の領域があるにもかかわらずテキストの切り詰めがある場合は、レイアウトに問題があるか、既定のウィンドウ サイズが小さすぎることを示しています。レイアウトや既定のウィンドウ サイズを修正して、テキストがすべて表示されるようにするか、より多くのテキストが表示されるようにしてください。詳細については、「[レイアウト](#)」を参照してください。

- 省略記号を対話型にしません。切り詰められたテキストを表示するには、代わりに[段階的表示コントロール](#)を使用します。

## 引用符とアポストロフィ記号

- テキストを文字どおりに参照させるには、引用符ではなく斜体を使用します (英語の場合)。
- 紛らわしくないようにする必要があるが、太字の書式設定ができない場合に限り、ウィンドウ タイトルとコントロール ラベルを引用符で囲みます。
- できる限り、直線型ではなく、曲線型の引用符やアポストロフィ記号を使用します (英語の場合)。
- 引用符は、二重引用符 (" ") を優先して使用し、一重引用符 (' ') は使用しないようにします。

## 正しい例:

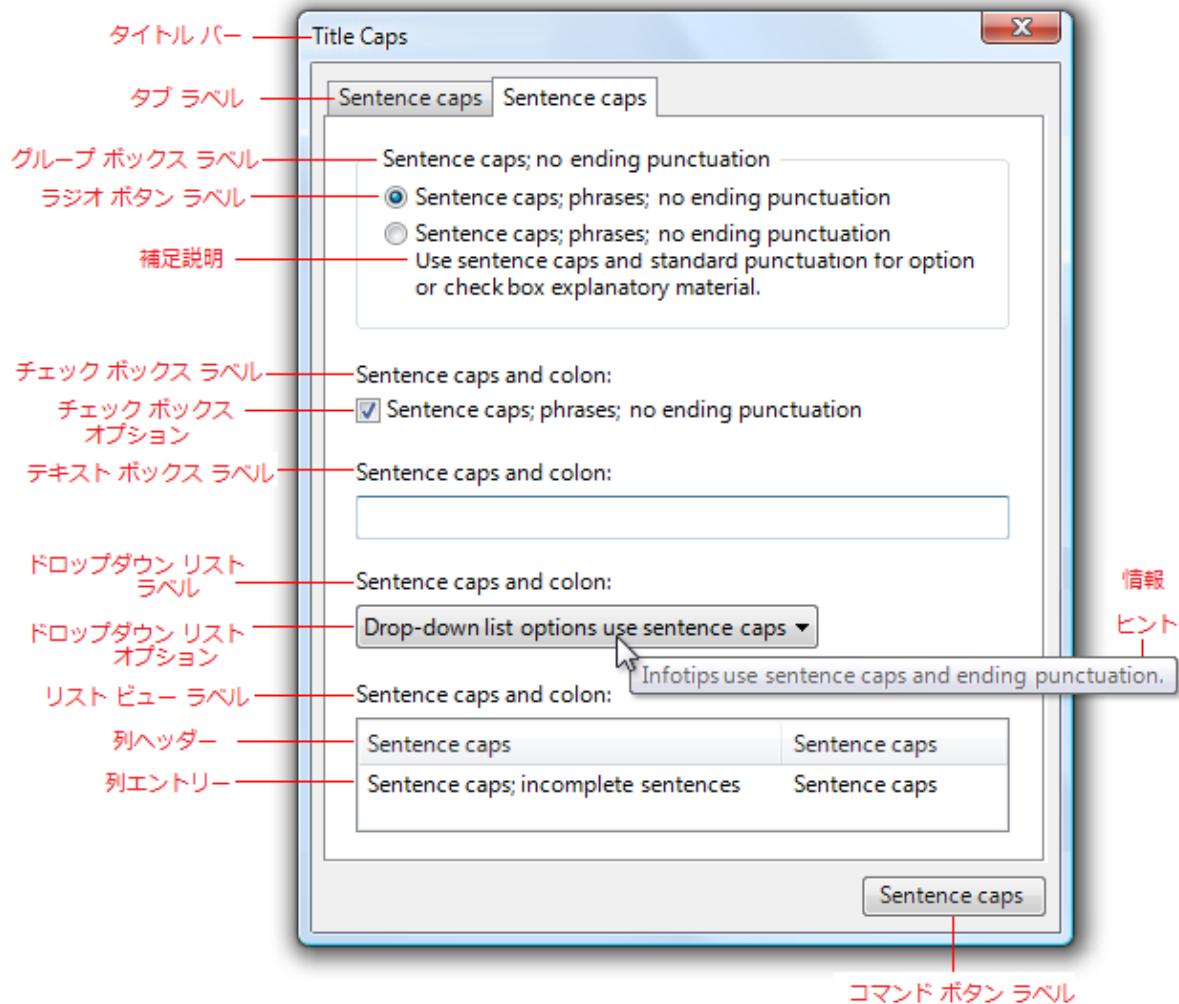
“Sparky’s cat フォルダー”を削除してよろしいですか?

## 間違った例:

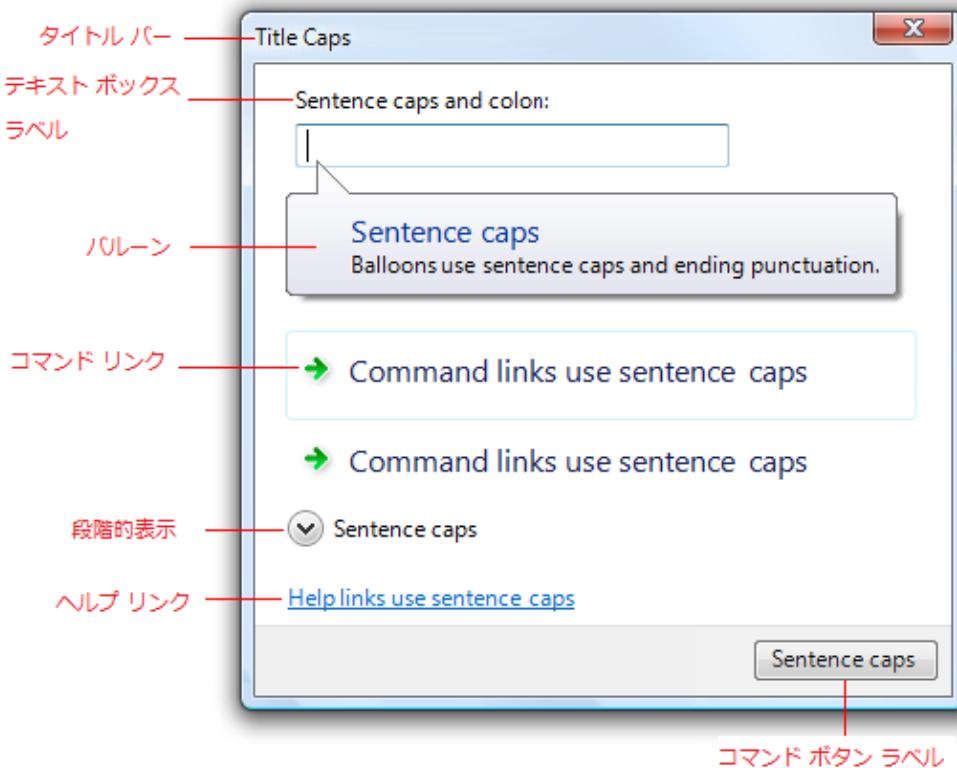
‘Sparky’s cat フォルダー’を削除してよろしいですか?

## 大文字化

- タイトルには[タイトルスタイルの大文字化](#)を使用し、その他すべてのUI要素には[センテンススタイルの大文字化](#)を使用します。これは[Windowsのトーン](#)に従っています。
  - 例外: 古いアプリケーションでは、大文字/小文字のスタイルが混在しないように、コマンドボタン、メニュー、列見出しにタイトルスタイルの大文字化を採用することもできます。



この一般的な例では、プロパティシート用の正しい大文字化と句読点を示しています。



この一般的な例では、ダイアログ用の正しい大文字化と句読点を示しています。

- 機能名やテクノロジーネームについては、大文字化は控えめにします。一般には、主要なコンポーネントのみを大文字化します(タイトルスタイルの大文字化を使用)。

#### 正しい例:

Analysis Services、cubes (キューブ)、dimensions (ディメンション)

*Analysis Services* は *SQL Server* の主要コンポーネントであるため、大文字化が適切です、*cubes* (キューブ) と *dimensions* (ディメンション) はデータベース分析ソフトウェアでよく使用される要素であり、大文字化の必要はありません。

- 機能名およびテクノロジーネームについては、大文字化に一貫性を持たせます。その名前が 1 つの UI 画面に何度も出現する場合、常に同じように表示する必要があります。同様に、プログラム内のすべての UI 画面で、その名前を同じように表示する必要があります。
- 汎用的なユーザーインターフェイス要素名は大文字で表記しません。たとえば、"toolbar (ツールバー)"、"menu (メニュー)"、"scroll bar (スクロールバー)"、"button (ボタン)"、"icon (アイコン)" などがあります。
  - 例外: Address bar (アドレスバー)、Links bar (リンクバー)。
- キーボード上のキーを表すのにすべて大文字の表記は使用しません。標準キーボードで使用される大文字化に従うか、キーボード上にラベルが付いていないキーには小文字を使用します。

#### 正しい例:

spacebar、Tab、Enter、PageUp、Ctrl + Alt + Del

#### 間違った例:

SPACEBAR、TAB、ENTER、PG UP、CTRL + ALT + DEL

- 強調する文字をすべて大文字で表すことはしません。こうすると読みにくくなり、"叫んでいる" ように受け取られる傾向にあることが、調査によりわかっています。警告する場合は、警告アイコンを使用し、明確な言葉で状況を説明します。たとえば、すべて大文字にして WARNING を追加する必要はありません。

詳細については、特定の UI コンポーネントのガイドランの "テキスト" または "ラベル" に関するセクションを参照してください。

#### グローバリゼーションとローカライズ

グローバリゼーションとは、すべての国や地域、文化圏で使用できるドキュメントまたは製品を作成することです。ローカライズとは、元の国や地域以外の場所で使用できるようにドキュメントまたは製品を適合させることです。UI テキストを記述する際は、グローバリゼーションとローカライズを考慮に入

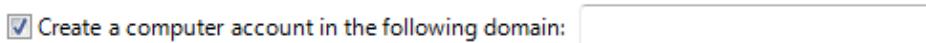
れます。作成したプログラムが他の言語に翻訳され、自国とは大きく異なる文化圏で使用されることがあります。

- 可変のコンテンツを含むコントロール(リストビュー、ツリービューなど)については、有効な最長データに適した幅を選択します。
- UI画面に30%の余白を残しておきます。ローカライズの対象となるすべてのテキスト(数値以外)について、30%(短いテキストの場合は最大200%)の余白を追加します。ある言語から別の言語に翻訳すると、多くの場合、テキストの行の長さが変わります。
- 実行時に部分的に文字列を組み合わせて文字列を構成しないようにします。代わりに、完全な文を使用して、翻訳する際にあいまいにならないようにします。
- 従属コントロール、それに含まれる値、または単位のラベルを使用して、文や句を作成しないでください。文の構造は言語によって異なるので、このように設計するとローカライズできなくなります。

間違った例:



正しい例:



この間違った例では、チェックボックスのラベル内にテキストボックスが配置されています。

- 文の一部にリンクを作成しないようにします。翻訳されるとそのテキストが残らないことがあります。リンクテキストはそれ自体で完全な文である必要があります。
  - 例外: 用語集へのリンクは文の一部としてインラインに挿入できます。

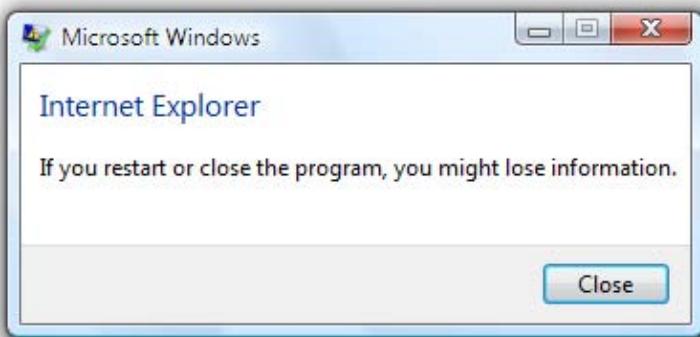
## タイトルバーのテキスト

- ウィンドウの種類に基づいてタイトルバーのテキストを選択します。
  - ドキュメント中心のプログラムの最上位ウィンドウ: "ドキュメント名 - プログラム名" の形式を使用します。ドキュメント中心の印象を与えるため、ドキュメント名を最初に表示します。
  - ドキュメント中心ではないプログラムの最上位ウィンドウ: プログラム名のみ表示します。
  - ダイアログボックス: ダイアログボックスの呼び出し元のコマンド、機能、またはプログラムを表示します。タイトルでダイアログボックスの目的を説明することはしません。説明はメイン指示テキストで行います。その他のガイドラインについては、「[ダイアログボックス](#)」を参照してください。
  - ウィザード: ウィザード名を表示します。"ウィザード"という語は、ウィザード名に含めないようにします。その他のガイドラインについては、「[ウィザード](#)」を参照してください。
- 最上位のプログラム ウィンドウの場合、タイトルバーのキャプションおよびアイコンがウィンドウの上部付近に明確に表示されている場合は、冗長にならないように、タイトルバーのキャプションおよびアイコンを非表示にします。ただし、Windows が使用するタイトルを内部で適切に設定する必要があります。
- ダイアログボックスの場合、タイトルに "ダイアログ" や "進行状況" という語は含めません。このような概念は暗黙的に示されているので、こうした言葉を除外する方がタイトルがすっきりします。

## メイン指示テキスト

- 表示されたウィンドウまたはページでユーザーがすべきことを簡潔に説明するために、メイン指示テキストを使用します。優れたメイン指示テキストは、単にUI操作に注目させるだけではなく、ユーザーの目的を明確にするものになっています。
- メイン指示テキストは、必須の指示や具体的な質問の形式で表現します。

間違った例:



この例では、メイン指示テキストは単にプログラム名を示しているだけです。ユーザーが特定の行動を起こすことを明示的に求めていません。

- 例外: エラー メッセージ、警告メッセージ、確認メッセージの場合は、メイン指示テキストに別の構文が使用されます。
- 可能な限り、具体的な動詞を使用します。ユーザーにとって、具体的な動詞(接続する、保存する、インストールする、など)は、汎用的な動詞(構成する、管理する、設定する、など)よりも有益です。
  - コントロール パネルのページとウィザード ページについては、具体的な動詞を使用できない場合、動詞を完全に省略することもできます。

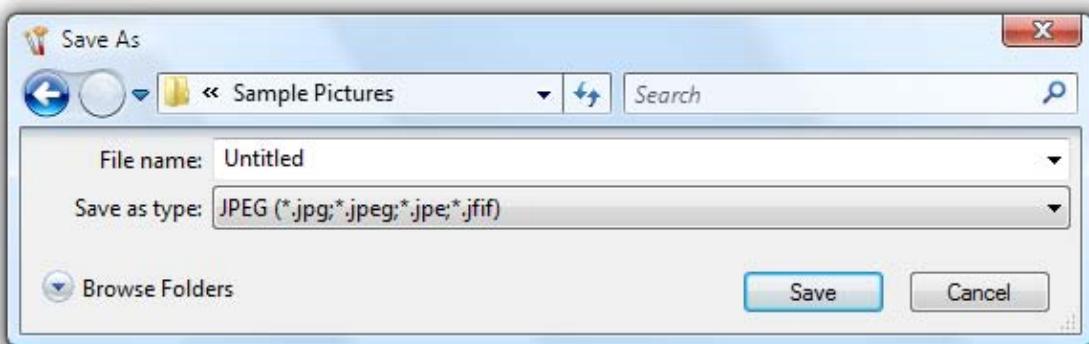
許容される例:

ロケール、地域、言語を入力してください

より良い例:

ロケール、地域、言語

- エラー メッセージや警告メッセージのようなダイアログについては、動詞を省略しません。
- メイン指示テキストを追加しても冗長になるだけの場合や、UI のコンテキストから明らかな場合は、メイン指示テキストを使用する必要はありません。



この例では、UI のコンテキストが既に明らかであるため、メイン指示テキストを付け加える必要がありません。

- 簡潔な 1 文だけを使用します。メイン指示テキストには必須の情報だけを含めます。さらに説明が必要な場合は、補足指示テキストを使用します。
- センテンス スタイルの大文字化**を使用します。
- 指示テキストが文の場合、文末の句点は含めません。指示テキストが疑問文の場合は、疑問符を付けます。
- 進行状況ダイアログ ボックスの場合、進行中の処理を簡潔に説明する語句を使用し、末尾に省略記号を付けてます。例: "写真を印刷しています..."
- ヒント: メイン指示テキストが適切かどうかを判断するには、ウィンドウまたはページの扱い方を友人に説明する場合を想像してみます。メイン指示テキストに対する応答が不自然であったり、不便であったり、使いにくかつたりする場合は、メイン指示テキストを変更します。

詳細については、特定の UI コンポーネントのガイドランの "メイン指示テキスト" に関するセクションを参照してください。

#### 補足指示テキスト

- 必要に応じて補足指示テキストを使用し、ウィンドウまたはページの理解や使用に役立つ追加の情報を提示します。その例を以下に示します。
  - ウィンドウの表示がプログラムまたはシステムによるものである場合は、表示の理由を説明するコンテキストを提示します。
  - メイン指示テキストに対してどう行動すべきかを決定するのに役立つ補足情報を提供します。
  - 重要な用語を定義します。
- 必要がない場合は補足指示テキストを使用しません。簡潔にまとめることができる場合は、メイン指示テキストですべての情報を伝えるようにします。
- メイン指示テキストの言い回しを少し変えて繰り返すことは避けます。追加する情報がない場合、補足指示テキストは省略します。
- 完全な文にし、センテンススタイルの大文字化を使用します。

## コントロール ラベル

- すべてのコントロール、コントロールのグループにラベルを設定します。次の場合は例外です。
  - テキストボックスおよびドロップダウンリストには、[プロンプト](#)を使用してラベルを設定することができます。
  - 通常、段階的表示コントロールにはラベルを付けません。
  - 従属コントロールには、関連するコントロールのラベルを使用します。スピンコントロールは常に従属コントロールです。
  - メイン指示テキストと同じ内容を繰り返すようなコントロールラベルは省略します。この場合、メイン指示テキストにアクセスキーを割り当てます。

許容される例:



この例では、テキストボックスのラベルで、メイン指示テキストと同じ内容が繰り返されています。

より良い例:



この例では、冗長なラベルは削除され、メイン指示テキストにアクセスキーが割り当てられています。

- ラベルの配置は次のとおりです。
  - パルーン、チェックボックス、コマンドボタン、グループボックス、リンク、タブ、ヒントでは、コントロールそのものによって直接ラベルを設定します。
  - ドロップダウンリスト、リストボックス、リストビュー、進行状況バー、スライダー、テキストボックス、ツリービューでは、上部、左揃え、または左側にラベルを設定します。
  - 通常、段階的表示コントロールにはラベルを付けません。シェブロンボタンには右側にラベルを設定します。
- リンクを除く、各対話型コントロールには一意のアクセスキーを割り当てます。詳細については、「[キーボード](#)」を参照してください。
- 簡潔なラベルにします。ただし、ラベルに1、2語追加すると明確になり、補足説明が不要になることもあります。
- ラベルは汎用的な内容ではなく、具体的になるようにします。ラベル以外のものを読まなくてもユーザーが意味を理解できることが理想です。

間違った例:

OK

正しい例:

Publish

正しい例では、コミットボタンに具体的なラベルが使用されています。

- ラジオボタンのようにラベルを列挙する場合は、同じ文法構造の表現を使用し、すべてのラベルの長さがおおよそ同じになります。
- ラベルを列挙する場合は、オプション間の違いがわかるラベルテキストにします。すべてのオプションが同じテキストで始まっている場合は、そのテキストをグループラベルに移動します。

間違った例:

Search

- Searching non-indexed locations include system directories
- Searching non-indexed locations include compressed files (ZIP,CAB...)

正しい例:

When searching non-indexed locations

- Include system directories
- Include compressed files (ZIP,CAB...)

正しい例では、同じ前置きの語句をラベルに移した結果、2つのオプションの差がより鮮明になっています。

- 一般に、肯定的な表現を優先して使用します。たとえば、"しません"の代わりに"します"を、"通知しません"の代わりに"通知します"を使用します。
  - 例外: チェックボックスのラベル"今後、このメッセージを表示しない"は広く使用されています。
- 特定の種類のコントロールにすべて適用される指示動詞は省略します。ラベルでは、そのコントロールに特有のこととに注目します。たとえば、テキストボックスコントロールでは"入力"が必要であることや、リンクを"クリック"することは言うまでもないことです。

間違った例:

Type your name:

Select your state:

[Click for more options](#)

正しい例:

Your name:

State:

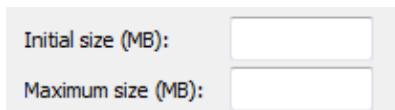
[More options](#)

間違った例では、コントロールのラベルに、その種類のコントロールにすべて適用される指示動詞が含まれています。

- 次のようにかっこを使用し、説明的な注釈をコントロールラベルに加えると役に立つ場合もあります。
  - 任意のオプションに対しては、ラベルに"(任意)"を追加することを検討します。
  - 強く推奨されるオプションに対しては、ラベルに"(推奨)"を追加します。この場合、この設定は任意であ

り、ただし、設定する必要があることを意味しています。

- 詳しい知識のあるユーザー向けのオプションに対しては、ラベルに "(詳細設定)" を追加することを検討します。
- ラベルの後ろに単位(秒、接続数など)をかっこで囲んで指定することができます。



Initial size (MB):

Maximum size (MB):

この例では、計測単位が MB(メガバイト)であることが示されています。

詳細については、特定の UI コンポーネントのガイドランの "テキスト" または "ラベル" に関するセクションを参照してください。

#### 補足説明

- コントロールに、ラベルで伝えるよりも多くの情報が必要な場合は、補足説明を使用します。ただし、必ずしも必要でない場合は補足説明を使用しません。コントロールのラベルで簡潔に伝えることができるのであれば、コントロールラベルですべてを伝えるようにします。通常、補足説明は、コマンドリンク、ラジオボタン、チェックボックスで使用します。
- 補足説明を付与する場合は、必要に応じて、コントロールラベルに太字を使用し、テキストを読みやすくします。



#### On (recommended)

This setting blocks all outside sources from connecting to this computer, except for those unblocked on the Exceptions tab.

#### Block all incoming connections

Select this option when you connect to less secure networks. All exceptions will be ignored and you will not be notified when Windows Firewall blocks programs.



#### Off

Avoid using this setting. Turning off Windows Firewall will make this computer more vulnerable to hackers or malicious software.

この例では、ラジオボタンのラベルが太字になっており、読みやすくなっています。

- グループ内の 1 つのコントロールに補足説明を追加した場合でも、そのグループ内のすべてのコントロールに説明を追加する必要はありません。関連情報は可能な限りラベルで提示し、必要がある場合にのみ説明を追加します。ラベルの言い回しを変えただけの補足説明は、一貫性を保つためであっても追加しないでください。

#### Standard user

(Users Group)

Standard account users can use most software and change system settings that do not affect other users.

#### Administrator

(Administrators Group)

Administrators have complete access to the computer and can make any desired changes. To help make the computer more secure, administrators are asked to provide their password or confirmation before making changes that affect other users.

#### Other:

Administrators



この例では、グループ内の 2 つのコントロールに補足説明が追加され、3 番目のコントロールには追加されていません。

- 補足説明をコマンドリンクの後ろに配置する場合、二人称で補足テキストを記述します。

• 例: コマンドリンク: [ワイヤレスネットワーク設定を作成して USB フラッシュ ドライブに保存する](#)

補足説明: USB フラッシュ ドライブを使用してルーターに転送可能な設定が作成されます。使用するワイヤレス ルーターが USB フラッシュ ドライブによる構成をサポートしている場合にのみ、これを実行します。

- 文を使用し、末尾に句点を付けます。

## コミット ボタンのラベル

次の表に、最もよく使用されるコミット ボタンのラベルとその使用方法を示します。

ボタン ラベル	意味	使用する場合	アクセス キー
OK	<ul style="list-style-type: none"> <li>ダイアログ ボックス: 変更の適用またはタスクのコミットを行い、ウィンドウを閉じます。</li> <li>親プロパティ ウィンドウ: 保留中の変更(ウィンドウが開かれた後のもの、または前回の[適用]ボタンのクリック以降のもの)を適用し、ウィンドウを閉じます。</li> <li>子プロパティ ウィンドウ: 変更を保留状態にし、ウィンドウを閉じて、オーナーウィンドウの変更が適用される際に変更を適用します。</li> </ul>	<ul style="list-style-type: none"> <li>タスクに特有でないウィンドウ(プロパティ シートなど)で使用します。</li> <li>ある特定のタスクの実行に使用するウィンドウには、代わりに、動詞から始まる特定のラベルを使用します(例: 印刷する)。</li> <li>ユーザーが変更を加えることができないウィンドウには、[閉じる]を使用します。</li> </ul>	Enter
[はい]/[いいえ]	[はい]は、"はい"か"いいえ"で答える質問への肯定的な応答であり、[いいえ]は否定的な応答です。	<ul style="list-style-type: none"> <li>"はい"か"いいえ"で答える質問に応答するだけの場合は、[はい]/[いいえ]ボタンを使用します。"はい"か"いいえ"で答える質問には、[OK]および[キャンセル]を使用しません。</li> <li>[はい]/[いいえ]ボタンよりも、具体的な応答を優先して使用します。[はい]/[いいえ]を使用しても問題はありませんが、具体的な応答を使用すると、即座に理解し、効率的に判断できます。</li> <li>ただし、具体的な応答にすると語句が長くなったり不自然になる場合は、"はい"/[いいえ]による応答の使用を検討します。</li> <li>"いいえ"という応答の意味が不明確な場合は、[はい]/[いいえ]ボタンを使用しないようにします。その場合は、代わりに具体的な応答を使用します。</li> <li>[はい]/[いいえ]ボタンは必ずペアで使用する必要があります。</li> </ul>	"Y" および "N"
キャンセル	<ul style="list-style-type: none"> <li>ダイアログ ボックス: 保留中の変更または進行中の作業をすべて破棄し、(重大な副次的な影響を残さず)元の状態に戻して、ウィンドウを閉じます。</li> <li>プロパティ シート: 保留中の変更(ウィンドウが開かれた後のもの、または前回の[適用]ボタンのクリック以降のもの)をすべて破棄し、ウィ</li> </ul>	<ul style="list-style-type: none"> <li>保留中の変更または操作をすべて破棄することができ、副次的な影響をすべて元に戻すことができる場合に使用します。</li> <li>破棄できない変更の場合は、[閉じる]ボタンを使用します。停止可能な進行中の操作には、[停止]を使用します。変更または操作が最初のうち破棄可能である場合、最初は[キャンセル]を使用し、その後、元に戻すことができなくなった時点で[閉じる]または[停止]</li> </ul>	Esc

	<p>ンドウを閉じます。</p> <ul style="list-style-type: none"> <li>コントロール パネル アイテム: 保留中の変更または進行中の作業をすべて破棄し、前の状態に戻して、このタスクの起動元であるハブページに戻ります。該当するハブページがない場合は、代わりにコントロール パネル アイテムのウィンドウを閉じます。</li> </ul>	ボタンに変更します。	
閉じる	ウィンドウを閉じます。変更も副次的な影響も破棄されません。	<ul style="list-style-type: none"> <li>変更または副次的な影響を破棄できない場合に使用します。メイン ウィンドウには、[キャンセル] の代わりに [閉じる] を使用します。</li> <li>ユーザーが変更を加えることができない ウィンドウには、[閉じる] を 使用します。</li> </ul>	Alt + F4、Ctrl + F4
停止	現在実行中のタスクを停止し、ウィンドウを閉じます。進行中の作業も副次的な影響も破棄されません。	<ul style="list-style-type: none"> <li>進行中の作業と副次的な影響を破棄できない場合や破棄しない場合(一般には進行状況バーまたはアニメーション)に 使用します。</li> </ul>	Esc
適用	<p>親プロパティ シート: 保留中の変更(ウィンドウが開かれた後のもの、または前回の [適用] ボタンのクリック以降のもの)を適用しますが、ウィンドウは開いたままになります。こうすることで、ユーザーがプロパティ シートを閉じる前に変更内容を評価できるようになります。</p> <p>子プロパティ シート: 使用しません。</p>	<ul style="list-style-type: none"> <li>プロパティ シートでのみ使用します。</li> <li>[適用] ボタンは、ユーザーが意味のある方法で効果を評価できる設定がプロパティ シートに(1つ以上)ある場合にのみ配置します。通常、[適用] ボタンは、設定によって目に見える変化が生じる場合に使用します。ユーザーが変更を適用し、その変更を評価して、その評価を基にさらに変更を行えるようにしてください。該当しない場合は、[適用] ボタンを無効にするではなく削除します。</li> </ul>	"A"
次へ	ウィザードおよび複数ステップのタスク: タスクをコミットすることなく、次の手順に進みます。	<ul style="list-style-type: none"> <li>ウィザードおよび複数ステップから成るタスクで、コミットすることなく次の手順に進む場合にのみ使用します。</li> <li>[戻る] をクリックすると、[次へ] ボタンによる影響を常に元に戻すことができる場合に使用します。</li> </ul>	"N"
終了(Finish)	ウィザードおよび複数ステップのタスク: ウィンドウを閉じます。タスクがまだ実行されていない場合は、タスクを実行します。タスクが既に実行されている場合は、変更も副次的な影響も破棄しません。	<ul style="list-style-type: none"> <li>ウィザードおよび複数ステップから成るタスクでのみ使用します。ただし、通常、これよりも具体的で適切なコミット ボタンがあるため、[終了] の使用は推奨されません。 <ul style="list-style-type: none"> <li>ボタンをクリックするとタスクをコミットする(つまり、タスクがまだ実行されていない)場合は、メイン指示テキストに対応した動詞で始まる具体的なラベルを使用します(例: 印刷する、接続する、開始する)。</li> <li>タスクがウィザード内で既に実</li> </ul> </li> </ul>	Enter

		<p>行されている場合は、代わりに [閉じる] を使用します。</p> <ul style="list-style-type: none"> <li>ただし、以下の場合は [終了] を使用することができます。           <ul style="list-style-type: none"> <li>具体的なラベルが汎用的である ("保存する"、"選択する"、"取得する" など)。</li> <li>タスクに単一の設定や設定の集まりの変更が含まれている。</li> </ul> </li> </ul>	
完了 (Done)	適用なし	<ul style="list-style-type: none"> <li>使用しません。コマンドとしての "Done" は文法的に正しくありません。</li> </ul>	適用なし

# スタイルとトーン

デザイン コンセプト  
ガイドライン  
Windows のトーンの使用  
実際に使われる言葉の使用  
正確性  
一貫性  
短縮形  
口語的表現、慣用句、俗語  
人称  
態  
ユーザーに対する姿勢  
文の構造と長さ

ライティングにおける "トーン" とは、書き手が読み手に伝える姿勢です。表現内容というより、むしろ表現方法の問題です。トーンは、読み手に特定の反応または感情を引き起こすためのものであり、トーン次第でユーザーの心を引き付けることも、突き放すこともできます。

注: ユーザー インターフェイスのテキストに関するガイドラインは、別の項目として記載しています。

## デザイン コンセプト

### Windows のトーン

Microsoft® Windows® のトーンを使用して、信頼感を与えるようにします。正確に、協力的に、洞察性と客觀性を持ち、ユーザー中心の姿勢を持って、ユーザーと個人レベルで対話するようにします。相手を混乱させるトーンや見下すトーン ("とにかく実行すること" など)、傲慢なトーンは使用しません。

"機械的" なトーン(言葉から話し手が見えない)や "営業的" なトーン(書き手が何かを販売しようとして相手をおだて、誘導し、あらゆるものを "簡単" と言いつくろう)は極力使用しません。

マイクロソフトの調査により、ソフトウェア業界の慣例として以下のことが明らかにされています。

- ユーザーが理解していないか誤って理解する可能性があるコンピューター用語や専門用語を必要以上に使っている。
- 用語の使用に一貫性がない。
- 理解できないメッセージや、意図せず見下したようなメッセージが使用され、ユーザーは理解するのに苦労している。

このような問題があると、ユーザーは当惑し、理解できずに落胆し、最終的にはソフトウェアを使用する気がなくなります。

ソフトウェア業界はこれまで「オンライン メッセージを配信するテクノロジーにばかり重点を置き、メッセージ自体の品質に費やす時間はあまりに少なすぎました」(Nick Usborne 氏『Getting the Message Right』)。テキストのトーンなど、メッセージの品質への関心を新たにすることで、言葉 자체が顧客満足全体に対する重要な要素であるということに気づきます。

初級ユーザーを中心に、技術知識のそれぞれ異なるユーザーをサポートして、プログラムで実行可能なことをすべて利用できるようにします。

### 最も重要な点

テキストの明確さ、自然さ、簡潔さを保ち、過度によそよそしくならないようにしながら、すべてのユーザーが理解できる用語を使用するようにします。

## ガイドライン

### Windows のトーンの使用

プログラムのトーンに関して、次の点を心がける必要があります。

- 正確性。その情報が技術的に正確であることに、ユーザーが安心できるようにします。情報が正確でない

と、その特定のタスクでの使用経験が損なわれ、その情報源から得られるサポート内容がすべて信用できなくなります。

- 協力的な姿勢。ソフトウェアによって何らかの実行が許可されるというのではなく、ユーザーが実行できるようになるということが伝わる言葉を使用します。たとえば、"Windowsによって可能となります"や"この機能により~できます"ではなく、"(ユーザーが)~できます"とします。(例外: 操作を許可したり禁止するセキュリティ機能などの場合は、"許可(allow)"を使用してもかまいません。)
- 洞察性。ユーザーは、アプリケーションとその提供者が、ある特定のタスクが複雑になる場合を把握し、手順を案内してくれるものと考えています。一方で、理解力はあるが、たまたま特殊な問題のサポートが必要になったユーザーとして扱われるとも考えています。
- 客観性。ユーザーは詳しい説明を必要としていることもありますが、多くの場合は、次に進むために必要なことを知りたいだけです。これには、目的(生産性、好奇心、楽しみ)がユーザーのものであり、書き手のものではないことを客観的に認識する必要があります。また、ユーザーに対する先入観を捨て去ることも必要です。
- ユーザーが中心。ユーザーの立場から、できればユーザーのためにできることを考えて記述します。ユーザーは、情報が適切であり、情報を受け入れられると感じます。

反対に、次のトーンは否定的な反応を引き起こすと考えられるので、使用しないようにします。

- 機械的なトーン。コンピューターやロボットと人間味や柔軟性のないやり取りをしているように感じます。
- 企業的なトーン。全権力を握り、すべてを知る非人格的な企業から長話を聞かされているように感じます。
- 法執行機関のトーン。立ち入った質問を次々に浴びているように感じます。
- 法律家のトーン。法的な意味を持つ行為の実行を求められているように感じます。
- 営業的なトーン。問題点をうまく取り繕って購入や試用を勧められているように感じます。
- 優越性、上位性、または険悪さを示すトーン。ソフトウェアがユーザーを軽く扱い、上位者の立場から発言しているように感じ、気分を害することもあります。このトーンは一般に、専門的で、ユーザーの誤りを必要に指摘することになり、無礼な感じを与えます。
- 傲慢なトーン。ソフトウェアが成果を自慢したり、過剰な注目を引き付けているように感じます。
- 軽薄なトーン。ユーザーの目的と感情が大切に扱われていないように感じたり、プログラムを使用することが当然であるかのような印象を与えます。

## 実際に使われる言葉の使用

- できれば日常的に使用する言葉を使用し、実際に使わない言葉の使用は避けます。複雑な技術上の概念または操作を説明する際には特に効果的です。ユーザーの肩越しにのぞき込み、タスクの実行方法を説明していく様子を想像します。

### 許容される例:

パスワードを変更するには、"この手続きをとります"。

### より良い例:

パスワードを変更するには、"次の手順に従います"。

- 可能であれば常に、短く、平易な言葉を使用します。言葉を短くする方が会話的であり、画面スペースが節約され、目を通しやすくなります。

### 許容される例:

"さらに"、ここでは～について説明します。

デジタルカメラでは微小なマイクロチップを"採用"しています。

デジタルカメラでは微小なマイクロチップを"利用"しています。

### より良い例:

ここでは、～について"も"説明します。

デジタルカメラでは微小なマイクロチップを"使用"しています。

- 新しく言葉を作ったり、標準的な言葉に新しい意味を当てはめないようにします。ユーザーにとって、言葉の既成の意味の方が、テクノロジー業界特有の特別な意味よりもなじみやすいと考えます。業界用語が必要な場合は、コンテキスト内に定義します。専門用語の使用は避けるようにしますが、一方でコンピューター

特有の一部の表現(ハッカー、CDを焼くなど)は既に日常会話の一部になっていることを忘れないようにします。

間違った例:

お気に入りを"カテゴリ化"するにはフォルダーを使用します。

正しい例:

お気に入りを"整理"するにはフォルダーを使用します。

- 単純な言葉の代用として記号を使用しないようにします。

間違った例:

ユーザー & コンピューター

ユーザー + コンピューター

ドメイン / ワークグループ

ユーザー #

正しい例:

ユーザーおよびコンピューター

ドメインまたはワークグループ

ユーザー数

## 正確性

- 明快な意味を持つ言葉を選択します。

許容される例:

*Since you created the table, you can make changes to it.* (表が作成できた"ので"、変更することができます。)

*'Keep your firewall turned on, as turning it off could create a security risk.* (ファイアウォールを無効にするとセキュリティ上のリスクが生じる可能性がある"ので"、ファイアウォールは有効にしておきます。)

より良い例:

*Because you created the table, you can make changes to it.* (表が作成できた"ので"、変更することができます。)

*Keep your firewall turned on, because turning it off could create a security risk.* (ファイアウォールを無効にするとセキュリティ上のリスクが生じる可能性がある"ので"、ファイアウォールは有効にしておきます。)

- 不要な言葉を省略し、1語で済むところに2語または3語使用しないようにします。

冗長な例:

パスワードを変更するためには、以下の手順を実行します。

より良い例:

パスワードを変更するには、次のようにします。

- 不要な副詞は省略します。

間違った例:

パスワードの変更は"それほど難しい"作業ではありません。

正しい例:

パスワードの変更は"難しい"作業ではありません。

- 動詞は、複合語よりも単一の単語を選択します。

許容される例:

コンピューターを"ロック ダウン"すると、…

より良い例:

コンピューターを"ロック"すると、…

- 動詞を名詞に変換したり、名詞を動詞に変換しません。

間違った例:

コンピューターの "パスワード保護" のためには、…

"接続確立" のためには、…

正しい例:

コンピューターをパスワードで "保護する" には、…

"接続する" には、…

## 一貫性

- 用語に一貫性があると、技術概念の理解が促進され、理解度を高めることができます。一貫性がないと、異なる言葉や操作が同じものを意味するかどうかをユーザーが判断する必要があります。例:  
switch (切り替え)、toggle (トグル)  
start (開始)、run (実行)、launch (起動)、boot (ブート)、execute (実行)  
enable (有効)、activate (アクティブ)、turn on (オン)  
burn (焼く)、copy (コピー)
- 構文に一貫性があると、ユーザーは予測できるようになります。予測できるようになると、一貫した構文が使用されたテキストを前よりも早く解析できます。たとえば、常に命令形で指示テキストを記述しておくと、ユーザーは特に命令文に注意するようになります。

## 短縮形

- 短縮形を使用すると、記述に簡潔で軽快な会話らしいリズムが生まれます。コンテキスト内で必要に応じて短縮形を使用します。製品名またはその他の固有名詞には使用しないでください。

## 口語的表現、慣用句、俗語

- 口語的表現や俗語は、製品ツアー、セットアップ画面、ローカライズされないコンテンツなど、特別な状況でのみ使用します。最近の調査により、ユーザーは思いがけない表現や聞き慣れた語句を好ましいと思うことがわかっています。口語的表現や俗語が使用されていると、効果的に翻訳するのが難しくコストがかかるので、使用することに効果があるかどうかを慎重に判断します。

例:

"一方で"、変更するべきではないレジストリには、ある種の価値があります。

SDSL を使用すると、回線がインターネット専用になり、ケーブル サービスを使用する場合のようなインターネット上の "交通渋滞" を体験することはありません。

## 人称

- 直接的、間接的にユーザーの行動に言及する場合は "you" を使用します (英語の場合)。
- ユーザーに操作を説明するには、二人称 (you/your) を使用します。多くの場合、二人称が默示的に含まれています。

例:

Choose the pictures you want to print. (印刷する画像を選択します。)

Choose an account (アカウントを選択します。) (默示的)

- ユーザーがプログラムに指示を与える場合は、一人称 (I/me/my) を使用します。

例:

Print the photos on my camera. (カメラの写真を印刷します。)

- "we" の使用は慎重に行います。一人称の複数形は、高圧的な組織や団体の存在を思わせます。ただし、アプリケーション名を使用するよりは好ましいといえます。 "it is recommended (推奨されます)" よりも "we recommend (推奨します)" を使用します。
- 三人称でユーザーに言及しないようにします。よそよそしくなり、当事者である感じが失われてしまします。

## 態

- 能動態を使用すると、人や物がその操作を実行することが強調されます。わかりにくく、型にはまったように聞こえる受動態よりも、より直接的で暖かみがあります。

**許容される例:**

Icons can be arranged by name in alphabetical order. (アイコンは名前でアルファベット順に "整理されます"。)

When a Personal Digital Assistant (PDA) or laptop is plugged in... (携帯情報端末 (PDA) またはラップトップが "接続される" と、...)

**より良い例:**

You can arrange icons in alphabetical order by the icon name. (アイコンをアイコン名でアルファベット順に "整理する" ことができます。)

When you plug in any Personal Digital Assistant (PDA) or laptop... (携帯情報端末 (PDA) またはラップトップを "接続する" と、...)

- 受動態は、語数が多くなったり構造上の不都合が生じるのを避けるためにのみ使用します。文の焦点が行為者よりも行為の方にある場合、主語が不明である場合、または、エラー メッセージ内で、能動態を使用すると主語であるユーザーがエラーを責められているように感じる場合に使用します。

**正しい例:**

新しいアイコンは左上隅に表示されます。

更新プログラムを有効にするには、ダウンロードおよびインストールされている必要があります。

- 肯定文で記述し、実行できないことよりも、実行できることを強調します。

**ユーザーに対する姿勢**

- ていねいに、サポートするように、また、やる気を起こさせるようにします。見下され、非難され、威圧されているとユーザーが感じることのないようにします。

**許容される例:**

新しいテキスト ドキュメントを削除できません。アクセスが拒否されました。

**より良い例:**

このファイルは保護されているため、特定のアクセス許可がなければ削除できません。

- 適正なバランスを取ります。ユーザーに対して、なれなれしくすることも、事務的になりすぎることもなく、好意的であるようにします。製品を初めて使用しようとしている友人を手伝っているところを想像してみます。このとき、親友や特別な人ではなく、近所の知り合いや家族の友人を想定します。ユーザーがプログラムを使用しながら快適さやくつろぎを感じられるようにしますが、言葉使いが無遠慮になつたり親くなりすぎないようにします。
- "please (してください)" は、以下に示すように、ユーザーに何らかの負担がかかる場合にのみ使用します。
  - 待機、タスクの繰り返し、プログラムの更新など、何らかの手間がかかることを実行するようにユーザーに求める。

**正しい例:**

"ファイルをコンピューターにコピーしています。しばらくお待ちください..."

- 機能不足、設計上の問題、プログラムのバグのために、ユーザーがタスクを完了できない。

**正しい例:**

指定された形式で保存できません。他の形式を選択してください。

- 顧客フィードバック プログラムへの参加やバグ報告の提出など、手数をかけさせる。

**正しい例:**

Fabrikam バックアップのエクスペリエンス向上のため、Fabrikam 顧客フィードバック プログラムに参加してください。

"してください" がなければぞんざいな感じを与えると思われる場合はいつでも使用します。

## Sign in

 Windows Live ID:   
(example555@hotmail.com)

 Please type your password.  
Password:

[Forgot your password?](#)

この例では、オンライン配置されたエラー メッセージに "please (してください)" がないとぞんざいな感じになります。

- "sorry (申し訳ございません)" は、ユーザーにとって重大な問題をもたらすエラー メッセージ内でのみ使用します (データ損失の場合、コンピューター使用を続行できない場合、技術サポートに問い合わせが必要な場合など)。プログラムの正常な動作の範囲内で発生した問題 (ネットワーク接続の検出に一定時間かかるなど) の場合は、謝罪表現は使用しないでください。

### 正しい例:

申し訳ございません。回復不可能な問題が検出されたため、Fabrikam バックアップを終了しました。

### 文の構造と長さ

- テキストは流し読みされることが多いため、1 語 1 語に気をつけます。文 (および段落) を平易かつ簡潔にすると、画面スペースを節約できるだけでなく、考え方や操作の重要性を最も効果的に伝えることができます。文を簡潔にするように最適に判断しますが、トーンが無骨になったり敵対的になったりすることがない程度の簡潔さにします。
- 繰り返しを避けます。各ウィンドウを確認し、語や文が重複しないようにします。重要なテキストは省かず、必要な場合はいつでも明示します。ただし冗長にならないようにし、自明のことは説明しません。
- 可能であれば、断片的な文を使用します。断片的な文は簡潔で強力です。一般に疑問文の形をとることが多く、ユーザーに直接訴えかけるにはよい方法です。

### 正しい例:

"My Photos" に加えた変更を保存しますか?

ファイルを保存したが、保存先を忘れた場合

- 必要な場合は、接続詞から文を開始します (さらに、ただし、あるいは)。
- 複雑な文の代わりにリストや表を使用します。リスト (番号/行頭文字) や表を使用すると、目を通しやすくなります。
- 同じ文法構造を使用します。同じ文法構造にするには、同じ役割を果たす語句を同じ形式で使用する必要があります。同じ重みの考えを表現する場合、また、機能が同質の UI 要素の場合 (見出し、ラベル、リスト、ページ タイトルなど) は、常に同じ文法構造を使用します。

### 正しい例:

Listen (聴く)

Watch (鑑賞する)

Share (共有する)

Collect (収集する)

上記の項目はすべて動詞の命令形の 1 単語であり、同じ文法構造になっています。

### 間違った例:

Music (音楽)

Video (ビデオ)

Share (共有する)

Listen (聴く)

上記では "Music (音楽)"、"Video (ビデオ)" が名詞であり、"Share (共有する)"、"Listen (聴く)" が動詞であるため、同じ文法構造ではありません。

## メッセージ

ここでは、Windows® ベースのアプリケーションで各メッセージを使用するためのガイドラインについて説明します。

- エラー
- 警告
- 確認
- 通知

## エラー メッセージ

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

提示方法

ユーザー入力エラー

トラブルシューティング

アイコン

段階的表示

今後、このメッセージを表示しない

既定値

ヘルプ

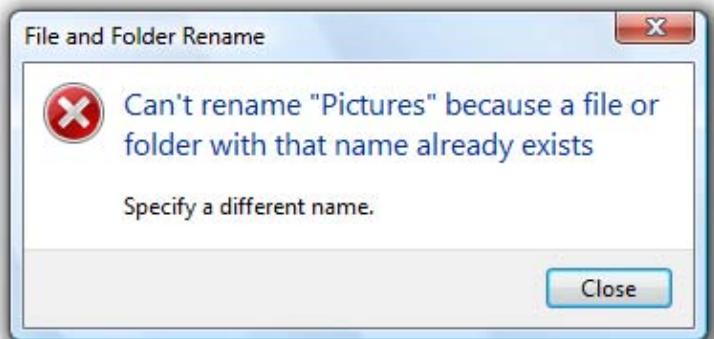
エラー コード

サウンド

テキスト

ドキュメント

"エラー メッセージ" は、既に発生した問題をユーザーに通知します。これに対して、"警告メッセージ" は、将来、問題を引き起こす可能性のある状況をユーザーに通知します。エラー メッセージの表示には、モーダル ダイアログ ボックス、インプレース メッセージ、通知、またはバルーンが使用されます。



典型的なモーダル エラー メッセージ。

効果的なエラー メッセージは、問題が発生したという事実に加え、発生した理由や、解決方法をユーザーに伝えます。ユーザーは、エラー メッセージの結果に応じて、何らかの対処をするか、自分の操作を見直すことになります。

完成度が高く、有益なエラー メッセージは、高品質のユーザー エクスペリエンスを実現するうえで不可欠な要素です。完成度の低いエラー メッセージは、製品に対する満足度を低下させ、無駄なテクニカル サポート コストにつながります。不要なエラー メッセージは、ユーザーの操作の流れを妨げることがあります。

注: ダイアログ ボックス、警告メッセージ、確認、標準アイコン、通知、レイアウトに関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- ユーザーインターフェイス (UI) に表示されているのは、既に発生した問題か。該当しない場合は、エラー メッセージとはいえません。将来、問題を引き起こす可能性のある状況をユーザーに伝える場合は、警告メッセージを使用します。
- 混乱させることなく問題を防ぐことができるか。該当する場合は、問題そのものを回避するようにします。たとえば、想定外の値を入力できるコントロールの使用は避け、有効な値の範囲しか受け付けない制約付きコントロールを使用すれば、エラー メッセージを表示する必要はありません。また、クリックすることによってエラーになるような場合は、コントロールを無効化します。コントロールが無効化されていても、その理由が明らかに伝わるのであれば、これも 1 つの方法です。
- 問題を自動的に修正できるか。該当する場合は、問題を処理し、エラー メッセージを表示しません。

ユーザーがメッセージの結果に応じて、何らかの対処をするか、自分の操作を見直す可能性はあるか。該当しない場合は、あえてユーザーの操作を中断する必要はなく、エラーを表示しない方がよいと判断できます。

- ユーザーが他のプログラムをアクティブに使用しているときにも、同じ問題が当てはまるかどうか。該当する場合は、[通知領域アイコン](#)を使用して問題を表示するようにします。
- 現在ユーザーが行っている操作に関連した問題ではなく、特に今すぐ何らかの対処を要するものではないので、ユーザーが無視してもかまわないかどうか。該当する場合は、エラー メッセージではなく、[操作の失敗通知](#)を使用します。
- メイン ウィンドウ内のバックグラウンド タスクの状態に関連した問題か。該当する場合は、[ステータスバー](#)を使用して問題を表示するようにします。
- 主な対象ユーザーが IT プロフェッショナルであるかどうか。該当する場合は、[ログ ファイル](#) エントリや電子メール通知などのフィードバックのしくみを使用します。IT プロフェッショナルの場合、重大ではない情報にはログ ファイルを使用する方がいっそう好まれます。

## デザイン コンセプト

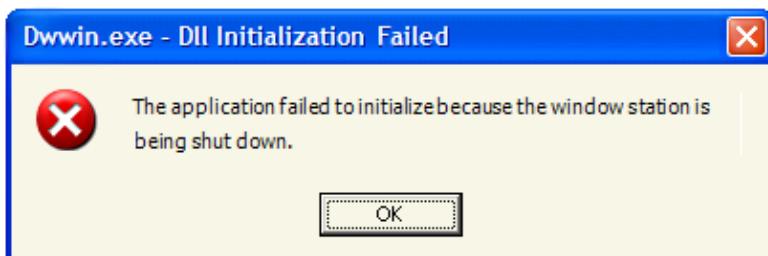
### 完成度の低いエラー メッセージの特徴

残念ながら、わざらわしく、役に立たない、完成度の低いエラー メッセージは数多く存在します。エラー メッセージはモーダル ダイアログ ボックスを使用して表示されるため、ユーザーが現在行っている操作が中断され、メッセージに対して同意しなければ、操作を続行できません。

完成度が低いといつても、間違いは 1 つではなく、さまざまなケースが考えられます。次に、不名誉の殿堂に入っているエラー メッセージの例を見ていきます。

### 不要なエラー メッセージ

#### 間違った例:



この例は、Windows® XP のエラー メッセージですが、最も完成度の低いものの 1 つといえます。Windows 自体がシャット ダウン処理の途中なので、プログラムを起動できなかったという意味です。これに関してユーザーが行えることはありません。また、ユーザーの最終的な目的は Windows をシャット ダウンすることなので、何かをする理由もありません。しかも、このエラー メッセージを表示することで、Windows も自らシャット ダウンを妨げていることになります。

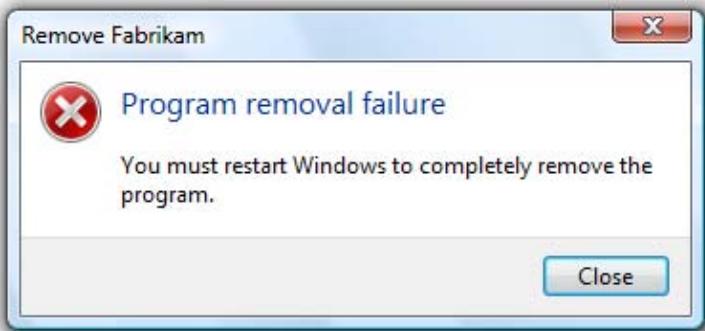
**問題:** エラー メッセージそのものが問題です。エラー メッセージを無視する以外、ユーザーにできることはありません。

**主な原因:** ユーザーの意図や視点を考えず、発生したエラーをすべて報告しようとしたことが原因です。

**推奨される代替策:** ユーザーが気に掛ける必要のないエラーは報告しないようにします。

### "成功" しているにもかかわらず表示されるエラー メッセージ

#### 間違った例:



このエラー メッセージは、プログラムの削除の直後に、ユーザーが *Windows* の再起動を拒否した場合に表示されます。ユーザーの視点から見れば、プログラムの削除は成功しています。

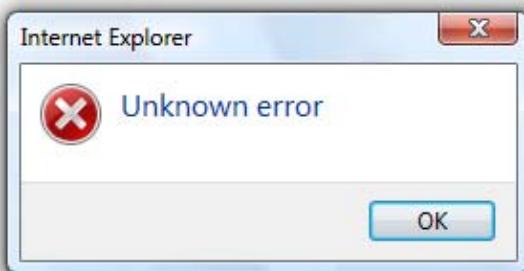
問題: ユーザーの視点から見れば、エラーは発生していません。エラー メッセージを無視する以外、ユーザーにできることはあります。

主な原因: ユーザーの視点から見ると作業は正常に完了していますが、アンインストール プログラムの視点で見ると確かに成功とはいえません。

推奨される代替策: ユーザーにとって容認可能な範囲であれば、エラーは報告しないようにします。

まったく意味のないエラー メッセージ

間違った例:



エラーが発生したことはわかりますが、ユーザーには、それが何なのか、また、何をすればよいのかがまったくわかりません。当然、[OK] ボタンをクリックしたいはずがありません。

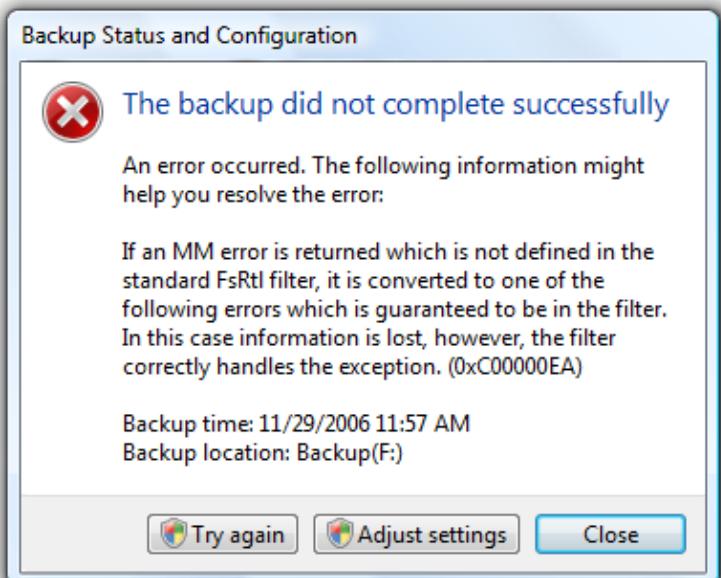
問題: エラー メッセージに具体的な問題が記述されておらず、ユーザーにできることは何もありません。

主な原因: ほとんどの場合、プログラム側のエラー処理が不完全であることが原因です。

推奨される代替策: 適切に設計されたエラー処理をプログラムに実装するようにします。

わかりにくいエラー メッセージ

間違った例:



この例では、問題を示す文は明瞭ですが、補足説明があまりに難解です。

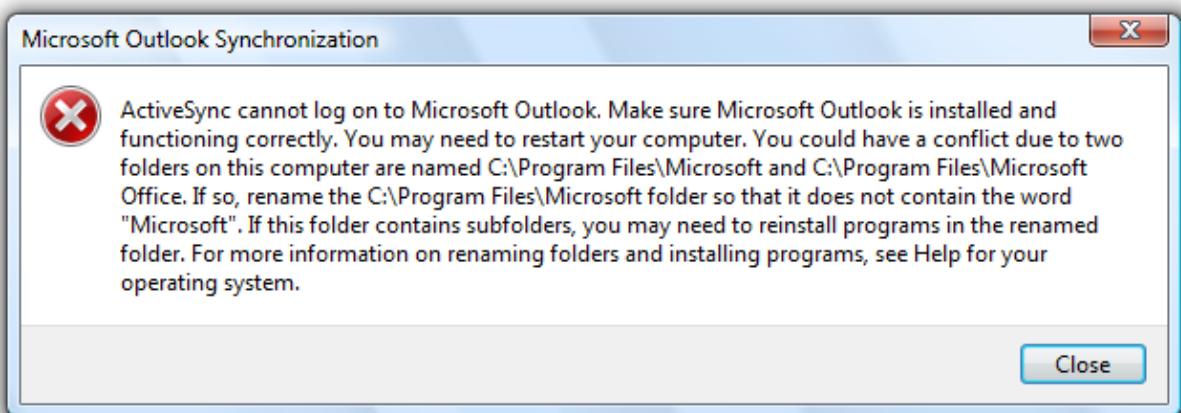
問題: 問題を示す文または解決方法が難解です。

主な原因: ユーザーの視点からではなく、コードの視点から問題を説明しています。

推薦される代替策: エラー メッセージの内容は、対象ユーザーが容易に理解できるように記述してください。ユーザーが実際に行うことのできる解決策を提供します。プログラムのエラー メッセージを 1 つの "エクスペリエンス" と考えて、適切に設計してください。プログラマが場当たり的にエラー メッセージを作成することは避けるようにします。

#### 情報量の多すぎるエラー メッセージ

間違った例:



この例では、エラー メッセージに記述するトラブルシューティング手順としては、明らかに細かすぎます。

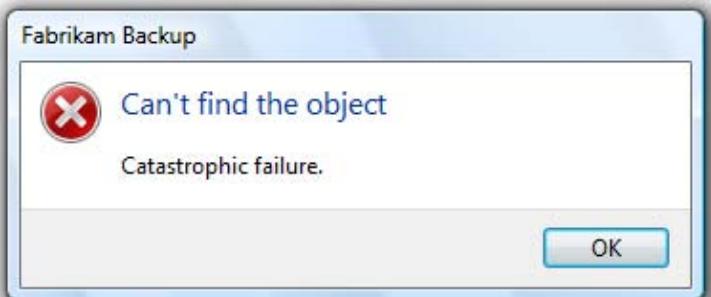
問題: 情報量が多すぎます。

主な原因: 複雑なトラブルシューティング プロセスをエラー メッセージ内で詳しく説明しようとしたことが原因です。

推薦される代替策: 不要な情報は省略します。トラブルシューティング ツールの使用も避けてください。トラブルシューティング ツールが必要な場合は、最も可能性の高い解決策を中心に扱い、それ以外は、対応するヘルプ トピックへのリンクを設けるのみにとどめます。

#### 不要で不親切なエラー メッセージ

間違った例:



オブジェクトが見つからないことと、"致命的"であることが結びつきません。また、致命的なエラーへの応答として、[OK] ボタンしか存在しないのでは、あまりに不親切です。

問題: 語調が必要に大げさで不親切です。

主な原因: 問題の原因は、プログラムの視点から見た場合の致命的なバグです。

推奨される代替策: 言葉の選択はユーザーの視点で慎重に行うようにします。

ユーザーに責任を押し付けるようなエラー メッセージ

間違った例:



ユーザーに過失があるかのような表現が使用されています。

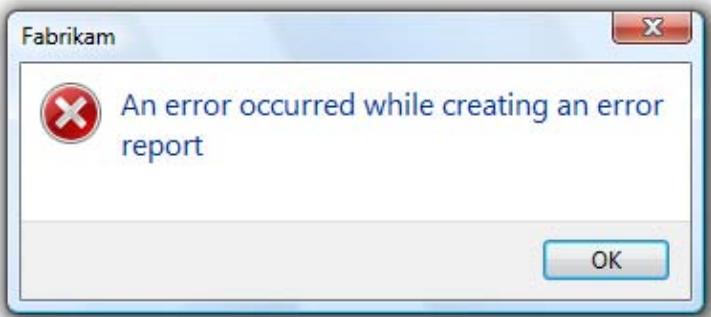
問題: エラー メッセージの表現に問題があります。エラーの原因が、いかにもユーザーに非があるかのような表現です。

主な原因: 問題そのものではなく、ユーザーの行動に焦点を当てた無神経な表現が使用されています。

推奨される代替策: 問題を引き起こしたユーザーの操作ではなく、問題そのものに焦点を当てます。必要に応じて受動態を用いるなど、間接的な表現を心がけます。

意味のないエラー メッセージ

間違った例:



この例では、問題を示す文が非常に風刺的で、解決策も提示されていません。

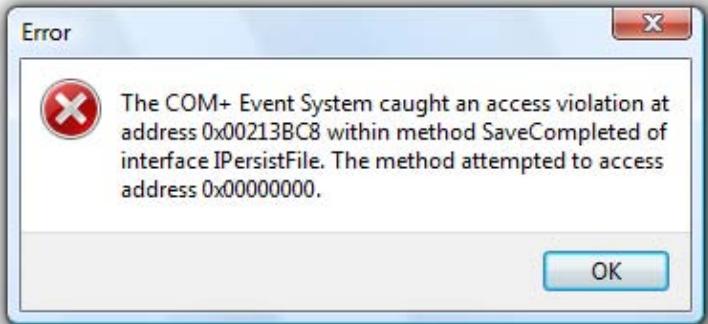
問題: エラー メッセージの文に意味がなく、不合理な内容です。

主な原因: コンテキストをよく考えずにエラー メッセージが作成されています。

推奨される代替策: エラー メッセージの作成と校閲をライターに依頼します。エラーを校閲する際は、コンテキストとユーザーの心境を考慮するようにしてください。

## プログラマ向けのエラー メッセージ

間違った例:



この例のエラー メッセージは、プログラムにバグがあることを示しています。プログラマにしか意味のないエラー メッセージです。

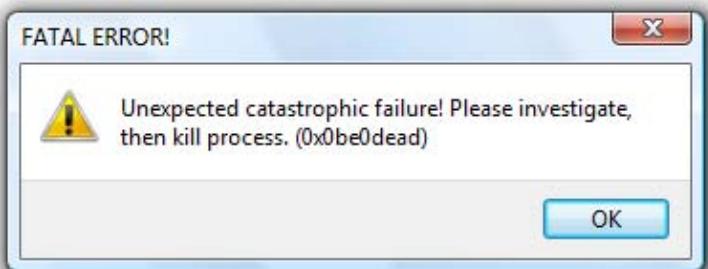
問題: プログラムの開発者がバグを特定するためのメッセージが、プログラムのリリース バージョンに残っています。こうしたエラー メッセージは、ユーザーにとって何の意味も価値もありません。

主な原因: プログラマが自分用のメッセージとして、通常の UI を使ったことに原因があります。

推奨される代替策: このようなメッセージは、製品のリリース バージョンから自動的に削除されるように、必ず条件付きコンパイルを使用してください。プログラマにしか意味のない内容なので、そのようなエラーをユーザーにわかりやすく記述しようとして無駄な時間を費やすことは避けましょう。

明らかに間違いのあるエラー メッセージ

間違った例:



この例には、メッセージの表示として、陥りやすいミスが多数存在します。

問題: エラー メッセージに表示されている情報は完全に誤りです。

主な原因: エラー メッセージのガイドラインが守られていません。このエラー メッセージは、ライター やエディターによって書かれたものではなく、校閲もされていません。

エラー処理の性質上、こうしたミスは頻繁に起こります。不名誉の殿堂入りしそうな完成度の低いエラー メッセージが多いことに驚きます。

完成度の高いエラー メッセージの特徴

これまで紹介してきた悪い例と比べて、完成度の高いエラー メッセージには、次のような情報が盛り込まれています。

- 問題。問題が発生したことを見事に示します。
- 原因。問題が発生した理由を説明します。
- 解決策。ユーザーが問題を解消できるように、解決策を提供します。

加えて、完成度の高いエラー メッセージには、次のような特徴があります。

- 関連性が高い。メッセージにユーザーが気に掛けている問題が反映されています。
- 実行性が高い。ユーザーがメッセージの結果に応じて、何らかの対処をするか、自分の操作を見直すことができる必要があります。
- ユーザーが中心。コードの問題ではなく、対象ユーザーの操作や目的という観点から問題が説明されています。

- 簡潔である。メッセージが最小限に、かつ短すぎないように書かれています。
- 明確である。対象ユーザーが問題と解決策を理解しやすいように、わかりやすい言葉で書かれています。
- 具体的である。関連するオブジェクトの名前、場所、値などをはっきりと示しながら、問題が具体的な言葉で説明されています。
- 丁寧である。ユーザーに非があるかのような言い回しや、ユーザーを見下した表現は使用しません。
- めったに表示されない。表示する頻度はできるだけ少なくします。高い頻度で表示されるエラー メッセージは、設計に問題がある印象を与えます。

完成度の低いエラー メッセージでプログラムの評判を落とすようなことのないように、以上の特徴を踏まえながらエラー処理を設計してください。

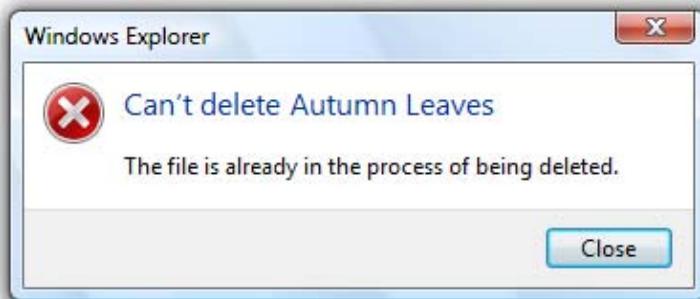
#### 不要なエラー メッセージを回避する

最も適切なエラー メッセージとは、決して発生しないエラー メッセージです。エラーの多くは、より良い設計をすることによって防ぐことができます。また、エラー メッセージよりも適切な方法が見つかることもあります。一般に、エラーは報告するよりも、防止するに越したことはありません。

最も避けるべきエラー メッセージは、対処のしようがないエラー メッセージです。ユーザーがメッセージを無視する以外に具体的な行動や変更を行うことができないのであれば、エラー メッセージを表示しないようにします。

ユーザーの視点で見ればまったく問題がなく、完全に排除してもかまわないエラー メッセージもあります。たとえば、既に削除処理が開始されているファイルを削除しようとすると、エラー メッセージが表示されることがあります。確かにコードの視点から見ると予期しないケースですが、ユーザーにとっては、最終的に目的が達成されるのでエラーとはいえません。

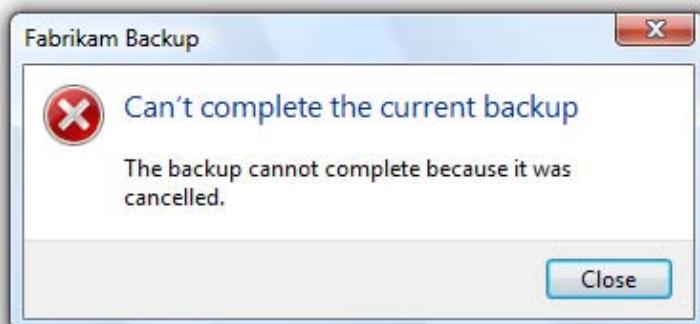
#### 間違った例:



ユーザーの視点から見ると処理は成功しているため、このようなエラー メッセージは表示しない方がよいと判断できます。

もう1つの例を考えてみます。ユーザーが明示的にタスクをキャンセルするケースです。ユーザーの視点から見ると、次の状況はエラーではありません。

#### 間違った例:



先ほどと同様、ユーザーの視点から見ると処理は成功しているため、このようなエラー メッセージは表示しない方がよいと判断できます。

テクノロジー中心ではなく、ユーザーの目的を中心に考えることによって、エラー メッセージを排除できる場合もあります。そのためには、エラーとは何なのかをもう一度よく考えることが必要です。

ユーザーの目的に関する問題なのか、または、それを達成するうえでのプログラムの能力的な問題のかを考えます。ユーザーの操作が現実世界において意味のある行動であれば、ソフトウェアでも同様に意味のある行動とする必要があります。

たとえば、e コマース プログラムの検索機能を使って製品を探したところ、入力したキーワードと完全に一致する製品が見つからないか、目的の製品が在庫切れだったとします。技術的にはエラーになりますが、エラーメッセージを表示する代わりに、プログラムで次のような処理を行うことが考えられます。

- 検索に対して最も近い条件を満たす製品を続けて検索します。
- 検索に明らかな誤りがある場合、修正候補を自動的に表示します。
- スペルミス、つづり字異形、名詞の単複に対する動詞の活用形の誤りなど、一般的な問題は自動的に処理します。
- 製品の入荷予定日を表示します。

ユーザーの要求が理にかなったものである限り、適切に設計された e コマース プログラムは、それに見合った結果を返す必要があります。ただエラーを返すだけのプログラムは適切とはいえません。

効果的にエラーメッセージを防ぐ方法は他にもあります。問題の発生そのものを防ぐことです。たとえば、次のようにしてエラーを防ぐことができます。

- 制約付きコントロールを使用する。有効な値しか入力できないコントロールを使用します。リスト、スライダー、チェックボックス、ラジオボタン、カレンダーコントロールなどのコントロールは、もともと有効な値の範囲が決まっていますが、値を自由に入力することができるテキストボックスでは、どうしてもエラーメッセージが必要となります。ただし、テキストボックスでも、特定の文字しか入力できないようにしたり、入力できる文字数を制限することは可能です。
- 対話操作を制限する。ドラッグ操作の場合、ドロップできる場所を制限します。
- コントロールやメニュー項目を無効にする。無効化されている理由をユーザーが容易に推測できる場合は、コントロールやメニュー項目を無効にします。
- 適切な既定値を設定する。既定値を表示することでユーザーの入力エラーを減らすことができます。値を変更する必要がある場合でも、既定値があれば、適切な入力形式を推測しやすくなります。
- 何もしなくても適切に動作するように配慮する。作業が自動的に実行されるか、作業そのものが不要であれば、ユーザーがミスをする可能性は低くなります。ユーザーが小さなミスをしたとしても、その意図がはっきりしていれば、問題を自動的に解決することが可能です。たとえば、書式に関する小さな問題は、自動的に修正することができます。

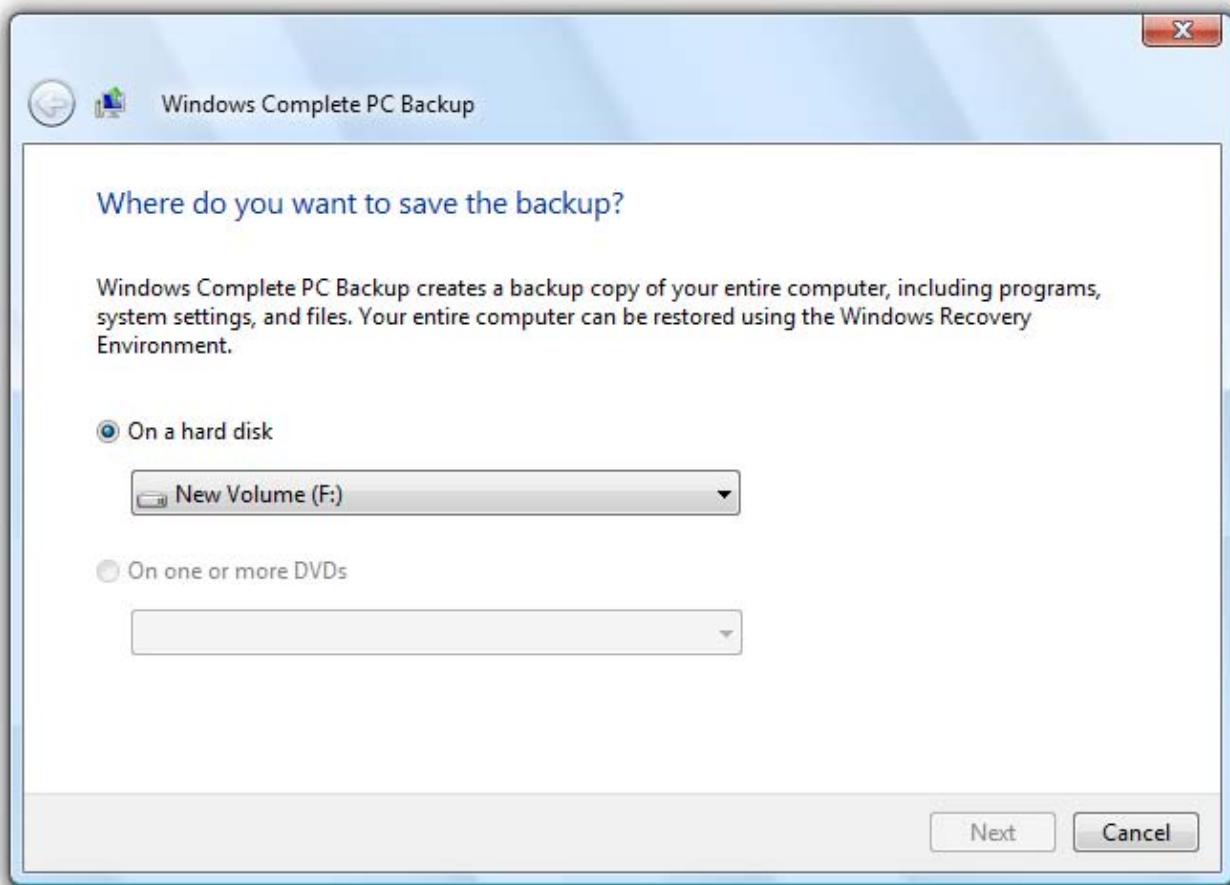
### 必要なエラーメッセージを提供する

本当に必要なエラーメッセージもあります。たとえば、ユーザーが何か誤った操作をした、ネットワークやデバイスが機能停止に陥った、オブジェクトが見つからない/変更されている、タスクを完了できない、プログラムにバグがあるなど、さまざまな問題が考えられます。こうした問題は回避するに越したことはありません。ソフトウェアを適切に設計することによって、ユーザーのさまざまなミスを防ぐことができる場合も確かにあります。しかし、こうした問題をすべて防ぐことは実際には困難です。また、こうした問題の 1 つが発生したとしても、エラーメッセージが適切であれば、ユーザーをすばやく適切な方向に導くことができます。

一般に、エラーメッセージは最悪のユーザー エクスペリエンスであって、確実に防ぐべきであると考えられています。しかし、正確にはそうではありません。最悪なのは、ユーザーを混乱させることであり、あらゆる手段を尽くして防ぐ必要があります。その手段の 1 つが、適切なエラーメッセージです。

コントロールの無効化について考えてみましょう。ほとんどの場合、コントロールが無効化されている理由は明確であるため、コントロールの無効化はエラーメッセージを回避する手段としては最適です。しかし、コントロールが無効化されている理由が明確でないとしたらどうでしょうか。ユーザーは先に進むことも、問題を特定するためのフィードバックを得ることもできません。ユーザーは立ち往生することでしょう。問題を推測するか、テクニカルサポートに問い合わせるしか方法がありません。このような場合は、コントロールを有效地にしたままで、適切なエラーメッセージを表示した方がはるかに効果的です。

間違った例:



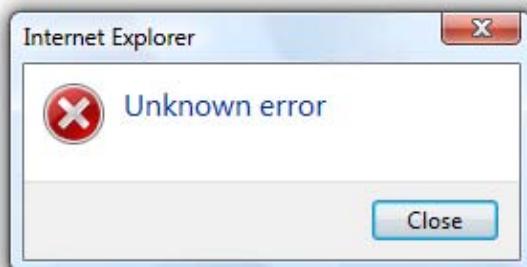
[次へ] ボタンが無効化されているのはなぜでしょうか。ユーザーを混乱させないためにも、この場合はボタンを有効にしておき、適切なエラー メッセージを表示する方がよいでしょう。

エラー メッセージを表示するかどうか迷った場合は、まず、実際にエラー メッセージを作成してみることです。ユーザーがその結果に応じて、何らかの対処をするか、自分の操作を見直す可能性が高いと判断できる場合は、エラー メッセージを提供するようにします。これに対して、ユーザーがメッセージを無視する以外に具体的な行動や変更を行うことができないのであれば、エラー メッセージを表示しないようにします。

#### 適切なエラー処理のための設計

完成度の高いエラー メッセージ テキストを作成したくても、プログラム側で適切なエラー処理がなされていないために、それができない場合もあります。たとえば、次のエラー メッセージを見てください。

間違った例:



おそらく、プログラム側のエラー処理が不十分であるため、問題が何なのか本当にわからないものと思われます。

エラー メッセージの文そのものに問題がある可能性もありますが、むしろ、根本的にコードでエラー処理が適切になされていない、つまり、問題について具体的な情報が何もないことが、この文からは見て取れます。

具体的で実行性が高く、ユーザーの立場に立ったエラー メッセージを作成するには、プログラムのエ

ラー処理コードによって、具体的で細分化されたエラー情報が提供される必要があります。その例を次に示します。

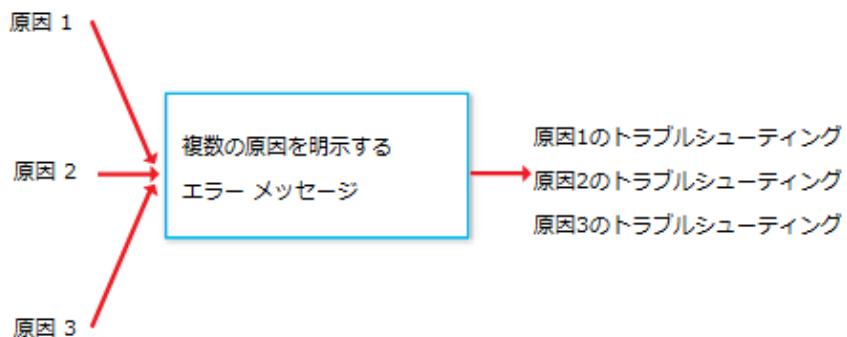
- 問題ごとに一意のエラー コードを割り当てる。
- 問題に複数の原因がある場合は、可能な限り、プログラム側で具体的な原因を特定する。
- 可変要素を伴う問題の場合は、それをパラメーターとして維持する。
- 低水準の問題は、高水準にして処理することで、ユーザーの視点からエラー メッセージを表示できるようにする。

完成度の高いエラー メッセージは、単に UI だけによるものではありません。ソフトウェア設計の問題でもあります。完成度の高いエラー メッセージ エクスペリエンスは、後から間に合わせで追加できるものではありません。

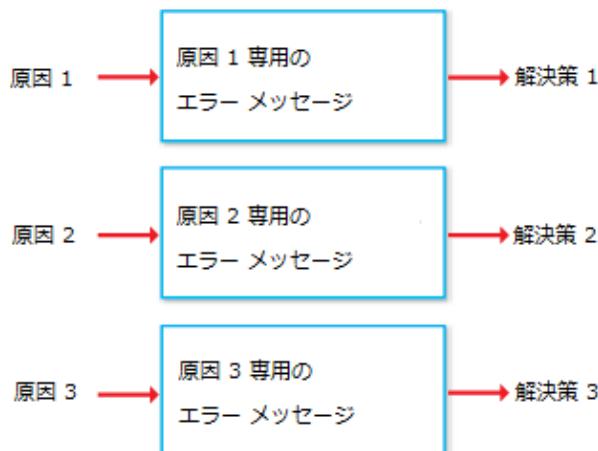
### トラブルシューティング (およびその防止法)

トラブルシューティングの必要性が生じるのは、1つのエラー メッセージで複数の原因が報告されたときです。

間違った例:



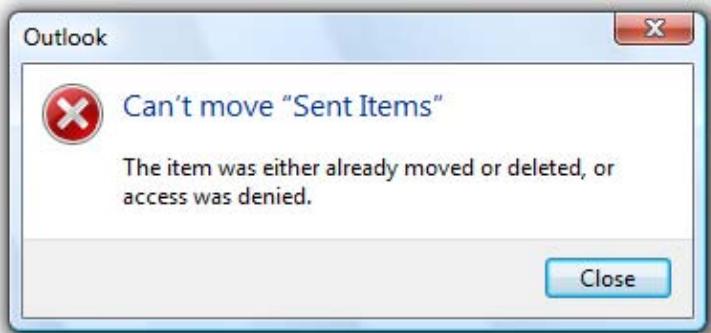
正しい例:



1つのエラー メッセージで複数の問題が報告された場合、トラブルシューティングが必要になります。

次の例では、アイテムを移動できなかった理由として、既に移動された、既に削除された、または、アクセスが拒否されたことを挙げています。プログラム側で容易に原因を究明できるのであれば、ユーザーに具体的な原因を調査させる必要はありません。

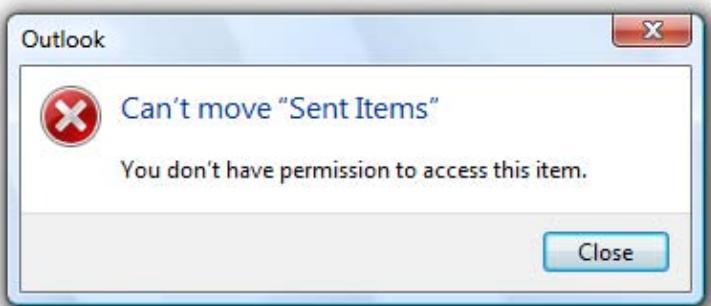
間違った例:



この情報だけでは、問題の原因がわからず、ユーザーがトラブルシューティングを行う必要が生じます。

アクセスが拒否された場合はプログラム側で判断できるので、この問題は具体的なエラー メッセージで報告する必要があります。

正しい例:



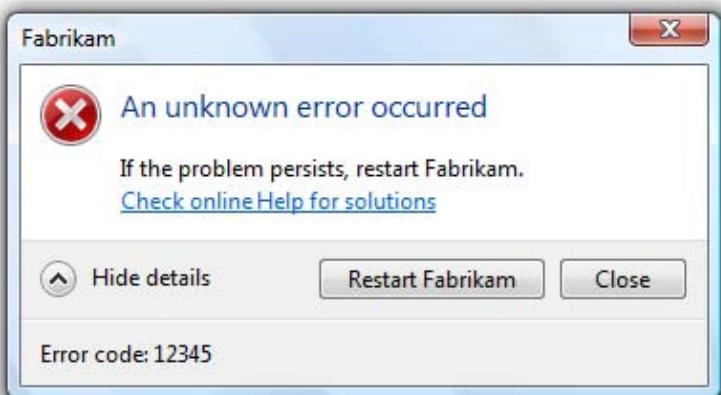
具体的な原因が示されていれば、トラブルシューティングを行う必要はありません。

1つのメッセージに複数の原因を記述するのは、具体的な原因を絞り込めない場合に限ります。この例では、アイテムが移動されたのか、削除されたのかを、プログラム側で判断することは困難である可能性があるため、1つのエラー メッセージに複数の原因を記述しても差し支えはないかもしれません。しかし、この例で、削除済みのファイルを移動できなかったとしても、それをユーザーが気にする可能性は低いでしょう。原因がこのようなことであれば、エラー メッセージを表示する必要はありません。

#### 不明なエラーの処理

何が問題で、何が原因なのか、また、どうすれば解決できるのか、本当に見当がつかない場合もあります。エラーを表示しないことが適切でない場合は、根拠のない問題、原因、解決策を表示するよりも、率直に、情報が不足していることを伝えた方がよいでしょう。

たとえば、プログラムにハンドルされない例外が存在する場合は、次のようなエラー メッセージが適しています。



不明なエラーを回避できない場合は、情報が不足していることを率直に伝えるようにします。

これに対し、役に立つ可能性が十分にあるならば、具体的で実行性が高い情報を積極的に提供するよう

にします。



不明なエラーでも、ネットワーク接続に問題がある可能性が高い場合は、このようなエラー メッセージが適しています。

#### 適切なメッセージの種類を判断する

問題によっては、強調するポイントや表現の仕方により、エラー、警告、または情報のいずれでも表現できる場合があります。たとえば、Windows Internet Explorer® の設定で、署名されていない ActiveX コントロールの読み込みが禁止されているために、Web ページが ActiveX コントロールを読み込めないという状況を考えてみます。

- エラー。"このページに署名なしの ActiveX コントロールを読み込むことができません。" (既存の問題として表現されている)
- 警告。"Windows Internet Explorer が、署名なしの ActiveX コントロールを読み込むように設定されていないため、このページは予期したとおりに動作しない可能性があります。" または "このページで署名なしの ActiveX コントロールをインストールできるようにしますか? 信頼できない発行元からのコントロールをインストールすると、コンピューターに危害を与える可能性があります。" (どちらも、将来問題を引き起こす可能性のある状況として表現されている)
- 情報。"Windows Internet Explorer は、署名なしの ActiveX コントロールをブロックするように設定されています。" (単なる事実として表現されている)

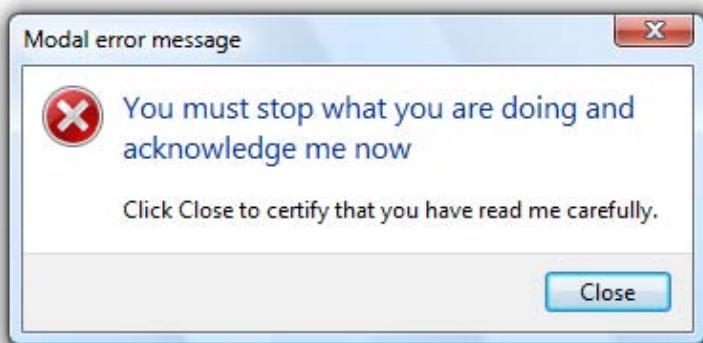
適切なメッセージの種類を決定するには、ユーザーが認識または対処する必要のある、問題の最も重要な側面に焦点を当てます。一般に、何かの問題によってユーザーが操作を続けられなくなっている場合は、その問題をエラーとして表示します。ユーザーが操作を続行できる場合は、警告として表示するようになります。こうした重要な要素に基づいて、[メイン指示テキスト](#)やその他の関連するテキストを作成したうえで、それに合ったアイコン ([標準アイコン](#)またはその他) を選択します。メイン指示テキストとアイコンは、常に調和している必要があります。

#### エラー メッセージの提示

このトピックで取り上げている例を見てもわかるように、Windows プログラムのエラー メッセージは、多くの場合、モーダル ダイアログ ボックスが使用されます。しかし、エラー メッセージの提示方法は、それだけではありません。たとえば、次のような方法があります。

- インプレース
- バルーン
- 通知
- 通知領域アイコン
- ステータスバー
- ログ ファイル (IT プロフェッショナル向けのエラーの場合)

モーダル ダイアログ ボックスでのエラー メッセージの表示は、その場でユーザーの注意を引き、確認を求めるという意味では有効な手段です。しかし、その注意が必ずしも必要でない場合は、逆に大きな欠点となります。



ユーザーの操作を中断してまで、[閉じる] ボタンをクリックしてもらう必要性は本当にあるか、考えてみましょう。必要なければ、モーダルダイアログ ボックスは使わずに他の方法を検討します。

モーダルダイアログ ボックスは、直ちにユーザーに確認を求める必要がある場合は有効ですが、そうでない場合は逆効果になることがあります。一般に、目的を果たすことができる範囲で最も控えめな表示方法を使用することをお勧めします。

#### 情報量を抑える

一般に、ユーザーはざっと見るだけで、きちんと読むことはしません。情報量が多いほどテキストを流し読みするのが難しくなり、ユーザーに全部読んでもらえる可能性は低くなります。テキストの情報量は必要最小限に減らし、追加情報を提供する場合は、段階的表示やヘルプ リンクを使用するようにします。

極端な例も多数ありますが、よく見られる典型的な例を 1 つ挙げます。次の例では、良いエラー メッセージの特徴を多く備えている反面、簡潔さに欠け、なかなか読む気になられません。

間違った例:



完成度の高いエラー メッセージではありますが、情報量が多すぎます。

このテキストが伝えようとしていることは、次のようなことではないでしょうか。

正しい例:



エラー メッセージの内容は基本的に同じですが、先ほどの例よりも、はるかにすっきりしています。

このエラー メッセージの特徴は、詳細な情報をヘルプに委ねるという、逆ピラミッド型の提示スタイルを使用していることです。

情報過多に関するガイドラインと例については、「[ユーザー インターフェイスのテキスト](#)」を参照してください。

#### 8つの重要な点

- 1.. エラー処理を考慮したプログラム設計を行う。
2. 不要なエラー メッセージは表示しない。
3. 必要なエラー メッセージを表示して、ユーザーの混乱を防ぐ。
4. エラー メッセージには、問題、原因、および解決策を必ず明記する。
5. 適切なエラー メッセージの条件(関連性が高い、実行性が高い、簡潔、明確、具体的、丁寧、めったに表示されない)を満たす。
6. プログラムの視点ではなく、ユーザーの視点からエラー メッセージを設計する。
7. ユーザーにトラブルシューティングをさせない(検出可能な原因ごとに異なるエラー メッセージを用いる)。
8. 目的を果たすことができる範囲で最も控えめな表示方法を使用する。

### 使用パターン

エラー メッセージには、いくつかの使用パターンがあります。

システムの問題 多くのシステムの問題は、ユーザーが解決できます。

オペレーティングシステム、ハードウェア デバイス、ネットワーク、プログラムでエラーが発生したり、タスクの実行に必要な状態になつていないなど。



この例では、プログラムがカメラを検出できず、ユーザー タスクを実行できません。



この例では、タスクを実行するために必要な機能をオンにする必要があります。

ファイルの問題  
ユーザーによつ  
て開始されたタ  
スクに必要な  
ファイルまたは  
フォルダーが見  
つからない、既  
に使用されてい  
る、適切な形式  
になっていない  
など。

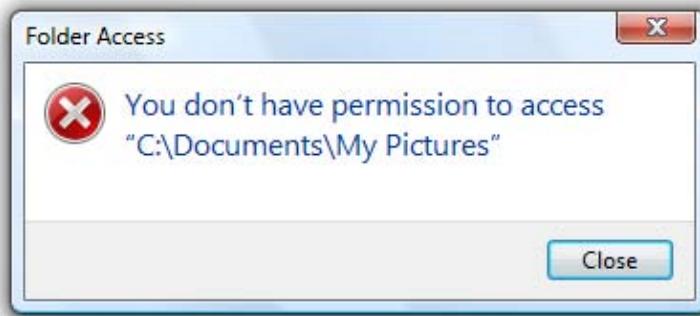


この例では、ファイルまたはフォルダーが見つからず、削除できません。



この例では、指定されたファイル形式が、プログラムによってサポートされていません。

セキュリティの  
問題  
ユーザーにリ  
ソースへのア  
クセス許可が  
ないか、ユーザー  
によつて開始され  
たタスクを実行  
するための十分  
な権限がない。



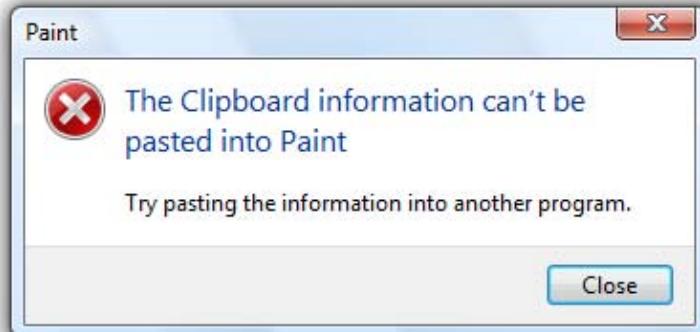
この例では、ユーザーにリソースへのアクセス許可がありません。



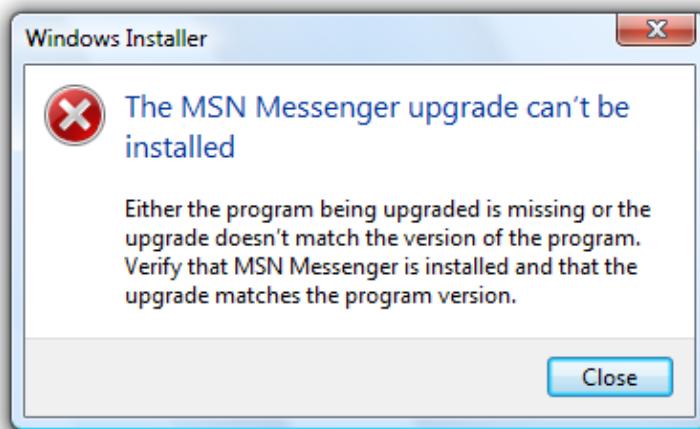
この例では、タスクを実行するための権限がユーザーにありません。

タスクの問題  
ユーザーによつ  
て開始されたタ  
スクを実行する  
うえで特定の問  
題が存在する

(システムの問題、ファイルが見つからない、ファイル形式の問題、セキュリティの問題を除く)。



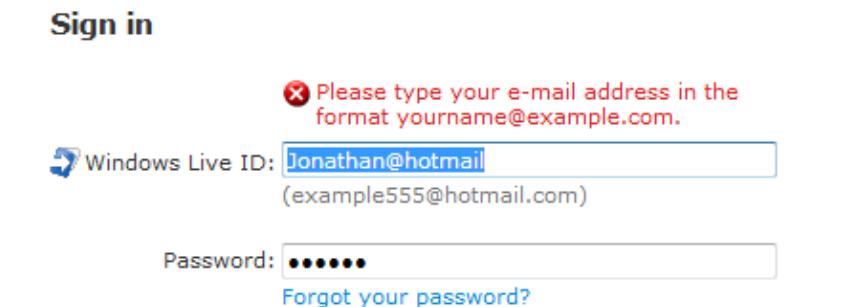
この例では、クリップボードのデータをペイントに貼り付けることができません。



この例では、ユーザーがソフトウェアのアップグレードをインストールできません。

ユーザー入力の問題  
ユーザーが間違った値を入力した、または他のユーザー入力と矛盾する値を入力した。

この例では、ユーザーが入力した時間値に誤りがあります。



この例では、ユーザー入力が正しい形式になっていません。

## ガイドライン

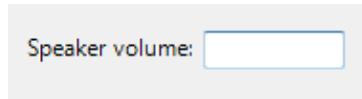
### 提示方法

- 適切な場合は常にタスク ダイアログ ボックスを使用します。タスク ダイアログ ボックスを使用することによって、一貫した外観とレイアウトを実現できます。タスク ダイアログ ボックスを使用するには、Windows Vista® 以降が必要であるため、以前のバージョンの Windows には適しません。メッセージ ボックスを使用する必要がある場合は、メイン指示テキストと補足指示テキストの間に改行を 2 つ入力して分離します。

### ユーザー入力エラー

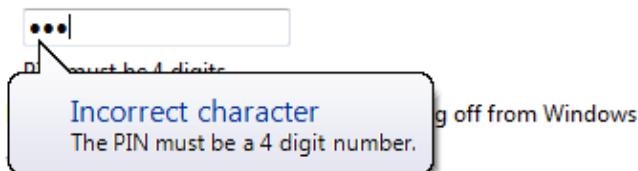
- 可能な限り、次の方法によりユーザー入力エラーを回避または低減します。
  - 有効な値しか入力できないコントロールを使用します。
  - クリックすることによってエラーになるような場合は、コントロールやメニュー項目を無効化します。コントロールやメニュー項目が無効化されても、その理由が明らかに伝わるのであれば、これも 1 つの方法です。
  - 適切な既定値を設定する。

間違った例:



この例では、入力に制限があるにもかかわらず、制約のないテキスト ボックスが使用されています。  
このような場合は、スライダーを使用してください。

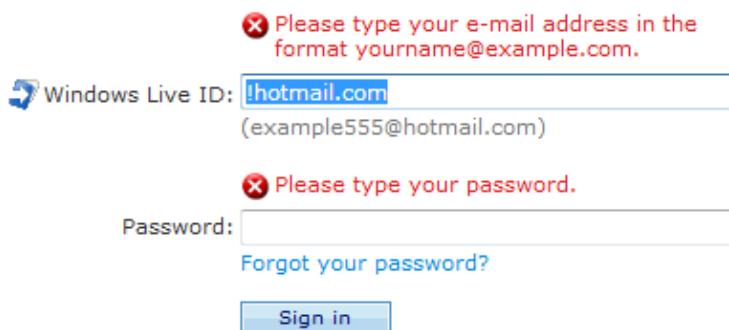
- ユーザー入力の問題をその場で指摘した方がよい場合は、モードレスのエラー処理(インプレース エラーまたはバルーン)を使用します。
- テキスト ボックスにフォーカスがあるとき、または、テキスト ボックスからフォーカスが移動した直後に、ユーザー入力の問題が検出された場合、問題がそれほど重大ではなく一過性のものであれば、バルーンを使用します。インプレース メッセージの表示に必要な空き画面領域または動的レイアウトは、バルーンには必要ありません。一度に 1 つのバルーンだけを表示します。問題がそれほど重大なものではないので、エラー アイコンは不要です。バルーンは、クリックされるか、問題が解決されるか、一定の時間が経過すると消えます。



この例では、コントロールにフォーカスがある状態で、入力に問題があることをバルーンを使って通知しています。

- 操作から時間が経過した後でエラーが検出される場合は、[インプレース エラー](#)を使用します。コミット ボタンをクリックした時点で見つかるエラーなどに使用します(即座に反映される設定にはインプレース エラーを使用しないでください)。一度に複数のインプレース エラーを表示できます。標準のテキストおよび 16 × 16 ピクセルのエラー アイコンを使用します。ユーザーが操作を確定し、他のエラーがないと判断されるまで、インプレース エラーは消えません。

## Sign in



この例では、コミット ボタンをクリックして初めてエラーが検出されるので、インプレース エラーが使用されています。

- 上記以外の問題には、モーダルなエラー処理([タスク ダイアログ ボックス](#)または[メッセージ ボックス](#))を使用します。複数のコントロールが関連するエラー、その場で指摘する必要がないエラー、コミット ボタンがクリックされた時点で検出される入力以外のエラーなどが、これに該当します。
- ユーザー入力の問題が報告された場合は、間違ったデータが入力されている最初のコントロールに入力フォーカスを設定します。必要に応じてコントロールをスクロールして見える状態にしてください。テキスト ボックスの場合は、その内容全体を選択状態にします。常にどのコントロールに対するエラー メッセージなのかが、明確に伝わるようにする必要があります。

間違って入力されているデータをクリアしないでください。入力内容をそのまま残すことで、ユーザーは間違いに気付き、問題を訂正することができます。文字を最初から入力し直す手間も省くことができます。

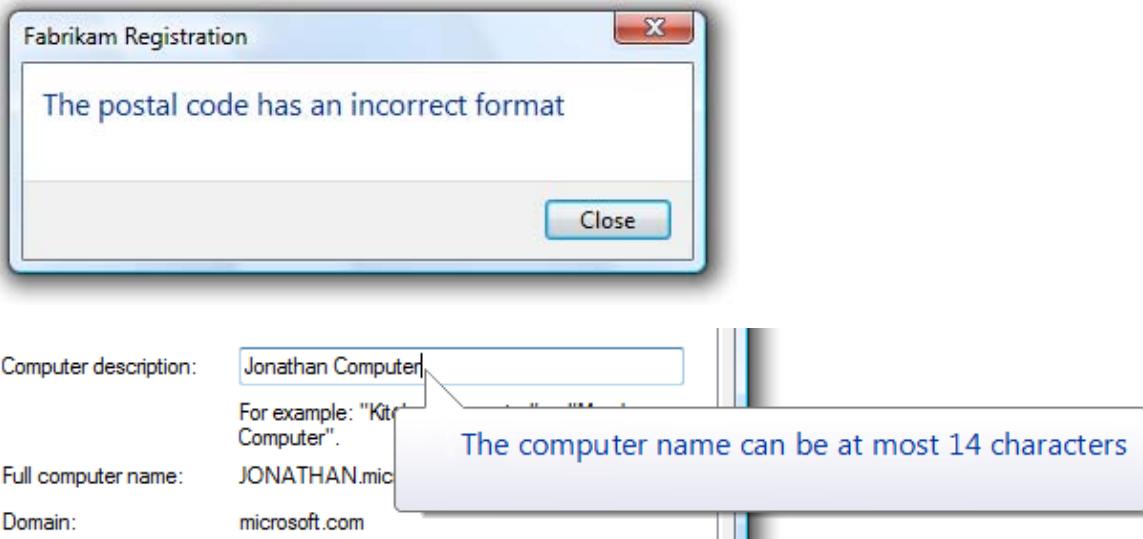
- 例外: パスワードや PIN のテキスト ボックスは例外です。入力内容がマスク処理され、そのままでは効率よく入力できないので、入力内容をクリアしてください。

## トラブルシューティング

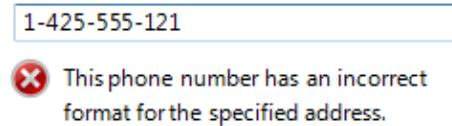
- トラブルシューティングの必要性が生じないように注意してください。検出可能な複数の原因をそのまま 1 つのエラー メッセージで報告することは避けます。
- 検出可能な個々の原因に対応する個別のエラー メッセージを使用します(通常は、それぞれに補足指示テキストを付けます)。たとえば、いくつかの理由でファイルを開くことができない場合は、理由ごとに補足指示テキストを提供します。
- 1 つのメッセージに複数の原因を記述するのは、具体的な原因を絞り込めない場合に限ります。この場合、問題の解決につながる可能性の高い解決策から順に列挙するようにします。そうすることで、ユーザーが効率よく問題を解決することができます。

## アイコン

- モーダル エラー メッセージ ダイアログ ボックスにはタイトルバー アイコンを割り当てません。タイトルバーアイコンは、メイン ウィンドウとサブ ウィンドウを視覚的に区別する場合に使用します。
- エラー アイコンを使用します。次の場合は例外です。
  - ユーザー入力の問題をモーダル ダイアログ ボックスまたはバルーンで表示する場合は、アイコンを使用しません。このような場合にアイコンを使用するのは、推奨される Windows のトーンに反するものです。ただし、インプレース エラー メッセージの場合は、エラー メッセージであることが明らかにわかるように、小さなエラー アイコン(16 × 16 ピクセル)を使用します。

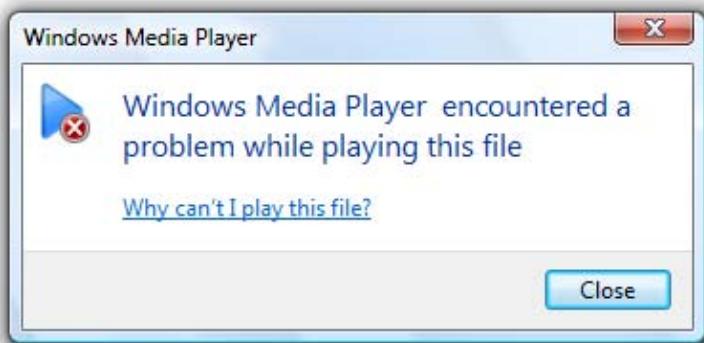


これらの例は、エラー アイコンが不要なユーザー入力の問題です。



この例は、インプレース エラー メッセージなので、エラー メッセージであることを明確にするため、小さなエラー アイコンが必要です。

- アイコンが割り当てられている機能でユーザー入力以外の問題が生じている場合は、その機能のアイコンにエラー アイコンを重ねて表示します。その場合は、機能の名前をエラーの表題として使用します。



この例では、該当する機能のアイコンにエラー アイコンを重ね、エラーの表題に、その機能の名称を表示しています。

- エラーに警告アイコンを使用しないでください。ユーザーの不安を和らげる目的でこのような手法が使用されていますが、エラーと警告は異なります。

間違った例:

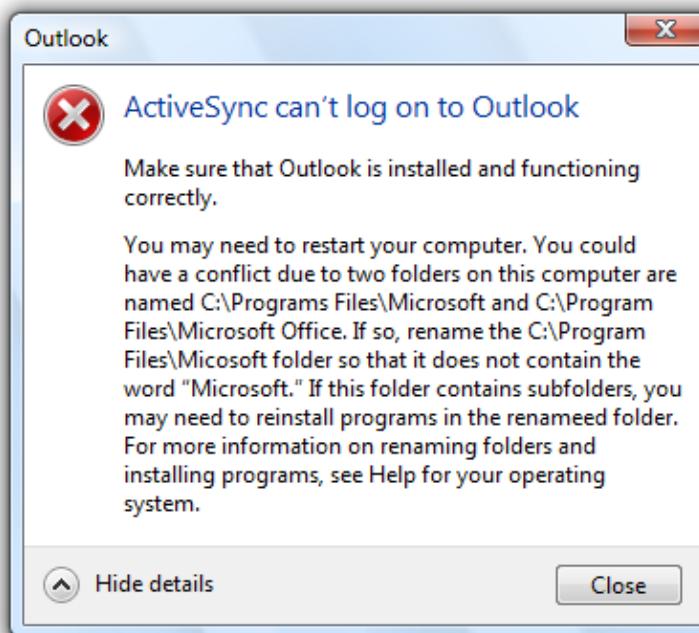


この例では、エラーの深刻さを和らげるという、本来とは異なる目的で警告アイコンが使用されています。

その他のガイドラインと例については、「[標準アイコン](#)」を参照してください。

#### 段階的表示

- 詳細の表示/非表示(段階的表示)ボタンを使用して、エラー メッセージに含まれる高度な情報や詳細な情報を非表示にします。標準的な用途では、このようにしてエラー メッセージを簡素化します。必要な情報は非表示にしないでください。非表示にすると、ユーザーが気づかない可能性があります。



この例は、段階的表示 ボタンを使用して、ユーザーが必要に応じて詳細な情報を閲覧できるようになっています。不要であれば展開しなくて済むため、UI もすっきりさせることができます。

- 実際に詳しい情報がない場合は、詳細の表示/非表示ボタンを使用しません。同じ情報を別の言葉で言い換えただけのものにならないように注意してください。
- ヘルプ情報を表示する目的で詳細の表示/非表示ボタンを使用しないでください。そのような場合は、ヘルプリンクを使用してください。

ラベルのガイドラインについては、「[段階的表示コントロール](#)」を参照してください。

今後、このメッセージを表示しない

- エラー メッセージに、このオプションが必要な場合は、エラーとその頻度をもう一度よく考えてください。完成度の高いエラー(関連性が高い、実行性が高い、めったに表示されない)の特徴をすべて備えているのであれば、ユーザーにその表示を制限させる理由はありません。

その他のガイドラインについては、「[ダイアログ ボックス](#)」を参照してください。

既定値

- 既定値には、最も安全で、最も障害の起こる可能性が低く、最もセキュリティの高い応答が得られる値を選択します。安全性を判断要素として考える必要がない場合は、最もよく使用されるコマンドまたは最も便利なコマンドを選択します。

ヘルプ

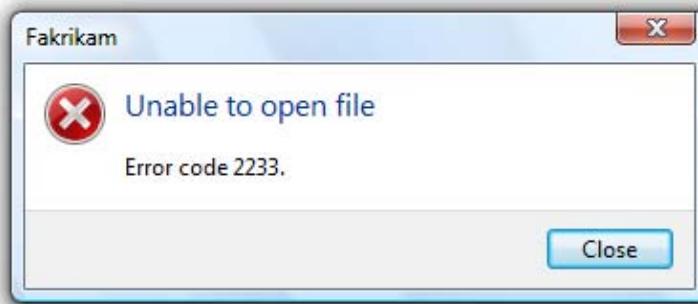
- エラー メッセージはヘルプを参照しなくても済むように設計します。解決策にいくつもの手順が伴う場合を除けば、通常、問題を理解し解決するためにユーザーが外部のテキストを読む必要はありません。
- ヘルプのコンテンツは、関連性の高い有益な内容となるように配慮します。エラー メッセージと同じ情報を別の言葉で置き換えただけのものにならないように注意してください。問題を未然に防止する方法など、エラー メッセージには盛り込まれていない有益な情報を含めるようにします。リンクできる情報があるという理由だけで、ヘルプへのリンクを設けることは避けてください。
- ヘルプ コンテンツへのアクセスには、具体的で、簡潔で、関連性の高いヘルプ リンクを使用します。この用途でコマンド ボタンや段階的表示を使用しないでください。
- 具体的で実行性が高いエラー メッセージを作成できない場合は、オンライン ヘルプ コンテンツへのリンクの提供を検討します。オンライン ヘルプという形で追加情報をユーザーに提供すれば、プログラムのリリース後に情報を更新することもできます。

その他のガイドラインについては、「[ヘルプ](#)」を参照してください。

エラー コード

- 具体的で実行性が高いエラー メッセージを作成することが難しく、また、ヘルプを利用する利点もない場合は、エラー コードを提供することも検討します。ユーザーがインターネットから追加情報を検索する場合に、エラー コードがよく利用されます。
- 常に問題や解決策の説明文を提供します。問題や解決策の説明として、エラー コードだけに頼るのは避けてください。

間違った例:



この例では、エラー コードが解決策の説明文として代用されています。

- 原因にはそれぞれ一意のエラー コードを割り当てます。これにより、ユーザーがトラブルシューティングを行う

必要がなくなります。

- インターネットで簡単に検索できるエラー コードを使用します。32 ビットのコードを使用する場合は、先頭に 16 進数であることを表す "0x" を付加し、大文字を使用します。

正しい例:

1234

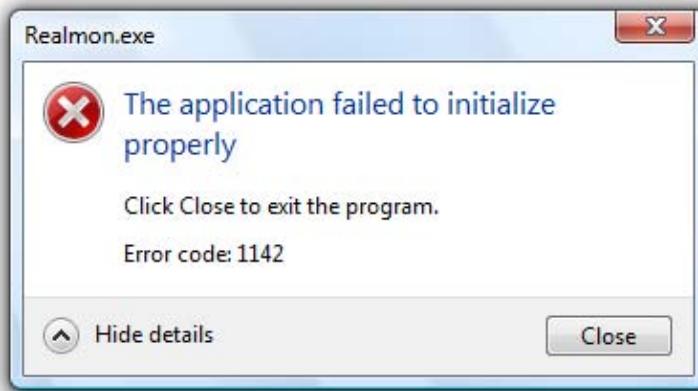
0xC0001234

間違った例:

-1

-67113524

- 詳細の表示/非表示ボタンを使用してエラー コードを表示します。「エラー コード:<エラー コード>」のように表記します。



この例では、エラーメッセージを補足するものとしてエラーコードが使用され、追加情報を得る手立てとなっています。

## サウンド

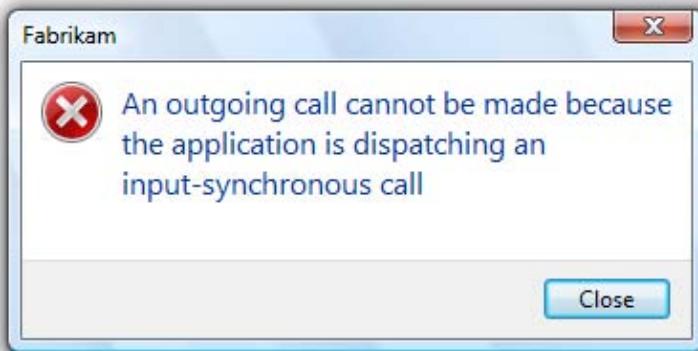
- エラーメッセージに効果音やビープ音は使用しないでください。耳障りになることが多く、必要性もありません。
  - 例外: システム エラーの場合は例外です。コンピューターの運用に重大な影響を及ぼす問題で、ユーザーがすぐに対処しなければ深刻な結果が予想される場合は、効果音を再生するようにします。

## テキスト

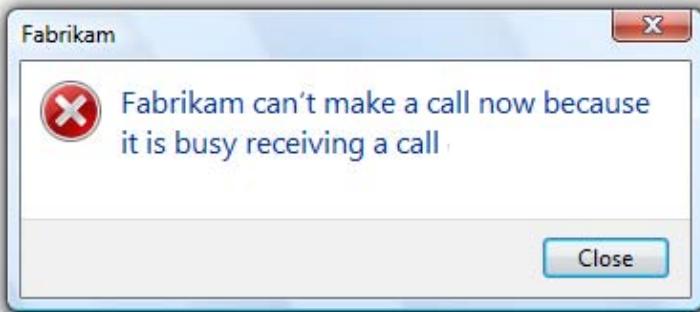
### 全般

- 冗長なテキストは削除します。タイトル、メイン指示テキスト、補足指示テキスト、コマンド リンク、コミット ボタンなどに冗長なテキストがないか確認してください。通常、メイン指示テキストと対話型コントロールのテキストはそのまま残し、他の部分にある冗長なテキストを削除します。
- ユーザーの立場に立った説明を使用します。ソフトウェアの観点からではなく、ユーザーの操作や目的という観点から問題を説明するようにします。対象ユーザーが理解できる有用な言葉を使用します。難解な技術的表現は避けます。

間違った例:



正しい例:

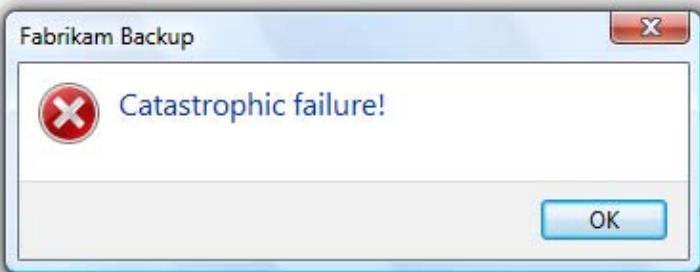


正しい例では、ユーザーが無理なく理解できる言葉が用いられているのに対し、間違った例では、難解な技術的表現が用いられています。

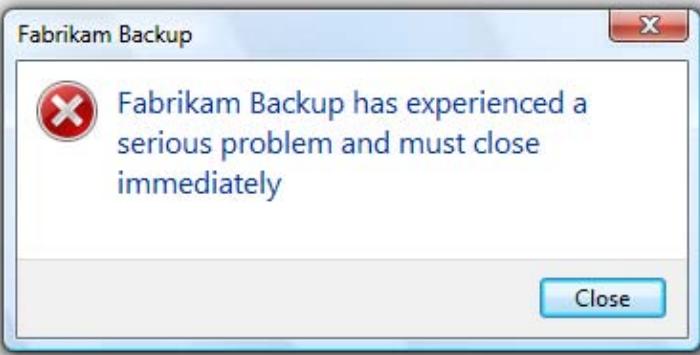
- 次の言葉は使用しないでください。
  - エラー、障害 (代わりに "問題" を使用)
  - 失敗しました (代わりに "できません" を使用)
  - 不正な、無効な、悪い (代わりに "正しくない" を使用)
  - 致命的な (代わりに "プログラムの終了" を使用)
  - 中止、キル、強制終了 (代わりに "停止" を使用)
  - 壊滅的な (代わりに "重大な" を使用)

以上に挙げた言葉は不要であり、推奨される Windows のトーンに反するものです。エラー アイコンを正しく使用すれば、問題の発生を効果的に伝えることができます。

間違った例:



正しい例:



間違った例で使用されている、"壊滅的な" および "障害" という表現は必要ありません。

- ユーザーに責任を押し付けたり、ユーザーに非があるような言い回しを避けます。"あなた" という表現は使わないようにします。一般には能動態が好まれますが、ユーザーを主語にすることでエラーの責任がユーザーにあるかのように感じさせてしまう可能性がある場合は受動態を使用します。

間違った例:



正しい例:



間違った例では、能動態が用いられているため、ユーザーに非があるかのように感じられます。

- 具体的にします。"構文エラー" や "無効な操作" など、あいまいな表現を避けます。関連するオブジェクトの名称、場所、値などを具体的に提示します。

間違った例:

ファイルが見つかりません。  
ディスクがいっぱいです。  
値が範囲外です。  
文字が無効です。  
デバイスを使用できません。

これらの問題は、具体的な名前、場所、値が記載されていれば、より簡単に解決できます。

- 具体的に記述した方がよいかといつて、可能性の低い問題、原因、解決策を挙げないでください。適切でない可能性のある問題、原因、解決策を提供することは避けてください。たとえば、不確実な情報を挙げるよりも、"不明なエラーが発生しました" とする方が適切です。
- "してください (please)" という表現は避けます。ユーザーに何か面倒なこと (処理の終了を待つなど) を依頼する場合や、ソフトウェア側の都合で何かを依頼する状況を除き、"してください" という表現は避けます。

正しい例:

"ファイルをコンピューターにコピーしています。しばらくお待ちください..."

- 謝罪表現は、ユーザーにとって深刻な問題を引き起こしたエラー メッセージでのみ使用します。たとえば、データが失われた、コンピューターを使用できないなどが該当します。プログラムの正常な動作の範囲内で発生した問題 (ネットワーク接続の検出に一定時間かかるなど) の場合は、謝罪表現は使用しないでください。

正しい例:

"申し訳ございません。回復不可能な問題が検出されたため、コンピューター上のファイルを保護するために Fabrikam バックアップを終了しました。"

- 製品に言及する際は省略名を使用します。製品のフルネームや商標記号は使用しません。ユーザーが製品から会社名を連想できない場合を除いて、会社名は含めません。プログラムのバージョン番号も省略します。

間違った例:



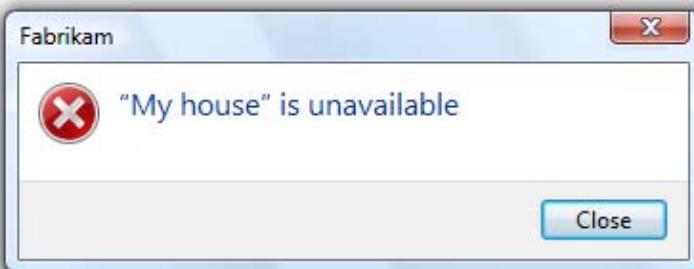
正しい例:



間違った例では、製品のフルネームと商標記号が使用されています。

- オブジェクト名は二重引用符で囲みます。テキストが読みやすくなり、意味の取り違えを未然に防ぐことができます。
  - 例外: 完全修飾のファイルパス、URL、ドメイン名などを引用符で囲む必要はありません。

正しい例:



この例では、オブジェクト名が引用符で囲まれていないと、エラー メッセージの意味が正しく伝わりません。

- オブジェクト名を文の冒頭に置くことは避けます。内容がわかりにくくなる場合があります。
- 感嘆符やすべて大文字での表現は使用しないでください。感嘆符や大文字は、ユーザーに向かって叫んでいる印象を与えます。

その他のガイドラインと例については、「[スタイルとトーン](#)」を参照してください。

## タイトル

- タイトルにエラーの発生源となったコマンドや機能を明記します。次の場合は例外です。
  - いくつもの異なるコマンドによってエラーが表示される場合は、プログラム名を明記するようにします。
  - タイトルが冗長であったりメイン指示テキストと紛らわしい場合は、代わりにプログラム名を使用します。
- タイトルで問題を説明したり要約することはしません。問題の説明や要約はメイン指示テキストで行います。

間違った例:



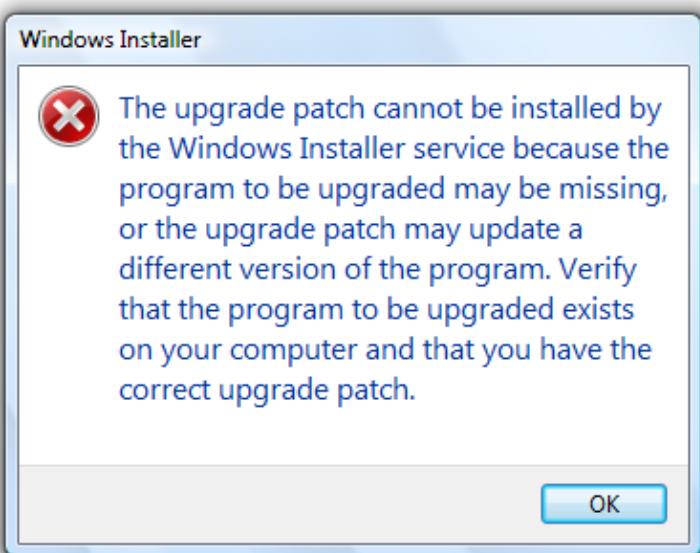
この例では、問題を説明するためにタイトルが間違って使用されています。

- タイトルスタイルの大文字化を使用し、末尾に句読点は付けません。

#### メイン指示テキスト

- メイン指示テキストでは、問題をはっきりと、わかりやすく、具体的な言葉で説明します。
- 簡潔な 1 文だけを使用します。メイン指示テキストには必須の情報だけを含めます。主語がプログラムまたはユーザーの場合は、暗黙の了解として省略してかまいません。簡潔にまとめることができる場合は、問題の理由を記述します。さらに説明が必要な場合は、補足指示テキストを使用します。

間違った例:



この例では、エラー メッセージ全体がメイン指示テキストに記述されているため、非常に読みにくいものになっています。

- 具体的に記述します。関連するオブジェクトがある場合は、その名前を提示します。
- 完全なファイルパスや URL をメイン指示テキストに記述することは避けます。代わりに略称(ファイル名など)を使用し、フルネーム(ファイルパスなど)は補足指示テキストに記述するようにします。ただし、他の補足指示テキストが必要なく、記述する完全なファイルパスや URL が 1 つのみであれば、メイン指示テキストに記述してもかまいません。



これはファイル名だけをメイン指示テキストに記述した例です。完全なパスは補足指示テキストに記述されています。

- コンテキストから明らかな場合は完全なファイルパスや URL を表示しません。



この例では、ユーザーが Windows エクスプローラーでファイルの名前を変更しています。この場合、完全なファイルパスはコンテキストから明らかなので必要ありません。

- 可能な限り現在形を使用する。
- センテンススタイルの大文字化を使用します。
- 指示テキストが文の場合、文末の句点は含めません。指示テキストが疑問文の場合は、疑問符を付けます。

#### メイン指示テキストのテンプレート

表現に関して厳密な規則はありませんが、メイン指示テキストには、可能な限り次のテンプレートを使用してください。

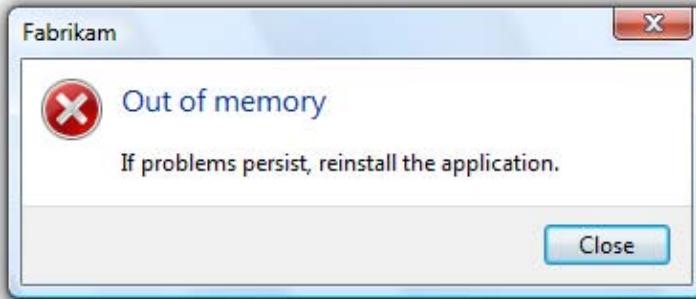
- <主語 (省略可)> は <動作を実行> できません
- <理由> のため、<主語 (省略可)> は <動作を実行> できません
- <主語 (省略可)> は "<オブジェクト名>" に対して <動作を実行> できません
- <理由> のため、<主語 (省略可)> は "<オブジェクト名>" に対して <動作を実行> できません
- <動作を実行> するのに十分な <リソース> がありません
- <主語> には、<目的> に必要な <オブジェクト名> がありません
- <デバイスまたは設定> がオフになっているため、<予期しない結果> になります
- <デバイスまたは設定> が <利用できません | 見つかりません | オンになっていません | 有効になっていません>
- "<オブジェクト名>" は現在利用できません
- ユーザー名またはパスワードが正しくありません
- "<オブジェクト名>" へのアクセス許可がありません
- <動作を実行> する権限がありません
- <プログラム名> で重大な問題が発生したため、直ちに終了する必要があります

文法的に正しく、ガイドラインに従う限り、必要に応じてメイン指示テキストのテンプレートを変更してかまいません。

## 補足指示テキスト

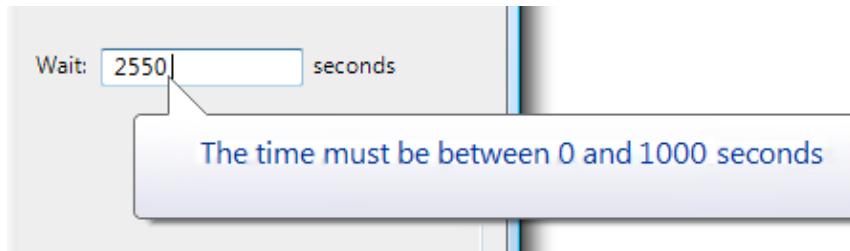
- 補足指示テキストは次の用途に使用します。
  - 問題に関する追加情報を提供する。
  - 問題の原因を説明する。
  - 問題を解決するための手順を列挙する。
  - 問題の再発を未然に防ぐための対策を提供する。
- ユーザーが問題を解決できるように、可能な限り、実用的で有益な解決策を提供します。ただし、解決の見込みが十分にある解決策を示すようにしてください。解決につながる見込みの低い、憶測だけの情報でユーザーに無駄な時間をとらせないようにします。

間違った例:



この例で問題に対して推奨されている解決策は、うまくいく可能性もありますが、決して有望な方法ではありません。

- ユーザーの入力内容に誤りがあることが問題となっている場合は、補足指示テキストを使用して、正しい値について説明します。他の資料を参照しなければ原因が究明できないということのないようになります。
- 問題を示す文から容易に解決方法がわかる場合は、解決策は提供しません。



この例では、解決策が問題を示す文から容易に推測できるため、補足指示テキストは不要です。

- 解決策に複数の手順が含まれる場合は、その順に記述します。ただし、多くても 2 つか 3 つの手順に収めないと、記憶するのが困難になります。その意味で、複数の手順から成る解決策はできるだけ避けてください。多くの手順が必要となる場合は、適切なヘルプトピックへのリンクを提供します。
- 補足指示テキストは簡潔にします。ユーザーが知る必要のある情報だけを提供します。不要な情報は省略します。適度な長さの文を最大 3 つで収めることを目標にします。
- 文の流れに従って操作した結果、予期せぬ事態に陥るということのないよう、操作よりも結果を先に記述します。

正しい例:

Windows を再起動するには、[OK] をクリックします。

間違った例:

[OK] をクリックして Windows を再起動します。

間違った例では、ユーザーが結果を認識する前に [OK] をクリックしてしまう可能性があります。

- 他に解決策がない場合を除き、管理者への問い合わせは提案しません。このような解決策は、本当に管理者以外に解決できない問題の場合にのみ採用します。

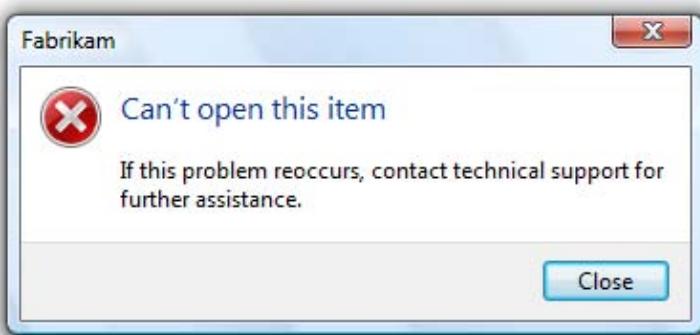
間違った例:



この例では、ユーザーのネットワーク接続に原因がある可能性が高く、管理者に問い合わせるほどの問題ではありません。

- テクニカルサポートへの問い合わせは提案しません。テクニカルサポートに問い合わせて問題を解決するという選択肢はいつでも利用できます。エラーメッセージを通じて促す必要はありません。むしろ、ユーザーがテクニカルサポートに頼らずに問題を解決できるような、有益なエラーメッセージを作成することを目指します。

間違った例:



テクニカルサポートへの問い合わせを提案する不適切なエラーメッセージの例です。

- 文を使用し、末尾に句点を付けます。センテンススタイルの大文字化を使用します。

#### コミット ボタン

- 問題を解決するためのコマンドボタンまたはコマンドリンクをエラーメッセージに設ける場合は、それぞれのガイドラインに従います(「[ダイアログ ボックス](#)」を参照)。
- それ以外の場合は、[閉じる]ボタンを使用します。エラーメッセージに[OK]ボタンは使用しないでください。問題を容認するような印象を与えてします。
  - 例外: 使用するエラー報告メカニズムで、固定のラベルとして[OK]ボタンがある場合は(ErrorMessage APIなど)、[OK]ボタンを使用してもかまいません。

#### ドキュメント

エラーに言及するときは、以下のこと留意します。

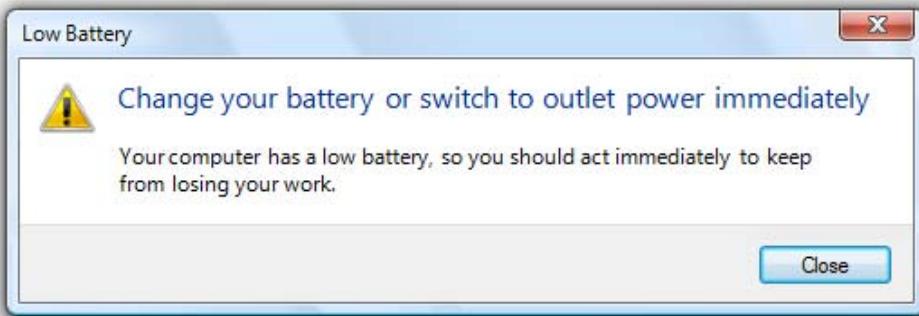
- エラーは、メイン指示テキストで示します。メイン指示テキストが長い、または詳細な場合は、内容を要約します。
- 必要に応じて、エラーメッセージダイアログボックスを"メッセージ"と呼ぶこともできます。プログラミングやその他の技術文書内でのみ、"エラーメッセージ"と呼びます。
- テキストを二重引用符(" ")で囲み、可能な場合は太字にします。

例: "ドライブに CD ディスクが挿入されていません"というメッセージが表示された場合は、新しい CD ディスクをドライブに挿入して、もう一度やり直してください。

## 警告メッセージ

適切なユーザーインターフェイスかどうかの判断基準  
デザインコンセプト  
使用パターン  
ガイドライン  
提示方法  
アイコン  
今後、このメッセージを表示しない  
段階的表示  
既定値  
テキスト  
ドキュメント

"警告メッセージ" は、将来、問題を引き起こす可能性のある状況をユーザーに通知する、モーダル ダイアログ ボックス、インプレース メッセージ、通知、またはバルーンです。



典型的なモーダル警告メッセージ。

警告は基本的に、次のいずれかのうち 1 つまたは複数のものが失われるおそれがある場合に使用します。

- 重要な財務データなどの貴重な資産
- システムへのアクセスまたは整合性
- プライバシーまたは機密情報の制御
- ユーザーの時間 (30 秒かそれ以上の長時間)

これに対して、確認は、ユーザーが実行した操作を続行するかどうかをたずねるモーダル ダイアログ ボックスです。警告の種類によっては、確認として提示する場合があります。この場合、確認に関するガイドラインも適用されます。

注: [ダイアログ ボックス](#)、[確認](#)、[エラー メッセージ](#)、[標準アイコン](#)、[通知](#)、[レイアウト](#)に関するガイドラインは、それぞれ別の項目として記載しています。

## 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- 将来、問題を引き起こす可能性のある状況をユーザーに伝えているか。該当しない場合は、警告メッセージとはいえない。
- UI に表示されているのは、既に発生したエラーまたは問題か。該当する場合は、代わりにエラー メッセージを使用します。
- ユーザーがメッセージの結果に応じて、何らかの対処をするか、自分の操作を見直す可能性はあるか。該当しない場合は、あえてユーザーの操作を中断する必要はなく、警告を表示しない方がよいと判断できます。
- その状況はユーザーが実行した操作の直接的な結果によるものか。該当しない場合は、代わりに重大ではないイベント通知を使用することを検討します。
- 特定のコントロールに関する状況か。該当する場合は、代わりにバルーンを使用します。
- 確認の場合、ユーザーが実行しようとしている操作がリスクを伴うものかどうか。該当する場合で、その操作が重大な結果をもたらす場合や簡単に元に戻せない場合には、警告を使用する方が適切です。
- その他の種類の警告の場合、直ちに、または近い将来に実行する必要がある操作かどうか。ユーザーが直ちに問題に対処する必要がなく、操作を継続できる場合は、警告を表示しません。状況がより切迫してから、または関連性の高い状況になってから警告を表示します。

## デザイン コンセプト

### 過剰な警告を避ける

Microsoft® Windows® プログラムは、警告が過剰です。Windows プログラムは、一般的に、あらゆる場面で警告が表示され、あまり重要ではない事柄についての警告も表示されるという印象があります。プログラムによっては、ほとんどすべての質問が警告として表示されているものもあります。過剰に警告を表示すると、プログラムを使用することが危険な行為のように感じられ、本当に重要な問題から注意がそらされてしまいます。

間違った例:



過剰に警告を表示すると、プログラムが危険なものであるかのような印象を与え、法律家が作成したプログラムのように思われます。

単にデータ損失や問題が発生する可能性があるというだけでは、警告を表示するのに十分ではありません。また、望ましくない結果というものは、予期できるものでも意図的なものでもなく、簡単に修正できるものでもありません。そうでなければ、ユーザーの間違いのほとんどは、データ損失や何らかの問題を引き起こすおそれがあると解釈できるため、警告を表示する意味があります。

### 完成度の高い警告の特徴

完成度の高い警告の特徴は次のとおりです。

- リスクに関連している。完成度の高い警告は、重要な情報をユーザーに通知します。

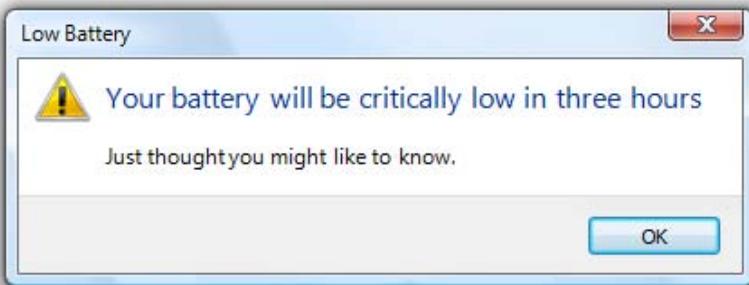
間違った例:



意図がわかりません。この確認は、ユーザーがいつも誤ってプログラムを終了すると想定していることになります。

- 即応性と関連性が高い。ユーザーが注意する必要があるだけでなく、まさに今、注意する必要があることを提示します。ユーザーは、一般的に、作業を継続できるのであれば、後で発生する可能性のある問題には関心を持ちません。

間違った例:

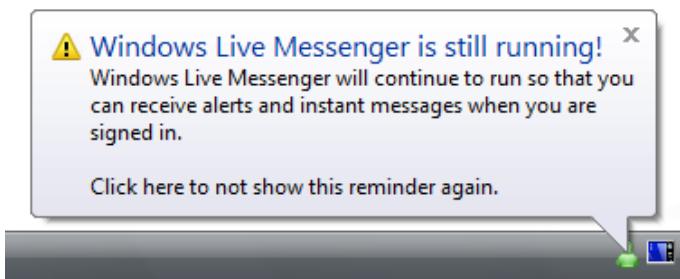


この例の警告は、3 時間後に表示した方が効果的です。

- 対応を促すものである。警告が表示された結果、ユーザーが実行したり注意しなければならない事項があります。おそらく

く、直ちに、または近い将来に実行するべき事柄です。そしておそらく、それに対するユーザーの対応はさまざまです。そこで、警告を無視した場合の結果を明確に提示する必要があります。対処方法のない警告は、ユーザーの混乱を招くだけです。

間違った例:



この通知を警告にする必要はありません。ユーザーは何を要求されているのか(さらには何に注意すればよいのか)わかりません。

- 明白なことではない。操作の結果が明白であれば、警告は使用しません。たとえば、タスクを完了しなかった場合の結果をユーザーは把握しているはずです。

間違った例:



ウィザードを完了する前にキャンセルすると、タスクは完了しません。誰でも想像できる結果です。

- 表示する頻度はできるだけ少なくする。頻繁に警告が表示されると、非効率でわずらわしく感じます。ユーザーが、問題に対処することより、警告表示を消すことに注意を払うようになります。

間違った例:



ユーザーが、根本的な問題に対処することより、警告表示を消すことに注意を払うようになる可能性が高くなります。

上記の特徴を備えていなくても、完成度の高いメッセージである可能性はありますが、少なくとも完成度の高い警告とはいえません。

#### 適切なメッセージの種類を判断する

問題によっては、強調するポイントや表現の仕方により、エラー、警告、または情報のいずれでも表現できる場合があります。たとえば、Windows Internet Explorer® の設定で、署名されていない ActiveX コントロールの読み込みが禁止されているために、Web ページが ActiveX コントロールを読み込めないという状況を考えてみます。

- エラー。"このページに署名なしの ActiveX コントロールを読み込むことができません。" (既存の問題として表現される)
- 警告。"Windows Internet Explorer が、署名なしの ActiveX コントロールを読み込むように設定されていないため、この

ページは予期したとおりに動作しない可能性があります。" または "このページで署名なしの ActiveX コントロールをインストールできるようにしますか? 信頼できない発行元からのコントロールをインストールすると、コンピューターに危害を与える可能性があります。" (どちらも、将来問題を引き起こす可能性のある状況として表現されている)

- 情報。"Windows Internet Explorer は、署名なしの ActiveX コントロールをブロックするように設定されています。" (単なる事実として表現されている)

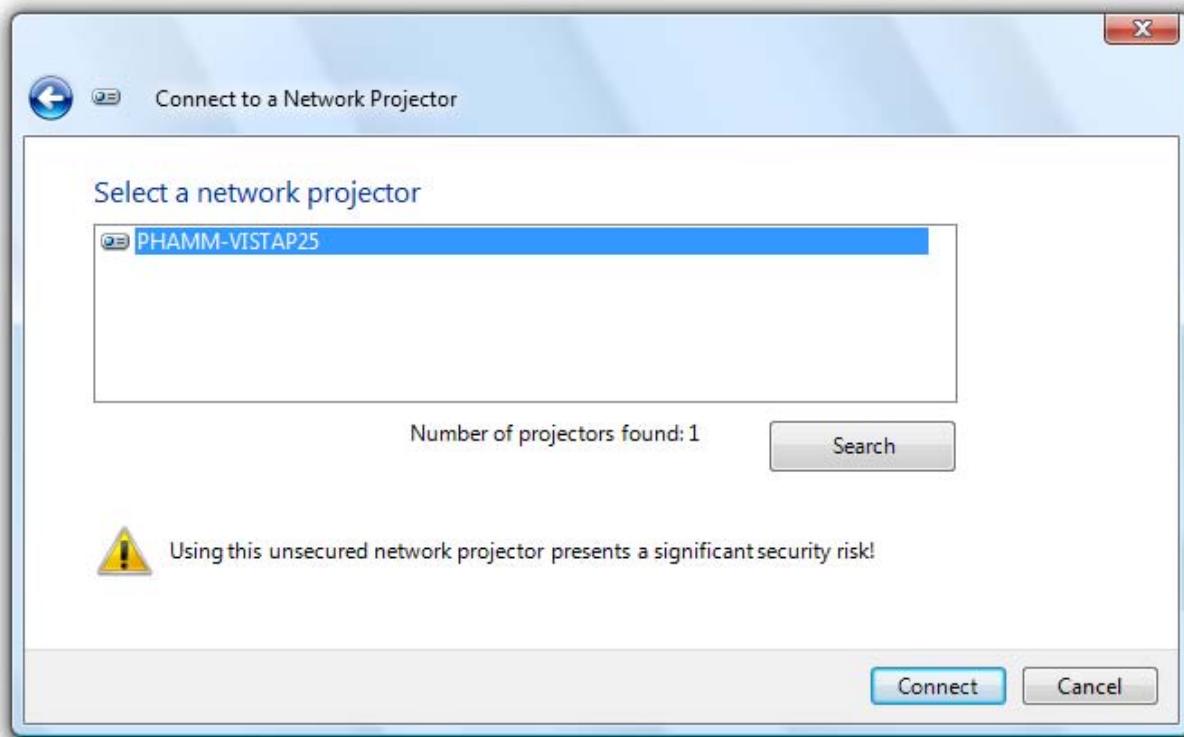
適切なメッセージの種類を決定するには、ユーザーが認識または対処する必要のある、問題の最も重要な側面に焦点を当てます。一般に、何かの問題によってユーザーが操作を続けられなくなっている場合は、その問題をエラーとして表示します。ユーザーが操作を続行できる場合は、警告として表示するようになります。こうした重要な要素に基づいて、[メイン指示テキスト](#)やその他の関連するテキストを作成したうえで、それに合ったアイコン ([標準アイコン](#)またはその他) を選択します。メイン指示テキストとアイコンは、常に調和している必要があります。

#### 具体的に記述する

次に示す情報が具体的で明確であれば、警告に対する注目度が高くなります。

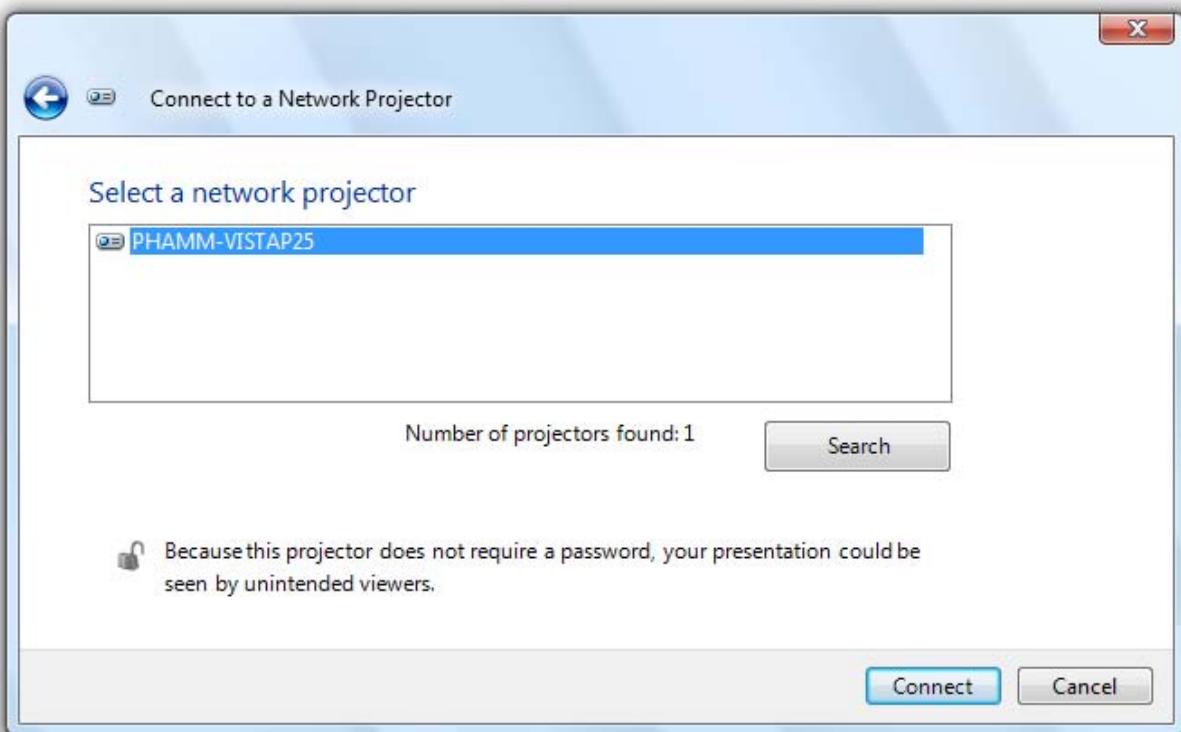
- 警告が表示された原因
- 具体的な状況と発生する可能性のある問題
- その問題に対するユーザーの対処方法
- ユーザーが何も対処を行わない場合の結果

#### 間違った例:



この例では、発生する可能性のある問題がわかりません。ネットワーク上のプロジェクターの使用を避ける以外に、ユーザーが実行できる操作もわかりません。より詳細な情報が提示されていないため、ユーザーは操作を続けることを不安に感じるので他に何もできません。

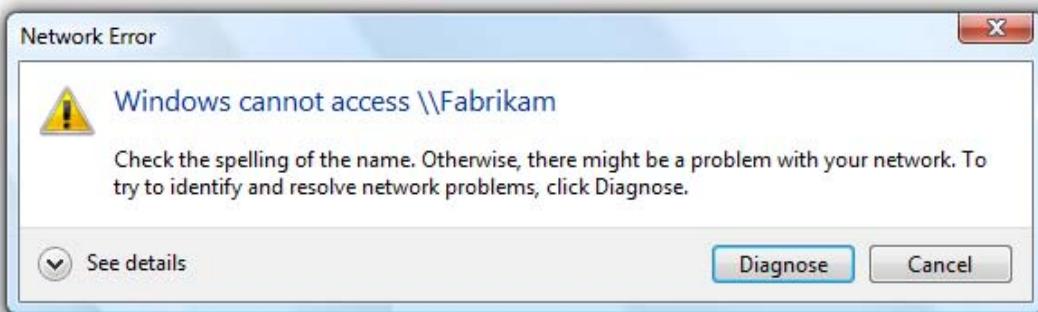
#### 正しい例:



この例では、問題とその結果が明確に示されています。

発生する可能性のある問題がユーザーに伝える必要のあるものであることは確かでも、その解決法と結果が明確でない場合があります。この場合は、あいまいな警告を表示するのではなく、最も可能性の高い情報や最も一般的な例を提示して具体的な表現にします。

正しい例:



この例では、最も可能性の高い解決法を提示して警告を具体的なものにしています。

ただし、このような場合は、他にも可能性があることを示す表現を使用するようにします。そうしないと、ユーザーに誤った情報を伝えるおそれがあります。

間違った例:



正しい例:



間違った例では、ケーブルが明らかに挿入されている場合にユーザーは戸惑うことになります。

## 2つの重要な点

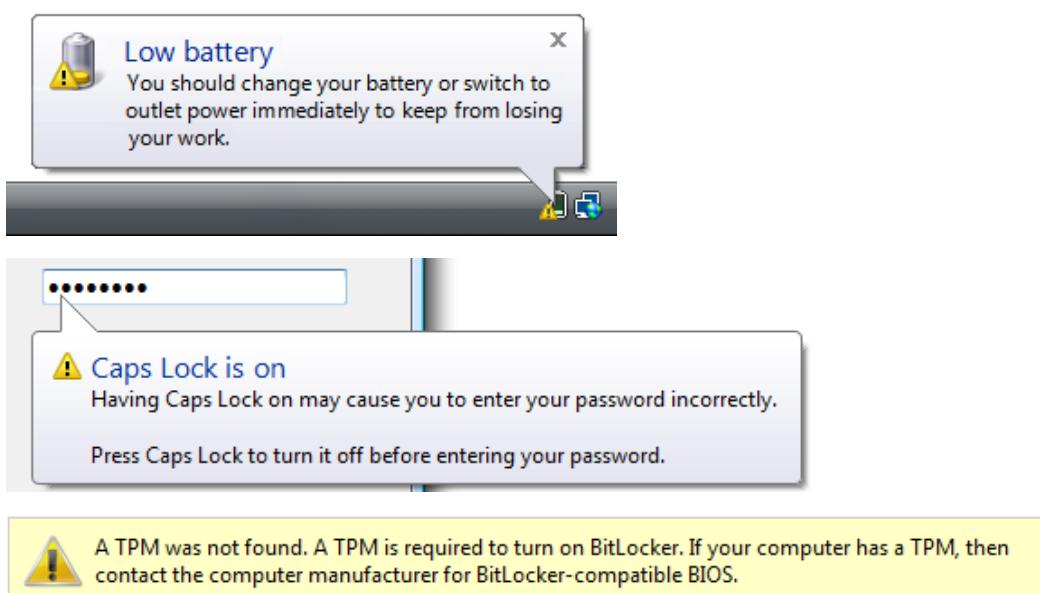
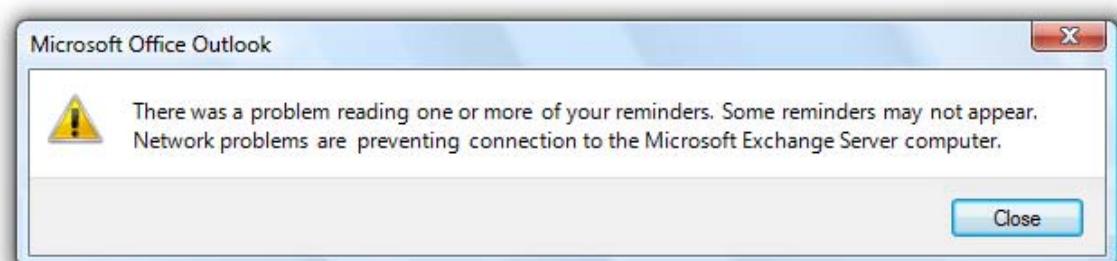
- 過剰な警告を避けます。警告を表示するのは、リスクを伴い、即応性と関連性が高く、実行性が高く、発生するのが明白ではなく、発生頻度の低い状況のみに限定します。該当しない場合は、メッセージを削除または変更します。
- 具体的で有用な情報を提供します。

## 使用パターン

警告にはいくつかの使用パターンがあります。

### 情報伝達

ユーザーに状況や発生する可能性の高い問題を知らせます。ただし、ユーザーが直ちに対応する必要がない場合もあります。



### 情報伝達警告の例。

情報伝達警告には、次の要素が含まれます。

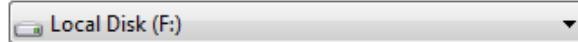
- メイン指示テキスト: 状況または発生する可能性の高い問題を示します。
- 補足指示テキスト: 発生する可能性のある問題とその重要性を説明します。
- コミットボタン: [閉じる] を使用します。

エラー防止  
特に選択を行う  
場合などに、問  
題の発生を防止  
するための情報

エラー防止警告は、インプレース警告アイコンと説明テキストを使用して提示するのが最適な方法です。

On a hard disk

をユーザーに伝えます。



Add Hardware

## Welcome to the Add Hardware Wizard

This wizard helps you install driver software to support older devices that do not support Plug-and-Play and which are not automatically recognized by Windows.

You should only use this wizard if you are an advanced user or you have been directed here by technical support.



If your hardware came with an installation CD, it is recommended that you click Cancel to close this wizard and use the manufacturer's CD to install this hardware.

To continue, click Next.

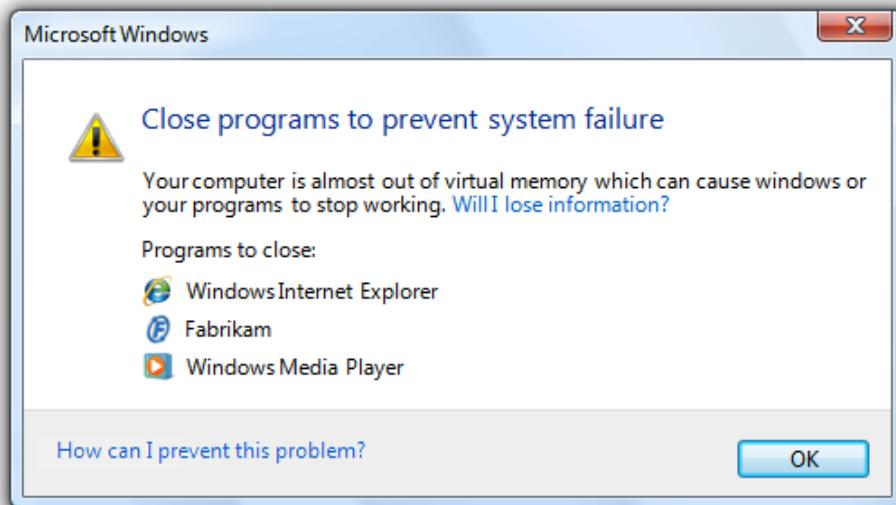
< Back

Next >

Cancel

エラー防止警告の例。

切迫している問題  
切迫している問題に対処するには、ユーザーは直ちに対応する必要があります。



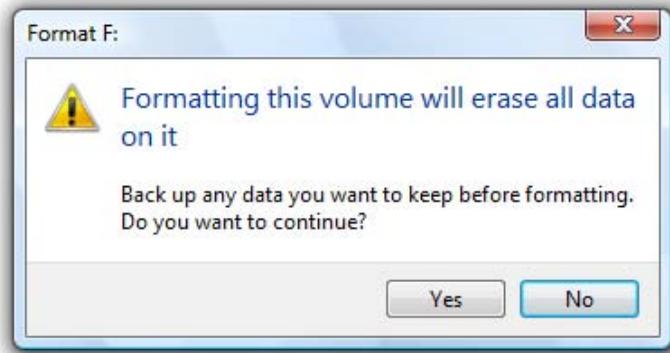
切迫している問題の警告の例。

切迫している問題の警告には、次の要素が含まれます。

- メイン指示テキスト: ユーザーが直ちに実行する必要のある操作を示します。
- 補足指示テキスト: 状況とその重要性を説明します。
- コミットボタン: 各オプションに対応するコマンドボタンまたはコマンドリンクを使用します。ダイアログボックス以外の場所で操作を実行する必要がある場合は、[OK] を使用します。

リスクを伴う操作の確認  
ユーザーが実行しようとしている

る操作にリスクが伴い、簡単に元に戻すことができない場合は、操作を継続するかどうかを確認します。



リスクを伴う操作の確認の例。

リスクを伴う操作の確認には、次の要素が含まれます。

- メイン指示テキスト: ユーザーが操作を継続するかどうかを判断するための質問を提示します。
- 補足指示テキスト: ユーザーが操作を継続しない方がよい、自明でない理由を説明します。
- コミットボタン: [はい]、[いいえ]を使用します。

この使用パターンのガイドラインについては、「[確認](#)」を参照してください。

## ガイドライン

### 提示方法

- 情報の種類に応じた UI を選択します。

ユーザーインターフェイス	情報の種類
モーダルダイアログボックス	ユーザーが直ちに対応する必要がある重大な警告(確認も含む)。
インプレース	問題の発生を防止するための情報(特にユーザーが選択を行う場合など)。
バナー	問題の発生を防止するための情報(特にタスクの完了に関係する場合など)。
通知	少なくとも一時的に無視しても安全な程度の重要性のイベントまたは状態。
パルーン	コントロールが入力に何らかの影響を及ぼす状況にあることを示す場合(意図せず設定された可能性の高い状態で、ユーザーが入力への影響を認識していない可能性がある)。

- モーダルダイアログボックスでは、次の点に注意します。

- 適切な場合は常にタスクダイアログボックスを使用します。タスクダイアログボックスを使用することによって、一貫した外観とレイアウトを実現できます。タスクダイアログボックスを使用するには、Windows Vista®以降が必要であるため、以前のバージョンのWindowsには適しません。
- 状況ごとに1つの警告メッセージだけを表示します。たとえば、メッセージを1回ずつ表示して状況の詳細を示す代わりに、状況を完全に説明する警告メッセージを1回だけ表示します。ある状況に対して、複数の警告ダイアログボックスを続けて表示すると、混乱を招き、わざわざになります。
- 1つの状況で複数の警告を表示しません。頻繁に警告が表示されると、非効率でわざわざ感じます。ユーザーが、問題に対処することより、警告表示を消すことに注意を払うようになります。1つの状況に対して警告を繰り返して表示する必要がある場合は、[段階的なエスカレーション](#)を使用します。

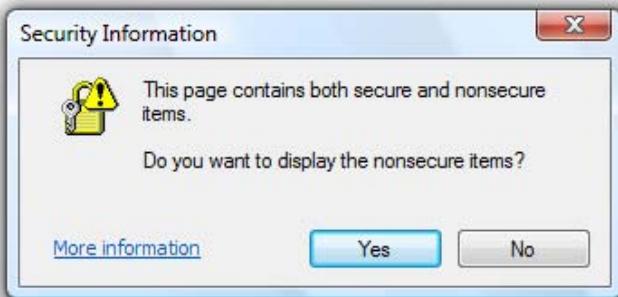
- 警告に効果音やビープ音は使用しません。耳障りになることが多く、必要性もありません。

- 例外: ユーザーが直ちに対応する必要がある場合は、サウンド効果を使用できます。

### アイコン

- ダイアログボックスのタイトルバーには警告アイコンを配置しません。
- 警告アイコンを使用します。次の場合は例外です。
  - アイコンが割り当てられている機能に対する警告の場合は、その機能のアイコンに警告アイコンを重ねて表示します。

正しい例:



この例では、該当する機能のアイコンに警告アイコンを重ねて表示しています。

- 警告が脚注に表示されるモーダル ダイアログ ボックスでは、コンテンツ エリアではなく脚注に警告アイコンを配置します。

正しい例:



この例では、警告アイコンが脚注に表示されています。

その他のガイドラインと例については、「[標準アイコン](#)」を参照してください。

今後、このメッセージを表示しない

- 警告ダイアログ ボックスに、このオプションが必要な場合は、警告とその頻度をもう一度よく考えてください。完成度の高い警告(リスクに関連する、即応性と関連性が高い、実行性が高い、明白ではない、頻度が低い)の特徴をすべて備えているのであれば、ユーザーがその表示を制限する理由はありません。

その他のガイドラインについては、「[ダイアログ ボックス](#)」を参照してください。

段階的表示

- 警告メッセージに詳細な情報を記載する必要がある場合は、[段階的表示](#)ボタンを使用して提示します([詳細の表示]など)。標準的な用途では、このようにして警告を簡素化します。必要な情報は非表示にしないでください。非表示にすると、ユーザーが気づかない可能性があります。
- より詳細な情報が実際にある場合以外は、[詳細の表示]は使用しません。既存の情報を別の形式で単に言い換えることはしないでください。

ラベルのガイドラインについては、「[段階的表示](#)」を参照してください。

既定値

- 既定値には、最も安全で、最も障害の起こる可能性が低く、最もセキュリティの高い応答が得られる値を選択します。

テキスト

全般

- 冗長なテキストは削除します。タイトル、メイン指示テキスト、補足指示テキスト、コンテンツ エリア、コマンド リンク、コミット ボタンなどに冗長なテキストがないか確認してください。通常、メイン指示テキストと対話型コントロールのテキストはそのまま残し、他の部分にある冗長なテキストを削除します。
- テキストでは "警告" または "注意" という用語を使用しないでください。警告アイコンを[正しく使用](#)すれば、注意しながら操作を継続する必要があることをユーザーに効果的に伝えることができます。

間違った例:

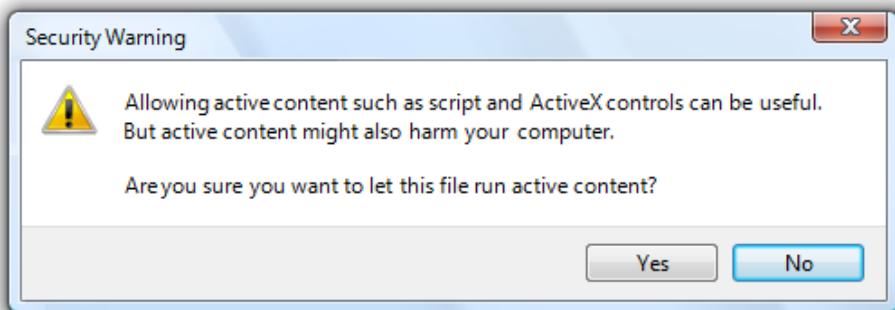


この例では、"WARNING" (警告) という用語は不要です。

#### タイトル

- タイトルに警告の発生源となったコマンドや機能を明記します。次の場合は例外です。
  - いくつもの異なるコマンドによって警告が表示される場合は、プログラム名を明記するようにします。
  - タイトルが冗長であったりメイン指示テキストと紛らわしい場合は、代わりにプログラム名を使用します。

間違った例:



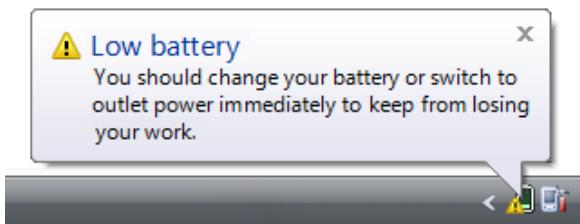
この例のタイトル "Security Warning" (セキュリティに関する警告) は、警告の発生源となったコマンドや機能を示していません。

- タイトルで、ダイアログ ボックスで実行する操作を説明することはしません。説明はメイン指示テキストで行います。
- タイトルスタイルの大文字化**を使用し、末尾に句読点は付けません。

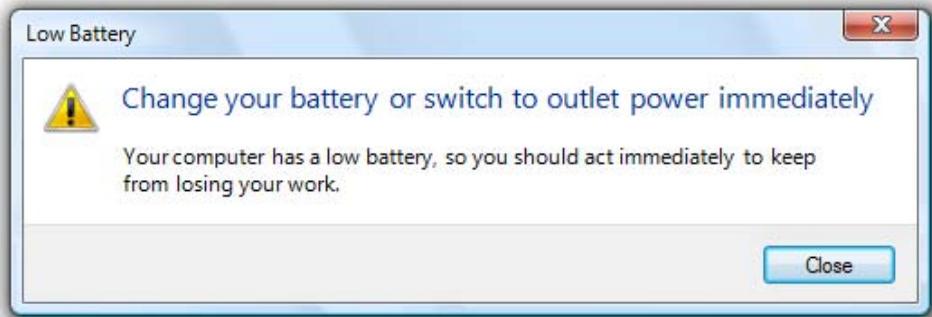
#### メイン指示テキスト

- 警告のメイン指示テキストは、設計パターンに基づいて決定します。

パターン	メイン指示テキスト
情報伝達	状況または発生する可能性の高い問題を示します。
切迫している問題	ユーザーが直ちに実行する必要のある操作を示します。
リスクを伴う操作の確認	ユーザーが操作を継続するかどうかを判断するための質問を提示します。



この例のバッテリ残量の減少を示す通知は情報伝達警告であるため、メイン指示テキストでは状況が説明されています。



この例のバッテリ残量の減少を示すダイアログ ボックスは切迫している問題であるため、メイン指示テキストではユーザーが直ちに実行する必要のある操作が示されています。

- 簡潔な 1 文だけを使用します。メイン指示テキストは必須の情報だけにそぎ落とし、さらに説明が必要な場合は、補足指示テキストを使用します。
- ユーザーが直ちに実行する必要がある場合は、"今すぐ"、"直ちに"などの言葉を使用します。緊急性が高くない場合には、これらの言葉を使用しません。
- 具体的な内容にします。関連するオブジェクトがある場合は、その完全な名前を提供します。
- センテンススタイルの大文字化**を使用します。

#### 補足指示テキスト

- 警告の補足指示テキストは、設計パターンに基づいて決定します。

パターン	補足指示テキスト
情報伝達	発生する可能性のある問題とその重要性を説明します。
切迫している問題	状況とその重要性を説明します。
リスクを伴う操作の確認	ユーザーが操作を継続しない方がよい、自明でない理由を説明します。

- メイン指示テキストの言い回しを少し変えて繰り返すことは避けます。追加する情報がない場合、補足指示テキストは省略します。
- 文を使用し、末尾に句点を付けます。センテンススタイルの大文字化を使用します。

#### コミット ボタン

- 警告ダイアログ ボックスでは、設計パターンに基づいてコミット ボタンを決定します。

パターン	コミット ボタン
情報伝達	[閉じる] を使用します。[OK] は使用しないでください。発生する可能性のある問題を容認するような印象を与えてしまいます。
切迫している問題	各オプションに対応するコマンド ボタンまたはコマンド リンクを使用します。ダイアログ ボックス以外の場所で操作を実行する必要がある場合は、[OK] を使用します。
リスクを伴う操作の確認	[はい]、[いいえ] を使用します。

間違った例:



問題が発生していることは OK ではないので、[閉じる] を使用します。

#### ドキュメント

警告に言及するときは、以下のことに留意します。

- 警告が質問形式の場合は、警告はその質問で示します。それ以外の場合は、メイン指示テキストを使用します。質問またはメイン指示テキストが長い、または詳細な場合は、内容を要約します。
- 必要に応じて、警告ダイアログボックスを "メッセージ" と呼ぶこともできます。
- テキストを二重引用符 (" ") で囲み、可能な場合は太字にします。

例: "続行しますか?" というメッセージが表示された場合は [はい] をクリックします。

## 確認

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

一般

アイコン

コミットボタン

コマンドリンク

既定値

今後、このメッセージを表示しない

一括操作

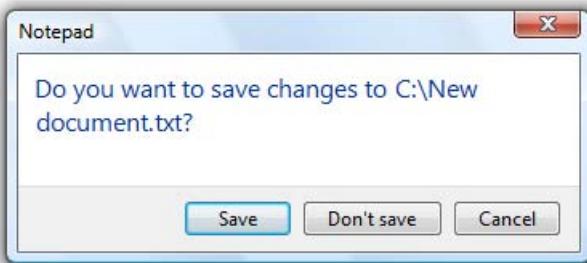
段階的表示

ユーザー アカウント制御

テキスト

ドキュメント

"確認" は、ユーザーが操作を続行するかどうかをたずねるモーダルダイアログボックスです。



典型的な確認

確認の基本的な特徴は次のとおりです。

- ユーザーが開始した操作の直接の結果として表示される
- ユーザーが操作を続行するかどうかを確認する
- 簡単な質問と 2 つ以上の応答で構成される

確認が最も有用なのは、その操作が、ユーザーによる(関連性のある明確な)選択を必要とする場合で、その選択が、後からは行えない場合です(この選択には、ユーザーにとって明白でないある種のリスク要素が関係することも多くあります、リスクがあるから確認が必要になるというわけではありません)。このような要素は、モーダルダイアログボックスへの応答で作業が中断されることに対する十分な理由になります。

対照的に、警告メッセージは、今後問題を引き起こす可能性のある状況を示します。警告メッセージの基本的な特徴は、次のようなリスクが関係することです。

- 次のうち 1 つ以上が失われる可能性がある
  - 価値の高い資産(データ損失、金銭的な損失など)
  - システムへのアクセスまたは整合性
  - プライバシーまたは機密情報の制御
  - ユーザーの時間(30 秒かそれ以上の長時間)
- 予期しない、または意図しない結果をもたらす
- 今すぐ適切な対処が必要(間違いが簡単には修正できず、元に戻せない可能性もある)

確認にリスクが関係する場合、その確認は警告と見なすこともできます。その場合は警告メッセージのガイドラインにも従います。

注: ダイアログボックス、エラーメッセージ、警告メッセージ、レイアウトに関するガイドラインは、それぞれ別の項目として記載しています。

## 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- ユーザーに操作の続行をたずねる質問で、2 つ以上の応答があるかどうか。該当しない場合は、そのメッセージは確認ではありません。
- UI が、発生したエラーまたは問題を示すものかどうか。該当する場合は、代わりにエラーメッセージを使用します。
- 操作の続行にはユーザーの選択が必要で、選択肢の中に既定とするのに適切なものがないかどうか。該当する場合は、おそらく確認が適切です。
- ユーザーがリスクのある操作を実行しようとしているか。該当する場合で、その操作が重要な結果をもたらすか、容易に元に戻せないものである場合は、確認が適切です。
- ユーザーがタスクを途中でやめようとしているか。該当する場合は、確認を表示しません。ユーザーはタスクを完了しないことによる結果を理解していると見なします。

- 操作によって生じる結果を、ユーザーが認識していない可能性があるかどうか。該当する場合は、おそらく確認が適切です。
- 現在のコンテキストで、ユーザーがエラーになる操作を実行する可能性があるかどうか。該当する場合は、おそらく確認が適切です。
- ユーザーがその操作を頻繁に実行するか。該当する場合は、別の設計を検討します。頻繁な確認はわざらわしく、ほとんど意味がありません。ユーザーは、考えずに応答するようになってしまいます。
- 確認の必要がなくなるような、別の設計ができるかどうか。確認が必要な設計には、問題がある可能性もあります。確認の必要のない、より適切な別の設計ができることも少なくありません。
- 操作にセキュリティ上の意味合いがあるか。該当する場合は、これまでの検討事項で確認を使用しないことになったとしても、確認の表示が必要な可能性があります。

## デザイン コンセプト

不必要的確認はわざらわしい

初期に作成されていた Windows の確認が、次のようなものであったことは否定できません。



### 初期のわざらわしい確認

これは非常に悪い出発点でした。このような確認を全体に散りばめれば、そのプログラムはユーザーから嫌われるものになります。この理由を理解するには、ユーザーの視点で考えてみるとわかります。何を偶然クリックしたか押したのでなければ、ユーザーは当然操作を続行しようとしています。このようなときに、操作の実行を単にたずねるために、定義どおりに確認が表示されたらどうでしょうか。

不必要的確認はわざらわしいだけでなく、間違いからユーザーを守る効果もありません。ユーザーはプログラムで不必要的確認が表示されることをすぐに見抜き、自然な反応として、できるだけ早く確認を消そうとします。ほとんどの場合は読むこともしません。結果的に、このような確認は、タスクに余分な手順を追加するだけになります。

ユーザーが間違いをする可能性があるという理由だけで確認を使用しないでください。確認は、重要な結果や意図しない結果をもたらす操作を確認するために使用すると最も効果的です。適切な確認は、明白なことを確認するものではなく、ユーザーが認識する必要があること、つまり続行しない十分な理由を伝えるものです。このような確認は、操作に本当に必要なときだけ使用されます。たとえば、保存が必要な可能性がある変更があるときだけ、変更を保存するかどうかをユーザーにたずねます。これで、正当な場合にだけ、ユーザーの注意を引くことになります。

これ以外の種類の確認の場合、多くは、ユーザーに対して質問への応答を強制しないような、より適切な別の設計が可能です。

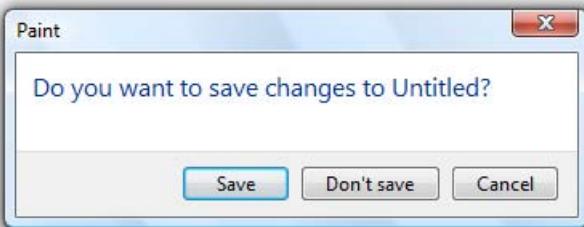
### 別の設計を検討する

日常的な操作に対する確認がなくなるような、別の設計の例を以下に挙げます。

- エラーを防ぐ。大きな間違いが偶然に起きたくいようにタスクを設計します。たとえば、リスクのあるコマンドを他のコマンドと物理的に隔てて、完了までに複数の操作が必要になるようにします。
- 元に戻す機能を用意する。ユーザーが操作を元に戻せるようにします。たとえば、Microsoft® Windows® でファイルを削除するときには、通常、確認は必要ありません。削除されたファイルをごみ箱から回復できます。操作が簡単に実行できるものであれば、元に戻す機能だけで十分な場合が多くあります。
- フィードバックを提供する。望ましくない結果を明確に示します。ユーザーがいつ間違えたかを認識していない場合、元に戻す機能があるだけでは不十分です。たとえば、直接操作（ドラッグ アンド ドロップ操作など）の結果は、常に明確に示す必要があります。
- 考えられる結果を想定しつつ、容易に変更できるようにする。ユーザーが何をしたいかがわからなくても、可能性のありそうな、安全でセキュリティ上問題ない選択肢があれば、その選択を想定します。結果を明確に示し、コンテキストメニューやスマート タグを使用して容易に変更できるようにします。たとえば、Microsoft Word では、ユーザーが単語を正しく表記しようとしていることが想定されています。間違った表記の単語が見つかり、正しい候補がある場合、Word では自動的に修正が行われますが、ユーザーが元に戻すこともできます。
- 選択肢を一切用意しない。選択が重要でなければ、ユーザーは気にしません。プログラムを簡素化し、選択肢をなくすのが良い方法です。

### 考えさせる確認にする

確認を有意義なものにするには、続行しない理由をユーザーが理解することが必要です。変更のあるドキュメントを保存しないで閉じようとしているときなど、理由が明らかな場合もあります。

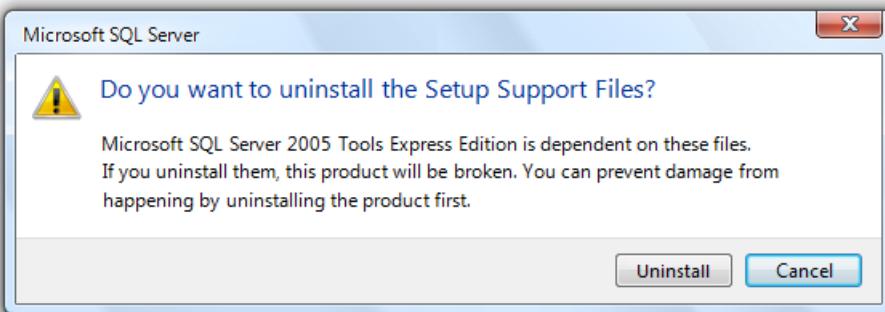


この例では、確認の理由は明らかです。

状況によっては、理由がそれほど明らかではないこともあります。

ダイアログ ボックスのコミット ボタンのラベルを選択する際の一般的なガイドラインは、メイン指示テキストに対応した具体的な応答のラベルを選択することです。こうすると、ユーザーは続行するのに最低限のテキストを読むだけで済むため、効率的な決定ができます。ただし、効率を目指すことが確認にとって逆効果となる場合もあります。次の例を考えてみます。

間違った例:

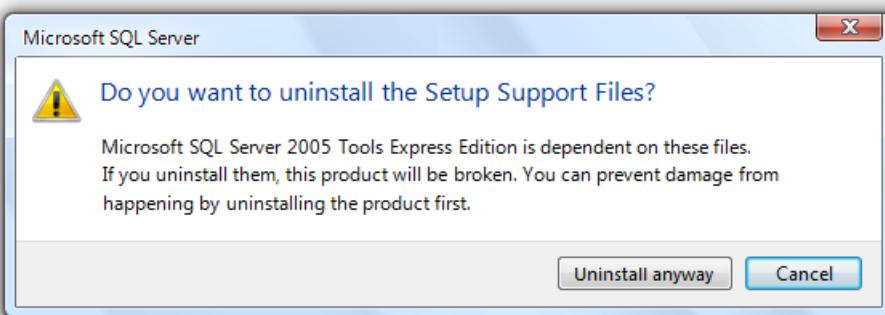


この例では、確認は、正しく応答するには一考が必要な内容です。

ユーザーがアンインストール コマンドを指定してすぐにこの確認が表示されたとすると、おそらくユーザーは "もちろんアンインストールする" と反応し、深く考えずにアンインストールのボタンをクリックするでしょう。

ユーザーに感情的な決定を急いで行わせることは、確認の意図するところではありません。ユーザーが応答について考えるようにするには、意思決定のために一呼吸置く必要があります。通常、慎重な言い回しのコミット ボタンを使用できるようであれば、そうする方が適切です。たとえば、言葉を追加して、続行しない理由があることを示すことができます。

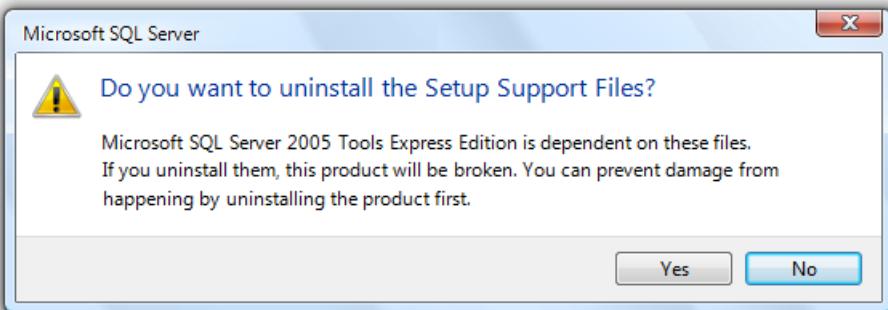
より良い例:



この例では、コミット ボタンのラベルに "とにかく (anyway)" という言葉が追加され、続行しないことの理由が説明されていることを示しています。

この方法が現実的でない場合は、[はい] と [いいえ] のコミット ボタンを使用できます。

別のこと:



この例では、[はい]と[いいえ]のコミット ボタンによって、ユーザーに少なくともメイン指示テキストを読ませるようにしています。

すべての情報を提供する

質問する際には、ユーザーがその質問に根拠を持って答えられるように、十分な情報を提供する必要があります。たとえば、Windows XP の[ファイルの上書きの確認]ダイアログ ボックスについて考えてみます。



Windows XP の[ファイルの上書きの確認]ダイアログ ボックス

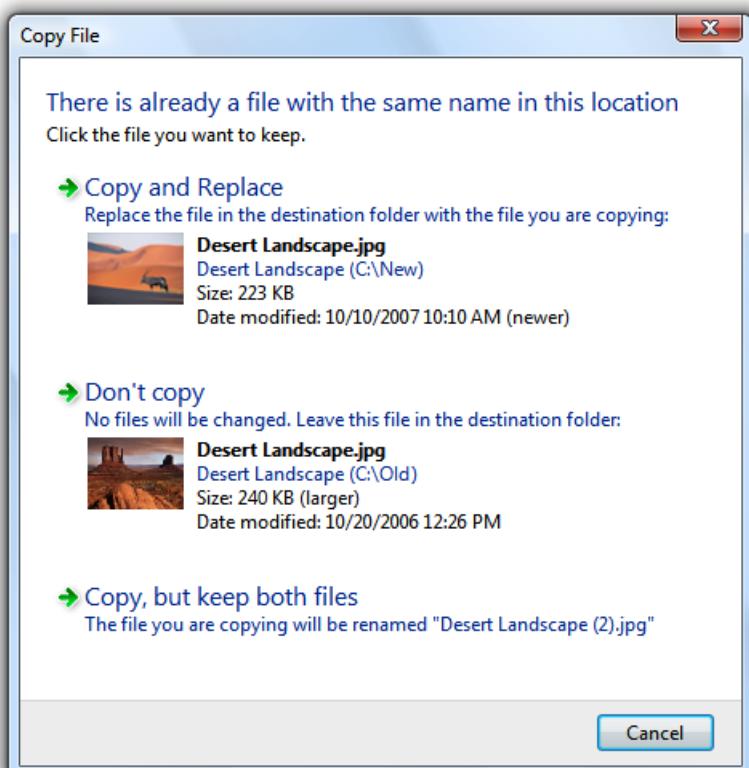
この確認で、ユーザーが質問に答えるのに必要な情報がすべて提供されているかどうかを考えるには、まず、最も一般的なユーザー シナリオを挙げてみます。それは次のとおりです。

1. もう 1 つのファイルをコピー(または移動)して、既存のファイルを置き換える
2. 既存のファイルを保持し、もう 1 つのファイルをコピーまたは移動しない
3. より新しいファイルを保持またはコピーする(最上位のシナリオ)
4. ファイルの内容やサイズなどの条件に応じて、既存のファイルを保持するか、もう 1 つのファイルをコピーするかを選ぶ
5. 既存のファイルを保持し、もう 1 つのファイルを違う名前でコピーする
6. 間違えたか予期していなかったため、操作を取り消す

シナリオ 1 の場合は、ユーザーは[はい]をクリックでき、シナリオ 2 の場合は、[いいえ]をクリックできます。シナリオ 3 の場合、ユーザーはファイルの日付を比較して、適切な方のボタンをクリックできます。しかし、最も一般的なシナリオである可能性が高いにもかかわらず、より新しいファイルを決定してから適切なボタンを決定するまでには、少し考える時間が必要になります。

シナリオ 4、5、および 6 もまた困難です。ファイル サイズは端数が切り捨てられるので、両方のファイルが同じサイズかどうかや、両方が同じファイルであるかどうかさえ、判断するのは不可能です。アイコンがファイルを開くアプリケーションを示していますが、ユーザーはこれを参考にファイルを開いて内容を調べるか比較する必要があります。ファイルの内容を示すサムネイルがあれば、はるかに応答の参考になります。

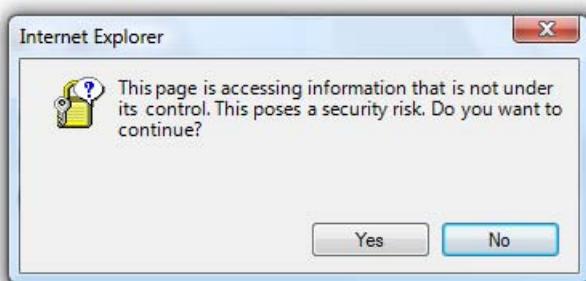
Windows Vista® の[ファイルのコピー]ダイアログ ボックスによる確認では、以前より多くの情報が提供され、両方のファイルを保持するオプションも追加されて、これらのシナリオにより適切に対応できるようになっています。



Windows Vista の [ファイルのコピー] ダイアログ ボックス

具体的で有用な情報を提供する

質問する際には、ユーザーが質問を理解し、別の応答をした場合にどうなるかを認識できるようにします。たとえば、Windows Internet Explorer® のセキュリティの確認について考えてみます。



#### あいまいなセキュリティの確認

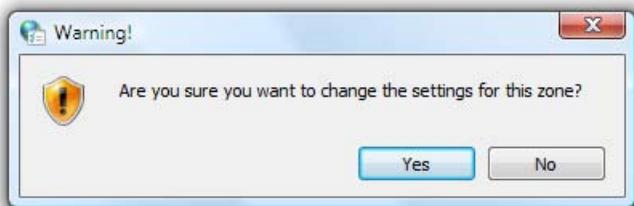
この確認は、ユーザーが根拠を持って答えられないような質問をしています。ユーザーは Windows Internet Explorer に対し、ページを表示することを要求しました。一方このメッセージでは、テキストの言い回しと、既定の選択肢として [いいえ] が強調表示されていることで、暗に反対のことが勧められています。

ページを表示すると引き起こされる具体的なセキュリティ上の問題について十分な説明はないため、続行することのリスクは明らかではありません。この確認の中に、ユーザーが [いいえ] をクリックする根拠となるような情報は見つかりません。メッセージがあいまいなため、この確認によりユーザーが続行をやめる可能性はあまりなく、ユーザーはただ嫌な気分になるだけです。

有用な確認にするためには、より多くの情報、つまりユーザーに続行しないと決断させるための具体的な情報を提供する必要があります。一般的には、確認のそれぞれの応答について、その応答が必要になるシナリオを考え、その応答を選択しようとするユーザーに十分な情報が提供されるようにします。提供するのは選択肢であり、ジレンマではありません。

#### 確認が必要かどうかを判断する方法

シナリオをすべて検討し、それぞれの応答が選択される可能性を考えると、確認が必要かどうかを体系的に判断できるようになります。ユーザーがすべての応答を選択する可能性がある場合、その確認は必要で有用です。1つの応答しか選択されない場合(たとえばその応答が常に 98% の確率で選択されるような場合)、明らかにその確認は不要で、削除する必要があります。ただし、セキュリティ上、法律上、および安全上の問題に関連する確認は、例外となることもあります。



この確認は必要なものでしょうか。また、ユーザーは「いいえ」を選択するでしょうか。可能性はあります、それはきわめて低いと考えられます。したがって、この確認は削除する必要があります。

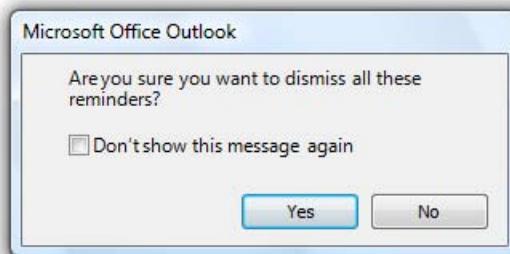
### 3つの重要な点

- 確認が本当に必要であることを確かめます。続行しない正当かつ明確な理由と、ユーザーがそれに従わない可能性がある場合にのみ、確認を使用します。
- 確認の理由が一見してわかるものでない場合は、ユーザーが応答について一考するようなコミットボタンにします。一般的には、確認部分は「はい」か「いいえ」で答えられるような質問形式にして、応答部分は、それだけで明確に意味が伝わるようなラベルにするか、「はい」と「いいえ」にします。
- すべてのシナリオを検討し、ユーザーが根拠を持って答えられるように、必要な情報を提供します。

## 使用パターン

確認にはいくつかの使用パターンがあります。

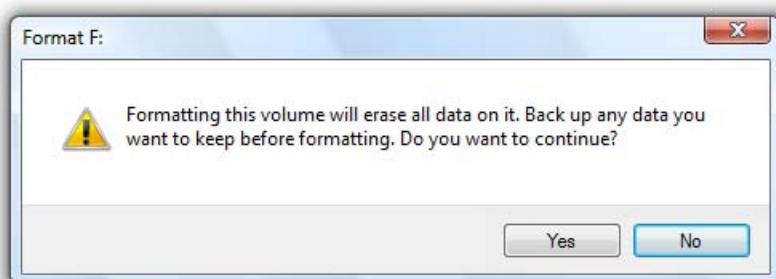
日常的な操作に対する確認	このパターンの確認は、通常、「(本当に) ... してもよろしいですか?」のような言い回しを使用します。 多くの場合、わずらわしさが最小限になるように、[今後、このメッセージを表示しない] チェックボックスを用意します。
リスクの少ない操作について、ユーザーが続行するかどうかを確認します。	

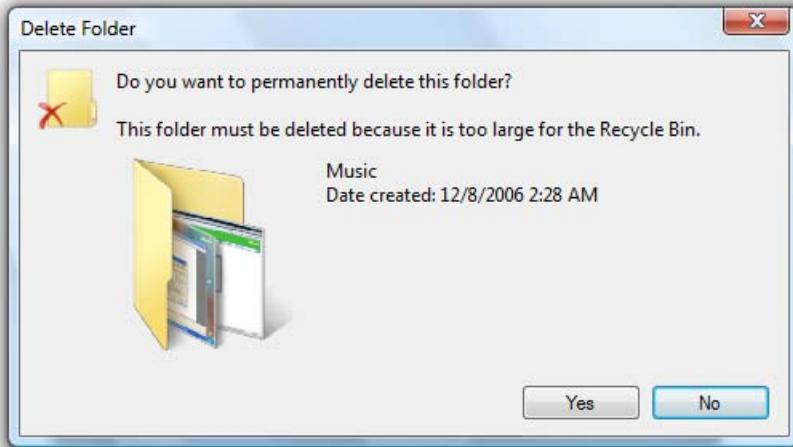


日常的な操作に対する確認の例。

注: 通常このパターンは不要であり、使用は避けるようにしてください。

リスクのある操作に対する確認	リスクのある操作であるため、通常は警告アイコンを使用します。
ユーザーが実行しようとしている操作にリスクが伴い、簡単に元に戻すことができない場合は、操作を継続するかどうかを確認します。	

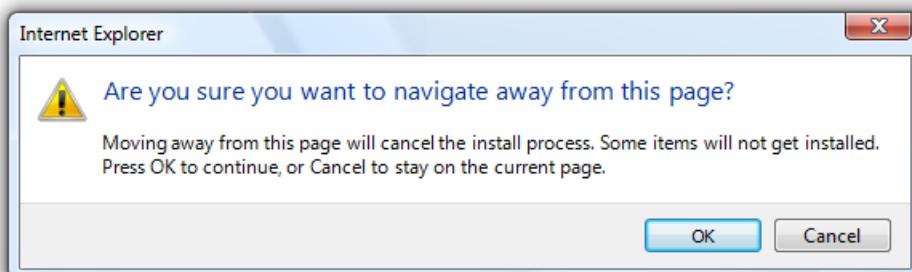




リスクのある操作に対する確認の例。

意図しない結果の確認  
予期または意図しない結果をもたらす操作について、ユーザーが続行するかどうかを確認します。

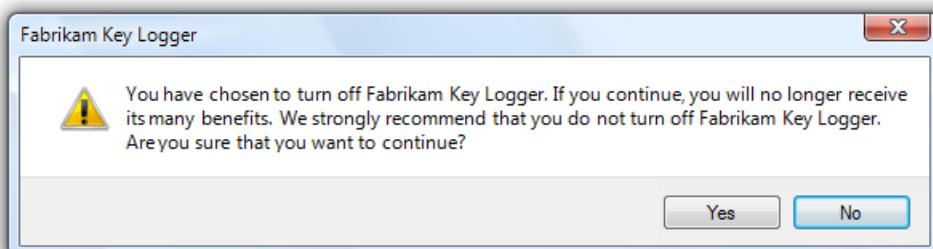
この確認では、質問を投げかけるだけでなく、意図しない結果を指摘します。意図しない結果をもたらす操作であるため、通常は警告アイコンを使用します。



意図しない結果の確認の例。

このパターンの使用は、結果が本当に意図しないものである場合に限られます。

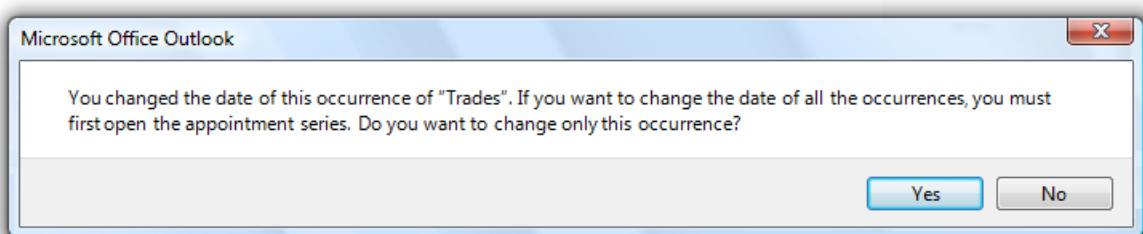
間違った例:

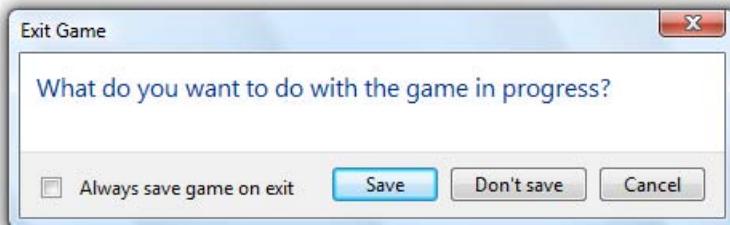


この結果は意図したものなので、これは日常的な操作に対する確認になります。

明確化のための確認  
あいまいな結果や予期しない結果をもたらす可能性のある操作について、ユーザーがどのように続行するかを明確にします。

ドラッグアンドドロップ操作が行われたとき、この操作の結果が誤解されている可能性のある場合は、明確化のための確認が必要になることがあります。





明確化のための確認の例。

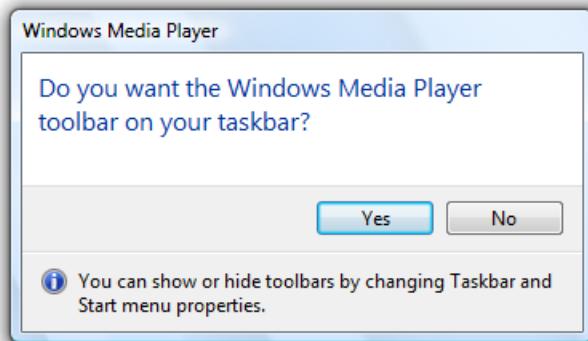
注: このパターンは使用しないようにします。あいまいな結果が生じないような操作を設計し、最も可能性が高く望ましい結果を想定するようにします。

セキュリティ上の確認  
セキュリティが関係する結果を伴う操作について、ユーザーが続行するかどうかを確認します。



セキュリティ上の確認の例。

別の意図のある確認  
操作に関する情報、確認の形式で提供します。



別の意図のある確認の例

注: このパターンは推奨されません。通常はより適切で、より直接的な方法が他にあります。たとえば、原因と結果の関係を示すには、[アニメーション](#)の方がより適しています。

## ガイドライン

### 全般

- ・ "変更の保存" の確認は、重要な変更があるときだけ使用します。ドキュメントの書式の自動再設定など、ユーザーが直接行っていない変更に対しては確認を表示しません。

間違った例:



この例の確認は、ユーザーが変更していない空の電子メールやドキュメントに対して使用される場合は、正しくありません。

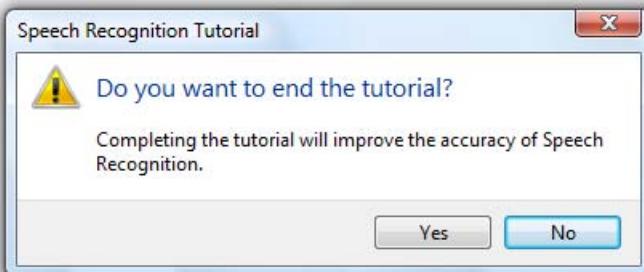
#### アイコン

- 確認ではタイトルバーのアイコンは使用しません。
- 確認のコンテンツエリアのアイコンは、設計パターンに基づいて決まります。

パターン	アイコン
日常的な操作に対する確認	アイコンを使用しません。
リスクのある操作に対する確認	警告アイコン。
意図しない結果の確認	リスクがある場合は警告アイコンを使用し、機能アイコンがあれば機能アイコンを使用します。それ以外は、アイコンを使用しません。
明確化のための確認	確認がドキュメントに関するもの場合は、ドキュメントのサムネイルを使用します。それ以外の場合は、機能アイコンがあれば機能アイコンを使用するか、アイコンを使用しません。
セキュリティ上の確認	警告アイコン。
別の意図のある確認	アイコンを使用しません。

- 日常的な操作に対する質問には警告アイコンを使用しません。このような方法での警告アイコンの使用は、[Windows のトーン](#)の推奨に反し、プログラムの使用を危険な操作だと感じさせることになります。ユーザーは完了前にタスクを取り消すことによる結果を理解していると見なします。

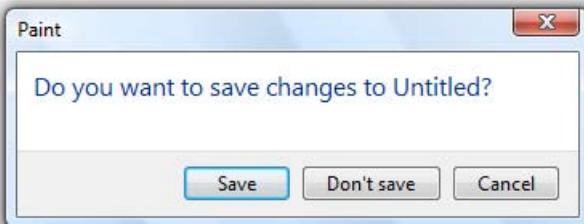
間違った例:



この例では、日常的な操作に対する質問に警告アイコンが使用されています。

#### コミット ボタン

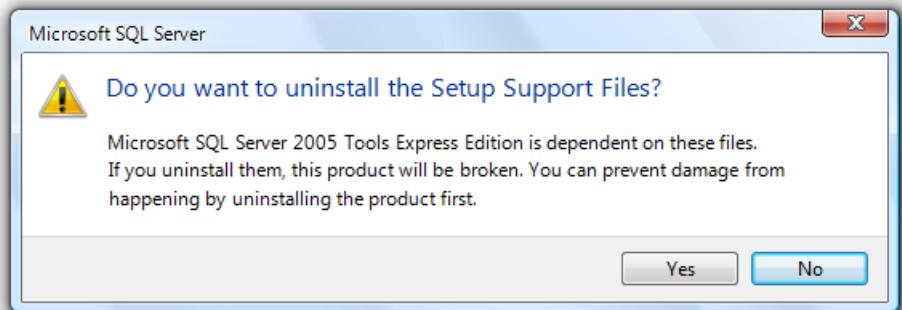
- 確認の理由が明らかな場合や、それだけで意味が伝わるような応答にできる場合は、メイン指示テキストに対応した具体的な応答を使用します。



この例では確認の理由が明らかなので、[保存] と [保存しない] は適切な応答です。

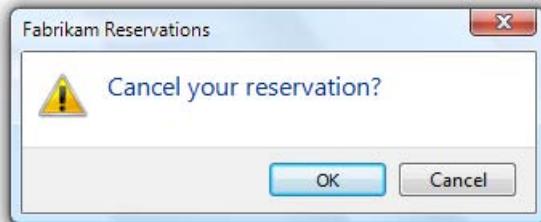
- 上記以外の場合は、[はい] および [いいえ] ボタンを確認の応答に使用します。こうすると、ユーザーは応答する前に確認内容について一考するようになります。[OK] と [キャンセル] を確認に使用しないでください。

正しい例:



この例では、[はい]と[いいえ]のコミット ボタンによって、ユーザーに少なくともメイン指示テキストを読ませるようにしています。

間違った例:

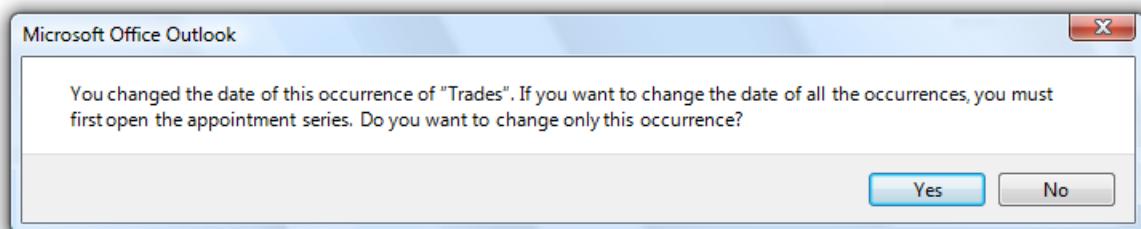


この例では、[OK] および [キャンセル] ボタンの使用が混乱を招きます。

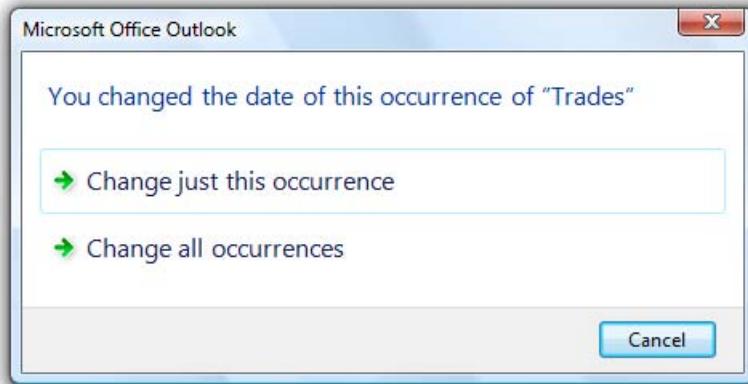
コマンド リンク

- 明確化のための確認のパターンには、代替案を明確にするコマンド リンクを使用するようにします。

許容される例:



より良い例:



より良い例では、コマンド リンクを使用することで代替案が明確になっています。

- 最も一般的に使われるコマンド リンクを先頭に置きます。使用される頻度順に主に従って並べますが、論理的にも自然になるようにします。
- コマンド リンクにさらに詳しい説明が必要な場合は、補足説明を記述します。補足説明には、そのオプションをどのような場合に選択する必要があるか、選択すると何が起こるかを記載します。

その他のガイドラインと例については、「[コマンド リンク](#)」を参照してください。

既定値

- 確認の既定の応答は、設計パターンに基づいて決まります。

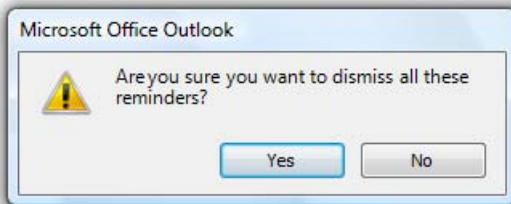
パターン	既定の応答

日常的な操作に対する確認	続行する。
リスクのある操作に対する確認	続行しない(または安全な選択)。
意図しない結果の確認	重大な結果の場合は続行しない。それ以外の場合は続行する。
明確化のための確認	最も可能性の高い応答。
セキュリティ上の確認	続行しない。
別の意図のある確認	続行する。

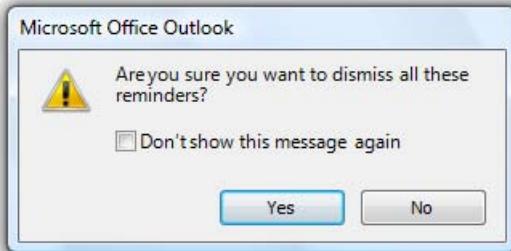
今後、このメッセージを表示しない

- このオプションは、日常的な操作に対する確認と、別の意図のある確認のパターンにのみ使用します。他のパターンの場合、必要な情報は常に表示される必要があります。
- 不必要的確認を表示することを正当化するためにこのオプションを追加することは避けます。その場合は、確認自体を削除します。

間違った例:



別の間違った例:

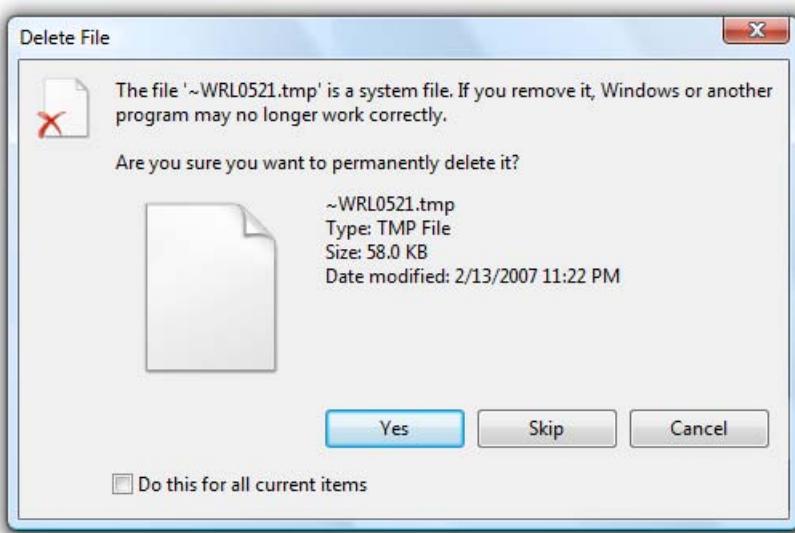


これらの例のように「今後、このメッセージを表示しない」オプションを追加することで、不必要的確認が修正されているわけではありません。

その他のガイドラインについては、「[ダイアログ ボックス](#)」を参照してください。

#### 一括操作

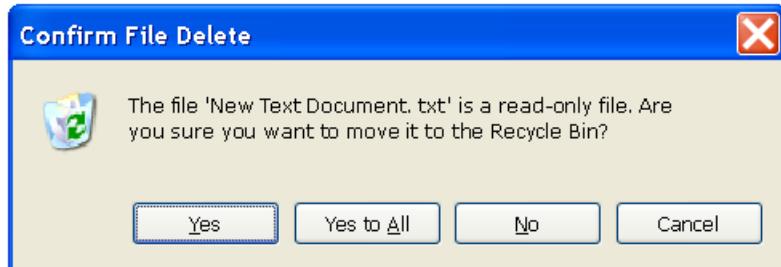
- 一括操作に適用される確認には、その確認を全部の操作に適用するオプションを追加します。



この例では、一括操作用のオプションが追加されています。

- 一括操作の確認を行わないか、後回しにします。

間違った例:



この例では、Windows XP の Windows エクスプローラーによって、ファイルの一括移動中に読み取り専用ファイルについて 1 つずつ確認が表示されます。読み取り専用ファイルの場合は確認なしでコピーするだけにするか、そのようなファイルの処理は後回しにしてタスクの最後に確認を表示する方がより適切です。

#### 段階的表示

- 確認メッセージに詳細な情報を含める必要がある場合は、[段階的表示](#) ボタン([詳細の表示]など)を使用して提供します。こうすると、通常の使用方法のように確認を単純化することができます。必要な情報は非表示にしないでください。非表示にすると、ユーザーが気づかない可能性があります。
- より詳細な情報が実際にある場合以外は、[詳細の表示]は使用しません。既存の情報を別の形式で単に言い換えることはしないでください。

ラベルのガイドラインについては、「[段階的表示](#)」を参照してください。

#### ユーザー アカウント制御

- ユーザー アカウント制御 (UAC) の[昇格 UI](#)を、確認の代用にはしません。操作に確認が必要な場合は、個別のダイアログ ボックスを使用します。昇格 UI でユーザーが考えなければならないのは、自分がタスクを開始したかどうかと、プログラムが信頼できるかどうかだけです。
- 確認は、昇格 UI より前に表示します。こうすると、不要な昇格が行われなくなります。

#### テキスト

##### 全般

- 冗長なテキストは削除します。タイトル、メイン指示テキスト、補足指示テキスト、コンテンツ エリア、コマンド リンク、コミット ボタンなどに冗長なテキストがないか確認してください。通常、メイン指示テキストと対話型コントロールのテキストはそのまま残し、他の部分にある冗長なテキストを削除します。
- テキスト内に "警告" や "注意" という語は使用しません。ユーザーが注意して続行する必要がある場合は、代わりに[警告アイコン](#)を使用します。

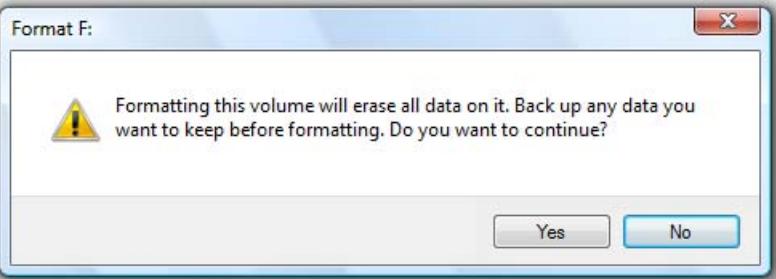
間違った例:



この例では、"WARNING" (警告) という用語は不要です。

#### タイトル

- 確認が表示される元となったコマンドまたは機能を特定するタイトルを使用します。次の場合は例外です。
  - 数種類のコマンドによって表示された確認の場合は、代わりにプログラム名を使用するようにします。
  - タイトルが冗長であったりメイン指示テキストと紛らわしい場合は、代わりにプログラム名を使用します。
- ただし、時間がかかるタスクによって表示される確認で、タスクの開始後に表示される可能性が高い確認の場合は、コンテキストを明確に示すために、常にコマンドまたは機能の名前を使用します。
- タイトルで、ダイアログ ボックスで実行する操作を説明することはしません。説明はメイン指示テキストで行います。
- より明確にするには、"確認" という言葉をタイトルに含めます。
- リスクのある操作に対する確認の場合は、特に強調するために、関連するオブジェクトの名前を追加することもできます。

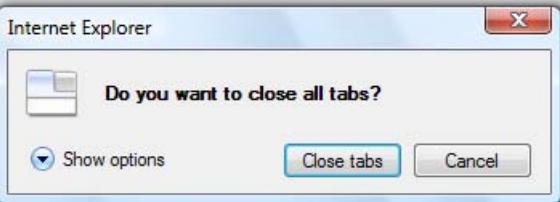


この例では、フォーマットされるドライブがタイトルに含まれています。

- タイトルスタイルの大文字化を使用し、末尾に句読点は付けません。

#### メイン指示テキスト

- 確認のメイン指示テキストは、設計パターンに基づいて決まります。

パ タ ン	メイン指示テキスト
意図しない結果の確認	意図しない結果について説明します。 例外: ユーザーに続行するかどうかをたずねる質問で、意図しない結果が明確にわかる場合は、確認の代わりに質問を使用します。
その他のすべて	 <p>この例では、ユーザーに続行するかどうかをたずねることで、操作の結果は伝わります。</p>
	ユーザーが続行するかどうかを判定する 1 つの質問を記述します。

- 簡潔な 1 文だけを使用します。メイン指示テキストは必須の情報だけにそぎ落とし、さらに説明が必要な場合は、補足指示テキストを使用します。
- 具体的な内容にします。関連するオブジェクトがある場合は、その完全な名前を提供します。
- 肯定文を使用します。肯定文の方が、ユーザーは理解しやすくなります。

正しい例:

ファイルとプリンターの共有を有効にしますか?

間違った例:

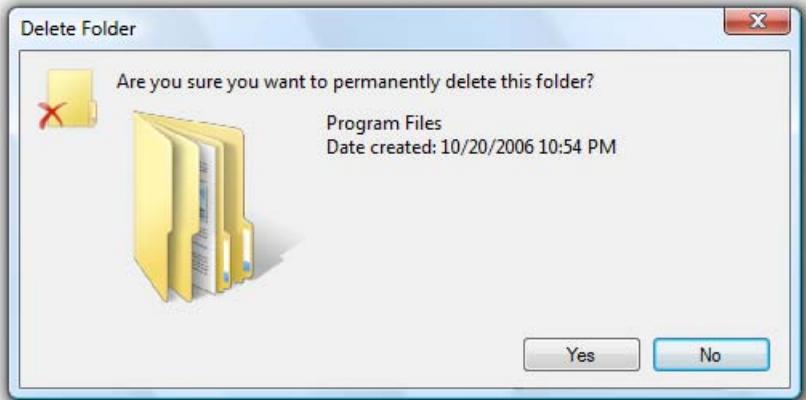
ファイルとプリンターの共有を無効にしますか?

ただし、表現は関連付けられているコマンドと一致させる必要があります。コマンドが否定的な表現であった場合、たとえば [無効にする] というコマンドの確認には、"無効にする" という表現を使用します。

- 表現に厳密なルールはありませんが、次のような一般的に使用される確認の表現には、言外の意味があります。

表現	言外の意味
(本当に) [操作を実行] してもよろしいですか?	ユーザーの要求から生じる直接の結果を確認
[操作を実行] しますか?	ユーザーの要求から生じる副次的な影響を確認
[結果を選択] しますか?	明確化が必要
[操作を実行]?	なし

- リスクのある操作に対する確認の場合は、"完全に" または "永久に" という語を使用して、操作を元に戻せないことを示します。



この例では、"完全に" を使用して、操作を元に戻せないことを示しています。

- センテンススタイルの大文字化を使用します。

#### 補足指示テキスト

- 確認の補足指示テキストは、設計パターンに基づいて決まります。

パターン	補足指示テキスト
意図しない結果の確認	ユーザーが続行するかどうかを判定する1つの質問を記述します。
その他すべて	ユーザーが操作を継続しない方がよい、自明でない理由を説明します。たとえば、次のようなことを説明します。 <ul style="list-style-type: none"><li>次のうち1つ以上が失われる可能性がある<ul style="list-style-type: none"><li>価値の高い資産(データ損失、金銭的な損失など)</li><li>システムへのアクセスまたは整合性</li><li>プライバシーまたは機密情報の制御</li></ul></li><li>元に戻せない操作</li></ul>

- メイン指示テキストの言い回しを少し変えて繰り返すことは避けます。追加する情報がない場合、補足指示テキストは省略します。
- 意図しない結果の確認の場合は、続行しない理由があることを簡潔に示すために、"無視して" や "かまわず" という語を使用するようにします。こうすると、ユーザーがメイン指示テキストを見ていなかった場合の助けになります。詳細については、「[デザインコンセプト](#)」を参照してください。
- 文を使用し、末尾に句点を付けます。センテンススタイルの大文字化を使用します。

#### ドキュメント

確認に言及するときは、以下のことに留意します。

- タイトルがその確認だけのものである(つまりプログラム名ではない)場合は、タイトルを引用します。それ以外の場合は、メイン指示テキストを引用します。
- 必要に応じて、確認のダイアログボックスを"メッセージ"として示すこともできます。
- 大文字と小文字の区別を含め、テキストを正確に引用します。
- テキストを二重引用符(" ")で囲み、可能な場合は太字にします。

例: "ファイルのコピー" メッセージで、新しい方のファイルをクリックします。

## 通知

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

全般

通知の内容

通知のタイミング

通知の表示時間

通知する回数

通知のエスカレーション

対話操作

アイコン

通知のキュー登録

システムの統合

テキスト

ドキュメント

"通知"とは、通知領域のアイコンからバルーンを短時間表示して、現在のユーザー操作とは無関係のイベントをユーザーに通知するものです。通知は、ユーザーの操作または重要なシステムイベントが原因で生じることもありますが、Microsoft® Windows® またはアプリケーションから役に立つ可能性のある情報が通知で提供されることもあります。

通知内の情報は、有用で関連性のあるものですが、重大ではありません。このため、通知にすばやく対応する必要はなく、無視することもできます。



典型的な通知

Windows Vista® 以降では、通知の表示は 9 秒間に固定されています。ユーザーが退席中であったり、スクリーンセーバーが実行されている間は、通知はすぐには表示されません。この期間の通知は、Windows で自動的にキューに登録され、ユーザーが通常の操作を再開すると、登録された通知が表示されます。このため、こうした特定の状況を処理するために何かをする必要はありません。

開発者向け情報: SHQueryUserNotificationState API を使用すると、ユーザーがアクティブになったことを判断できます。

注: 通知領域、タスクバー、バルーンに関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

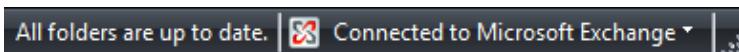
以下の点に基づいて判断します。

- 情報が、ユーザーのアプリケーションに対する対話操作によって即座にかつ直接的に生じるものであるかどうか。該当する場合は、ダイアログボックス、メッセージボックス、バルーン、インプレースなどの UI を使用するのではなく、直接アプリケーション内に同期情報を表示します。通知で扱うのは、非同期型の情報のみです。



この例では、Windows ファイアウォールの例外のダイアログ ボックスが、ユーザーの対話操作から直接生じたものとして表示されています。この場合、通知は適切ではありません。

- ユーザーが実際にアプリケーションを使用しているときにのみ関連する情報であるかどうか。該当する場合は、その情報をアプリケーションのステータスバーまたはその他のステータス領域に表示します。



この例では、Outlook のステータスバーに接続状態と同期状態が表示されています。

- 連続的に絶えず変化するリアルタイムの情報であるかどうか。例として、処理の進行状況、株価、スポーツ速報などがあります。該当する場合は、通知は頻繁に変化する情報に適していないので、このような情報には使用しないでください。
- 有用で関連性のある情報かどうか。または、その情報を受け取ることによって、ユーザーが操作を変えたり不都合な状況を避けたりできるかどうか。該当しない場合は、情報を表示しないか、ステータス ウィンドウまたはログ ファイルで対応します。
- 重大ですぐに対応が必要な状態かどうか。該当する場合は、このような情報は、モーダル ダイアログ ボックスまたはメッセージ ボックスのような、注意を喚起し、簡単には無視できないインターフェイスを使用して表示します。プログラムがアクティブではない場合は、プログラムのタスクバー ボタンを 3 回点滅させ、プログラムがアクティブになるまで強調表示することによって、重大な情報に注意を引き付けることができます。
- 主な対象ユーザーが IT プロフェッショナルであるかどうか。該当する場合は、通知ではなく、ログ ファイル エントリや電子メール メッセージなどのフィードバックのしくみを使用します。IT プロフェッショナルの場合、重大ではない情報にはログ ファイルを使用する方がいっそう好まれます。また、サーバーはリモート管理されることが多く、一般にはログオン ユーザーがいなくても実行されるので、通知を行っても効果的ではありません。

## デザイン コンセプト

### 概要:

ユーザー エクスペリエンスを向上させる効果的な通知を以下に示します。

- 非同期性(現在のユーザー操作とは無関係)、有用性、関連性、非重大性、対応可能性を備え、適切に表示されています。
- ユーザーが作業に集中しているときに簡単に無視できる方法で情報提供され、作業を中断しなくて済みます。
- ユーザーはすばやく対応する必要はなく、無視することもできます。
- 最も重要な点は、わざらわしくないことです。

### 3 つの重要な点

- 通知は、本当に必要な場合にのみ使用します。通知を表示すると、ユーザーの作業を中断したり、

ユーザーに不快感を与える可能性があります。正当な理由で中断が行われるようにします。

2. 通知は、ユーザーがすばやく対応する必要がない、重大ではないイベントまたは状況に使用します。ユーザーがすばやく対応する必要のある重大なイベントまたは状況の場合は、他の UI (モーダル ダイアログ ボックスなど) を使用します。

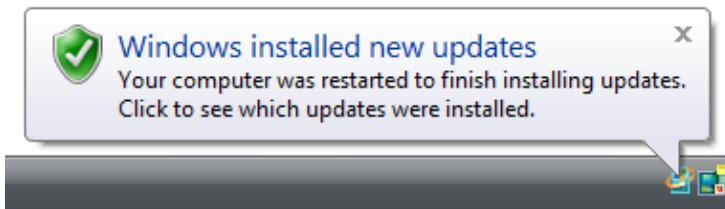
3.. 通知を使用する場合は、快適なユーザー エクスペリエンスをもたらすようにします。通知の確認は強制しないでください。ユーザーが作業に没頭しているときに通知に気付かなければ、設計は成功です。

詳細と例については、「[通知のデザイン コンセプト](#)」を参照してください。

## 使用パターン

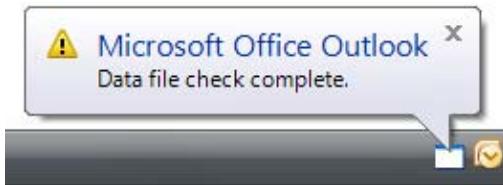
通知にはいくつかの使用パターンがあります。

**操作の成功** 正しい例:



この例では、Windows Update により、ユーザーのコンピューターが正常に更新されたことが通知されています。

間違った例:



この例では、Microsoft Outlook® により、データ ファイルのチェックが完了したことが通知されています。ユーザーは何を要求されているのかわかりません。また、正常に完了したことを警告で通知する理由も不明です。

**表示のタイミング:** 非同期型のタスクが完了した時点。操作の完了を待機している、または最近失敗したことがある場合にのみ、成功を通知します。

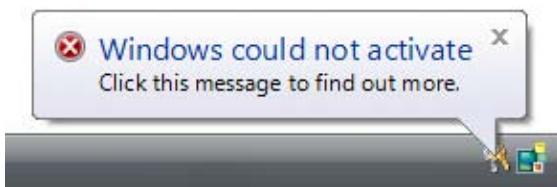
**表示方法:** リアルタイム オプションを使用し、ユーザーが全画面表示のアプリケーションを実行中の場合、および現在コンピューターを使用していない場合に、この種の通知がキューに登録されないようにします。

**表示回数:** 1 回

**迷惑度:** 少し前に失敗したため失敗を予期している場合、重大またはきわめてまれなエラーの後に成功したためユーザーに追加のフィードバックが必要な場合、またはユーザーが完了を待機している場合は低くなります。このような条件に該当しない場合は高くなります。

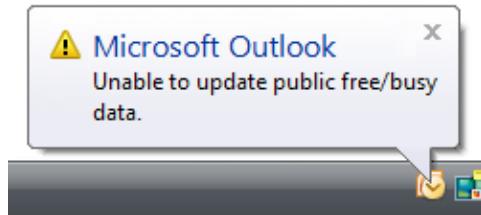
**代替方法:** 操作の実行中に通知領域にアイコンを表示するか既存のアイコンを変化させ、操作が完了したらアイコンを削除するか以前のアイコンに戻することで、"オン デマンド" のフィードバックを提供します。

**操作の失敗** 正しい例:



この例では、Windows ライセンス認証の失敗が通知されています。

間違った例:



この例では、*Microsoft Outlook*により、ユーザーが関心を持ちそうにないエラーが通知されています。

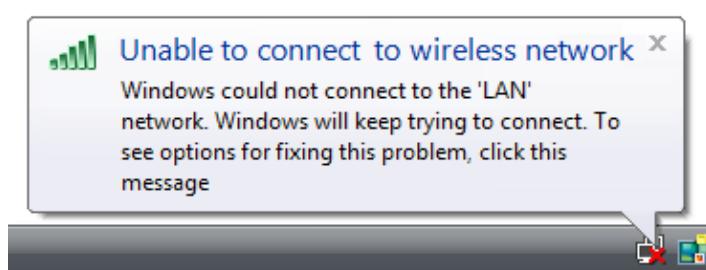
表示のタイミング: 非同期型のタスクが失敗した時点。

表示回数: 1回

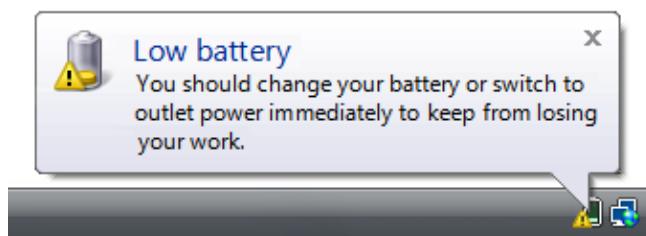
迷惑度: 有用で関連性のある場合は低くなります。その問題がすぐに自動的に解決されたり、ユーザーにとって関心のないものである場合は高くなります。

代替方法: ユーザーがエラーにすばやく対処する必要がある場合は、モーダルダイアログボックスを使用します。

重大ではないシステムイベント  
少なくとも一時的に無視しても安全な程度の重要性のシステムイベントまたは状態を通知します。



この例では、ワイヤレスネットワークへの接続に失敗したことが*Windows*から通知されています。



この例では、*Windows*からバッテリの低下が通知されていますが、ユーザーが対処するまでの時間は十分にあります。

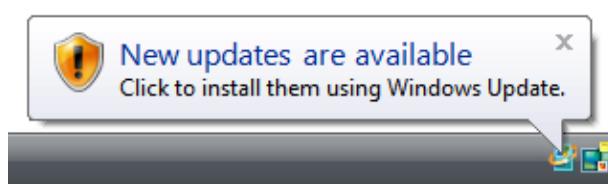
表示のタイミング: イベントが発生した時点でユーザーがアクティブであるとき、またはある状態が存続する場合。何らかの問題が原因で生じたものである場合、その問題が解決されたらすぐに、現在表示されている通知を削除します。操作に関する通知の場合、システムイベントの完了を待機していると思われる場合、または少し前の失敗の後にのみ、成功を通知します。

表示回数: イベントの初回発生時に1回。ユーザーが解決する必要がある問題が原因で生じたものである場合は、1日に1回、再表示します。

迷惑度: 通知がそれほど頻繁に表示されない限り、低くなります。

代替方法: 最終的にユーザーが問題を解決する必要がある場合は、問題解決が必須になったときに最終的にモーダルダイアログボックスを表示する段階的なエスカレーションを使用します。

任意選択のユーザー タスク  
ユーザーが実行する必要のある非同期型のタスクを通知します。任意、必須のどちらでも、そのタスクは実行を先に延ばしても安全なものです。



この例では、*Windows Update*により、新しいセキュリティ更新プログラムが利用可能であることが通知されています。

表示のタイミング: タスクの実行の必要性が明らかになった時点で、ユーザーがアクティブであるとき。

表示のタイミング: 1日1回、最多で1日3回。

迷惑度: ユーザーがそのタスクを重要であると考えるか、通知がそれほど頻繁に表示されなければ、低くなります。

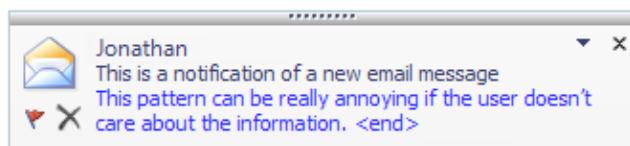
代替方法: 最終的にユーザーがタスクを実行する必要がある場合は、タスクが必須になったときに最終

的にモーダル ダイアログ ボックスを表示する段階的なエスカレーションを使用します。

#### FYI (参考)

有用で関連性のある可能性がある情報を通知します。任意選択でユーザーがオプトインを選択している場合は、かろうじて関連性がある程度の情報でも通知できます。

正しい例:



この例では、新着の電子メール メッセージがあることが通知されています。

正しい例:



この例では、連絡先に登録済みのユーザーがオンラインになったとき、かつユーザーがこの追加情報を受け取ることを選択している場合に通知されます。

間違った例:



この例では、高速 USB ポートが既にインストールされている場合にのみ情報が有用です。そうでない場合は、通知によってユーザーが何らかの操作を行うとは考えられません。

表示のタイミング: トリガーするイベントが発生したとき。

表示方法: リアルタイム オプションを使用し、ユーザーが全画面表示のアプリケーションを実行中の場合、および現在コンピューターを使用していない場合に、この種の通知がキューに登録されないようにします。

表示回数: 1 回

迷惑度: 有用性および関連性の受け止め方によって、中から高になります。ユーザーの関心が低いと考えられる場合は推奨しません。

代替方法: 通知を行いません。

機能の告知  
新しくインストールされた機能、使用されていないシステムまたはアプリケーションの機能を通知します。

機能の告知に通知を使用しないでください。代わりに、以下の別の方法で機能を確認できるようにします。

- 必要とされるコンテキストで、その機能を見つけやすいように設計します。
- 特別なことは何もせず、ユーザーが自分で機能を見つけるようにします。

間違った例:



機能の告知には通知を使用しません。

## ガイドライン

### 全般

- 通知の使用方法に基づいて通知のパターンを選択します。各使用パターンの説明については、上の表を参照してください。
- 最初に Windows を使用するときには通知を使用しないでください。Windows 7 では初回使用時のエクスペリエンスを向上させるために、最初の数時間、通知を表示しないようにしています。ユーザーにはこのような通知が表示されないことを踏まえて、プログラムを設計してください。

## 通知の内容

- 以下の場合を除き、操作の成功については通知しないでください。
  - セキュリティ。ユーザーはセキュリティ操作を最も重要であると考えています。このため、セキュリティ操作に成功したことを通知します。
  - 最近失敗したことがある場合。ユーザーが直前に失敗している場合は、操作の成功が当然であるとは考えていません。このため、少し前に失敗している場合は、成功したことを通知します。
  - 不便をなくすための場合。操作の成功を通知することによってユーザーの不便がなくなると考えられる場合は通知します。したがって、操作にかなり時間がかかる場合、または予想よりも短時間または長時間で完了した場合など、操作が予想外の事態で実行され成功した場合は通知します。
- その他の場合は、成功したことのフィードバックを提供しないか、"オンデマンド" で提供します。ユーザーは操作の成功が当然であると考えていることを前提としてください。操作の実行中に通知領域にアイコンを表示するか既存のアイコンを変化させ、操作が完了したらアイコンを削除するか以前のアイコンに戻すことで、"オン デマンド" のフィードバックを提供できます。
- FYI パターンでは、ユーザーが正常に作業を継続できる場合、または、通知によってユーザーが何らかの操作を行うとは考えられない場合、通知を行いません。

間違った例:



この例では、ポートが既にインストールされている場合にのみ情報が有用です。そうでない場合は、通知によってユーザーが何らかの操作を行うとは考えられません。

- 例外: 任意選択でユーザーがオプトインを選択している場合、関連性が不確かな情報でも通知できます。

正しい例:



この例では、連絡先に登録済みのユーザーがオンラインになったとき、かつユーザーがこの追加情報を受け取ることを選択している場合に通知されます。

- 重大ではないシステムイベントと FYI のパターンについては、1つのイベントに対して1つのまとまった通知を使用します。通知を複数に分けて表示しないでください。

間違った例:





上記の例は、ユーザーが特定の USB キーボードを接続すると Windows XP によって表示される 8 個の通知のうち、4 つを示しています。情報が少しづつ加えられています。

正しい例:



この例では、USB キーボードを接続すると、1 つのまとまった通知が表示されています。

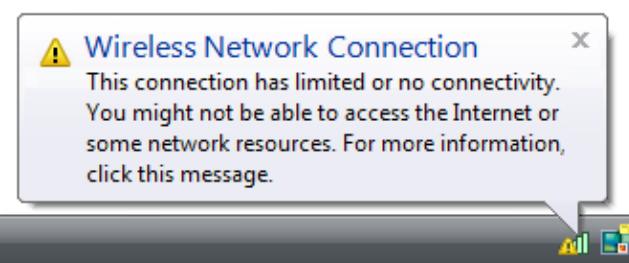
#### 通知のタイミング

- 以下に示すように、設計パターンに基づいて通知を表示します。

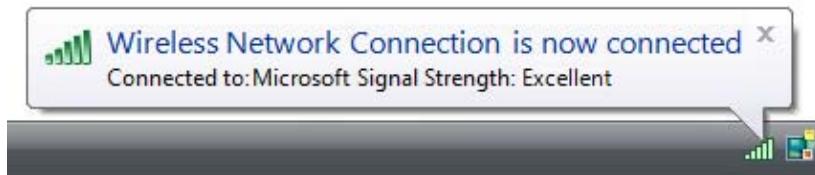
パターン	通知のタイミング
操作の成功	非同期型のタスクが完了した時点。操作の完了を待機している、または最近失敗したことがある場合にのみ、成功を通知します。
操作の失敗	非同期型のタスクが失敗した時点。
重大ではないシステムイベント	イベントが発生した時点でユーザーがアクティブであるとき、または状態が存続する場合。何らかの問題が原因で生じたものである場合、問題が解決されたらすぐに、現在表示されている通知を削除します。
任意選択のユーザー タスク	タスクの実行の必要性が明らかになった時点で、ユーザーがアクティブであるとき。
FYI (参考)	トリガーイベントが発生したとき。

- 操作の失敗パターンについては、その問題が数秒以内に自動的に修正される可能性がある場合、失敗の通知を適切な時間だけ遅らせます。問題が自動的に修正される場合は、何も通知しません。失敗したことがはっきりわかる時間が経過した後に初めて通知してください。通知が早すぎると、ユーザーが報告された問題を認識できない可能性が高く、必要のない通知に注目することになります。

間違った例:



直後に以下の通知が表示されています。



この例では、ワイヤレス接続なしの通知の直後に正常な接続の通知が表示されているので、最初の通知は時期尚早です。

- 操作の成功と FYI のパターンについては、ユーザーが全画面表示のアプリケーションを実行中、または現在コンピューターを使用していない場合は、無効な通知がキューに登録されないようにリアルタイムオプションを使用します。
- 重大ではないシステムイベントのパターンについては、ユーザー ログオンなどの一般的なイベントに短い時間差で関連付けることによって、大量の通知が表示されることがないようにしてください。短い時間差は避け、イベント発生から一定の期間を置いて関連付けます。たとえば、ユーザー ログオンの 5 分後に製品の登録を通知します。

#### 通知の表示時間

Windows Vista 以降では、通知は 9 秒間表示されます。

#### 通知する回数

- 以下に示すように、通知を表示する回数は設計パターンに基づいて決まります。

パターン	通知する回数
操作の成功	1 回。
操作の失敗	1 回。
重大ではないシステムイベント	イベントの初回発生時に 1 回。ユーザーが解決する必要がある問題が原因で生じたものである場合は、1 日に 1 回、再表示します。
任意選択のユーザー タスク	1 日 1 回、最多で 1 日 3 回。
FYI (参考)	1 回。

- 任意選択のユーザー タスクの場合は、通知でユーザーにしつこく要求しないでください。タスクが要求されたら、通知を使用せずに、直ちにモーダルダイアログ ボックスを表示してください。

#### 通知のエスカレーション

- ユーザーが通知を確認することを前提としないでください。以下の場合、ユーザーは通知を確認しません。
  - 作業に集中している場合。
  - 注意を払っていない場合。
  - コンピューターから離れている場合。
  - 全画面表示のアプリケーションを実行している場合。
  - 管理者がユーザーのコンピューターの通知をすべて無効にしている場合。
- ユーザーが最終的に何らかの操作を行う必要がある場合は、[段階的なエスカレーション](#)を使用して、ユーザーが無視できない別の UI を表示します。

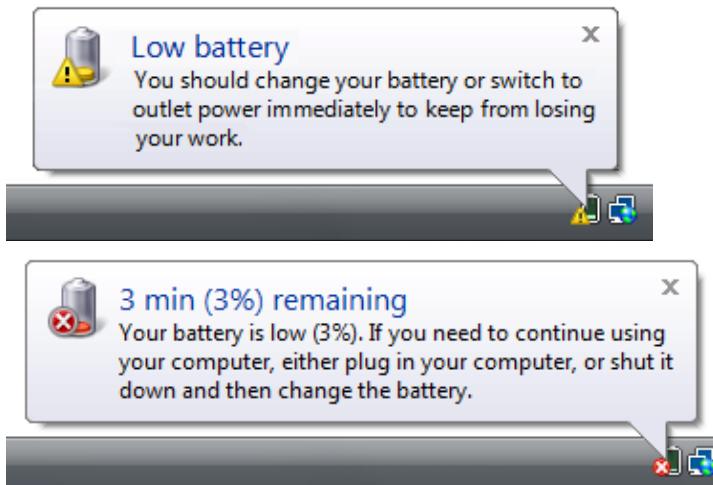
#### 対話操作

- 以下の場合は、通知をクリック可能にします。
  - ユーザーが操作を行う必要がある場合。通知をクリックするとウィンドウが表示され、ユーザーは操作を行うことができます。操作の失敗と任意選択のユーザー タスクの設計パターンでは、この方法が推奨されます。
  - ユーザーが詳細情報を表示する可能性がある場合。通知をクリックするとウィンドウが表示され、ユーザーは追加情報を確認することができます。
- クリックして操作を行う場合は必ずウィンドウを表示します。クリックで操作が直接実行されないようにします。

- 詳細情報を表示するためにクリックした場合は、必ず詳細情報を表示します。既に通知に提供されている情報の単なる言い換えにならないようにしてください。

## アイコン

- 操作の失敗のパターンには、[標準のエラー アイコン](#)を使用します。
- 重大ではないシステムイベントのパターンには、[標準の警告アイコン](#)を使用します。
- その他のパターンには、対象に関連するか、対象を示唆するオブジェクトを示すアイコンを使用します。例として、セキュリティを示すシールドや電源を示すバッテリーのアイコンがあります。
- 対象ユーザーに認識されており、代わりに適切なものが存在しない場合は、アプリケーションまたは会社の[ブランド化](#)に基づくアイコンを使用します。
- [段階的なエスカレーション](#)を用いる場合は、状況の緊急度が増すにつれて、アイコンの外観が段階的に強調されるものを使用するようしてください。



上の例では、緊急度が増すにつれて、アイコンの外観が目立つようになります。

- [標準の情報アイコン](#)は使用しないでください。通知が情報であることは言うまでもありません。
- 以下の場合は、大きいアイコン (32 × 32 ピクセル) の使用を検討してください。
  - アイコンの方がテキストよりも簡単に理解できる場合。
  - 大きいアイコンの方が標準のアイコン (16 × 16 ピクセル) よりも明確かつ効果的に意味を伝えることができる場合。
  - アイコンが [Aero style](#) を使用している場合。



この例では、大きいアイコンを使用する方が、一見しただけで通知の内容を理解できます。

## 通知のキュー登録

注: 別の通知が表示されている場合、ユーザーが全画面表示のアプリケーションを実行している場合、ユーザーがコンピューターを使用していない場合などのように、直ちに通知を表示できない場合は必ず、通知はキューに登録されます。リアルタイムの通知の場合、キューにとどまるのは 60 秒間です。

- 操作の成功と FYI のパターンではリアルタイムオプションを使用し、通知が長時間キューにとどまらないようにします。このような通知は即座に表示されることに意味があります。
- 通知の意味が失われた場合は、キューに登録された通知を削除します。  
開発者向け情報: uFlag に NIF\_INFO フラグを設定し、szInfo を空の文字列に設定することによって、この操作を実行できます。通知がキューに存在しないときにこの操作を行っても悪影響はありません。

## システムの統合

- アプリケーションの実行中にアイコンが[通知領域](#)に常に表示されているわけではない場合、通知を発生させる非

同期型のタスクまたはイベントが実行されている間、一時的にアイコンを表示します。

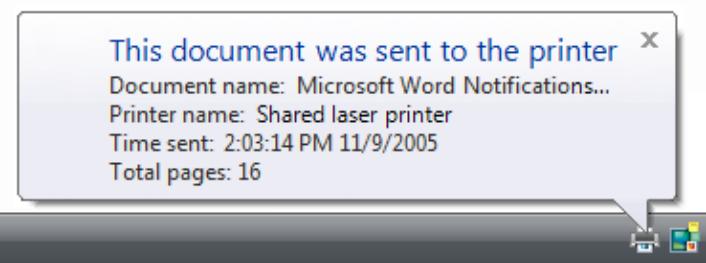
## テキスト

### タイトル テキスト

- ユーザーに伝える必要のある最も重要な情報を、明確でわかりやすい言葉で短く簡潔に要約したタイトル テキストを使用します。ユーザーが最小の努力で通知情報の目的をすばやく理解できるテキストである必要があります。
- 語句または文を使用し、末尾に句読点は付けません。
- センテンス スタイルの大文字化**を使用します。
- ローカライズを想定して、英語の場合の文字数は半角換算で 48 文字以内とします。タイトルの最大文字数は半角 63 文字ですが、英語テキストを翻訳する場合は 30% の拡張を見越しておく必要があります。

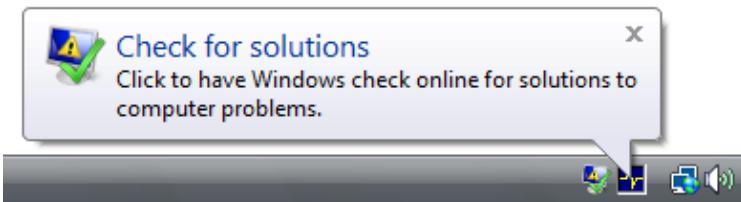
### 本文テキスト

- 本文には説明(タイトルに含まれる情報は繰り返しません)を挿入し、必要に応じて、通知の詳細を提供したり、どのような操作が提供されるかを知らせます。
- 文を使用し、末尾に句点を付けます。
- センテンス スタイルの大文字化を使用します。
- ローカライズを想定して、英語の場合の文字数は半角換算で 200 文字以内とします。本文の最大文字数は半角 255 文字ですが、英語テキストを翻訳する場合は 30% の拡張を見越しておく必要があります。
- 本文には、特定のオブジェクト名などの必須の情報を含めます(例: ユーザー名、ファイル名、URL など)。ユーザーが別のウィンドウを開いてこのような情報を探す必要がないようにします。
- オブジェクト名は二重引用符で囲みます。
  - 例外: 以下の場合は、引用符を使用しないでください。
    - ユーザー名の場合のように、オブジェクト名に常に**タイトルスタイルの大文字化**が使用される場合。
    - オブジェクト名がコロンでオフセットされる場合(例: プリンター名: My printer)。
    - オブジェクト名をコンテキストから容易に見分けることができる場合。
- ローカライズを想定し、オブジェクト名を所定の最大文字数に切り詰める必要がある場合は、省略記号を使用して切り詰めが行われていることを示します。



この例では、省略記号を使用してオブジェクト名を切り詰めています。

- 通知が操作可能なものである場合は、次の語句を使用します。
  - 通知をクリックして操作を行うことができる場合:
    - <重要な情報の簡潔な説明>
    - <オプションの詳細>
    - <操作内容>するには、クリックしてください。



この例では、ユーザーはクリックして操作を実行できます。

- 通知をクリックして詳細情報を表示できる場合:
  - <重要な情報の簡潔な説明>

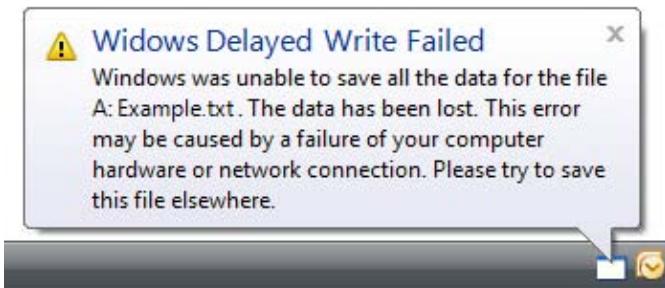
<オプションの詳細>  
詳細を表示するには、クリックしてください。



この例では、ユーザーはクリックして詳細を表示できます。

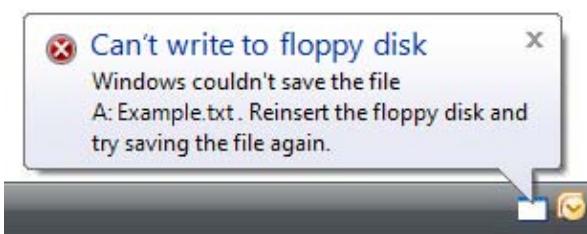
- 通知では、ユーザーがある操作を実行 "しなければならない" という表現は使用しません。通知は、ユーザーが無視することもできる、重大ではない情報のためのものです。ユーザーの操作が本当に必要な場合は、通知を使用しないでください。
- ユーザーが操作を行う必要がある場合は、重要な点を明確にします。
- 操作の失敗と重大ではないシステムイベントのパターンについては、わかりやすい言葉で問題を説明します。

間違った例:



この例では、問題の説明にあまりにも専門的でわかりにくい言葉が使用されています。

正しい例:



この例では、わかりやすい言葉で問題が説明されています。

- 対象ユーザーに関連を持たせる方法でイベントを説明します。通知によってユーザーがタスクを実行したり、対応を変える可能性が十分にある場合に、通知に関連性があることになります。多くの場合、技術的な問題を説明するのではなく、ユーザーの目的という観点から通知を説明することにより、このことを実現できます。

## ドキュメント

通知に言及するときは、以下のことに留意します。

- 大文字と小文字の区別を含め、タイトルテキストを正確に引用します。
- "バルーン" や "警告"、"アラート" ではなく、"通知" としてコンポーネントを示します。
- ユーザー操作を説明する場合は、"クリック" を使用します。
- タイトルテキストを二重引用符 (" ") で囲み、可能な場合は太字にします。

例: "重要な更新をインストールする準備ができました" 通知が表示されたら、通知をクリックして処理を開始します。

通知領域に言及するときは、以下のことに留意します。

- 通知領域は "通知領域" と示し、"システムトレイ" とはしないでください。

# 通知のデザイン コンセプト

## 通知

ユーザー エクスペリエンスを向上させる効果的な通知を以下に示します。

- 非同期型である。ユーザーの Microsoft® Windows® やアプリケーションの対話操作に対して、通知が即座に行われない。
- 有益である。通知の結果としてユーザーがタスクを実行したり、動作を変更するのに値する有益性がある。
- 関連性が高い。ユーザーが関心を持つ、未知の有益な情報が表示されている。
- 緊急性が低い。モーダルではなく、ユーザーの操作を必要としないため、自由に無視できる。
- 実行性が高い。操作の実行を求める通知の場合、通知をクリックして操作を開始できる。ただし、操作の開始はいつでも延期できるようになっている。
- 適切に表示されている。通知の表示(期間、頻度、テキスト、アイコン、インタラクティビティ)が状況に応じたものである。
- 不快ではない。イベントの通知が控え目になるか、しつこくなるかは紙一重であるので、この点に配慮されたものである。

残念ながら、世の中には不快、不適切、無駄、無関係な通知が数多く存在します。Windows XP の不名誉の殿堂に入っている次のような通知を確認してください。



これらの例では、Windows XP は表向きには初期構成に従ってユーザーを支援しようとしています。しかし、これらの通知はポップアップの頻度が多すぎて、役に立つどころかおせっかいな機能の宣伝のようになっています。

## ユーザーのフローを維持する

理想的には、作業に没頭するユーザーが通知をまったく確認する必要がなく、フローが中断されたときにだけ通知を確認できるようにします。

『Flow: The Psychology of Optimal Experience』の著者 Mihaly Csikszentmihalyi 氏は、ユーザーが完全に活動に没頭し、時間の感覚を失って大いなる満足感に浸っているとき、ユーザーはフロー状態であるとしています。

効果的な通知は、有益で関連性の高い情報を表示する一方で通知自体を無視でき、ユーザーのフロー維

持に貢献します。この場合、通知は控え目に表示され、ユーザーの操作を必要としません。

通知が**モードレス**であったとしても、不快な中断になることがあります。ユーザーの注意を求めてはいませんが、必要としていることは確かです。ユーザーのフローを中断する要因は以下のとおりです。

- ユーザーの関心が低い通知を表示する。
- 通知を表示する頻度が高い。
- 1回で十分な通知を何回も表示する。
- 通知を表示するときに音を出す。

Windows 7 では、ユーザーが通知を自由に制御できます。プログラムの通知があまりにも不快だと感じた場合は、プログラムからのすべての通知を非表示にできます。有益で関連性の高い情報を表示し、このガイドラインに従っていれば、ユーザーがプログラムの通知を非表示にすることはありません。

## 通知を無視できる

ユーザーがすぐに操作する必要がなく、自由に通知を無視できるようにします。

多くの場合、開発者や設計者は、"自分たちが必要と考える" 通知をユーザーが無視できないかたちで表示することを考えます。これによりユーザーのフローが中断されるため、通知の利点そのものが完全に損なわれます。ユーザーが通知によって気が散ったり、通知を読むことを義務と感じたりするなら、通知の設計は失敗です。

ユーザーに通知を無視されることが不安な場合は、次の点について検討してください。

- 通知を正しく使用しており、ユーザーがすぐに操作する必要がない場合、ユーザーが通知を無視できる設計にします。このルールは守ってください。
- ユーザーがすぐに操作する必要がある通知の場合は、ユーザーが無視できない別のユーザー インターフェイス (UI) を使用します。別の UI については、「[適切なユーザー インターフェイスかどうかの判断基準](#)」を参照してください。

## 適用できる部分には段階的なエスカレーションを使用する

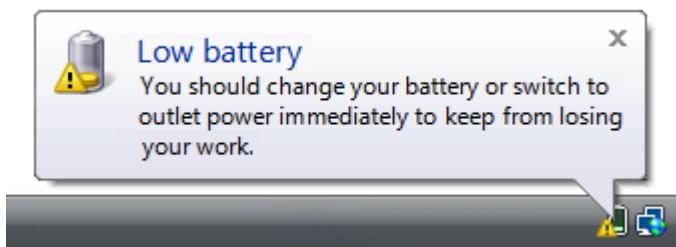
最初は無視しても安全だが、最終的には対処する必要があるイベントで通知を使用する場合は、状況が深刻になったときに別の UI を使用する必要があります。この技術を "段階的なエスカレーション" といいます。

たとえば、Windows 電源管理システムでは、最初は通知領域アイコンを変更するだけでバッテリ低下を示します。



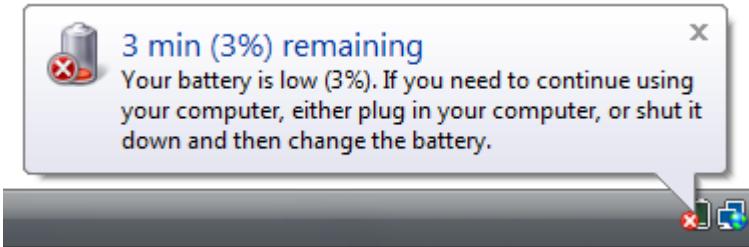
この例では、Windows 電源管理の通知領域アイコンによってバッテリ低下を段階的に通知されています。

バッテリが低下するにつれ、通知によってユーザーにバッテリ低下が警告されます。



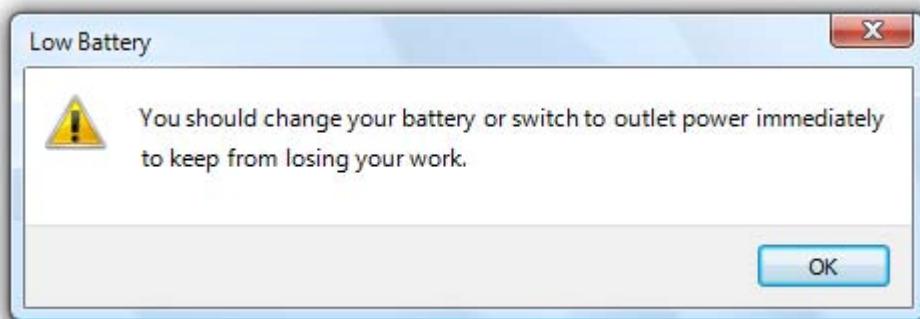
この例では、Windows 電源管理の通知によってユーザーにバッテリ低下を知らせています。

この通知が表示されている間、ユーザーには、電源に接続する、電源オプションを変更する、作業を中止してコンピューターをシャットダウンする、通知を無視して作業を継続する、などの複数の選択肢が残されています。バッテリ低下が続くと、通知のテキストやアイコンに新たな警告が表示されます。



この例では、テキストとアイコンで新たな警告を知らせています。

ただし、バッテリがかなり低下したためユーザーがすぐに操作する必要がある場合は、Windows 電源管理のモーダル メッセージ ボックスでユーザーに通知します。



この例では、Windows 電源管理のモーダル メッセージ ボックスで緊急を要するバッテリ低下を知らせています。

### 3つの重要な点

1. 通知は、本当に必要な場合にのみ使用します。通知を表示すると、ユーザーの作業を中断したり、ユーザーに不快感を与える可能性があります。正当な理由で中断が行われるようにします。
2. 通知は、ユーザーがすばやく対応する必要がない、重大ではないイベントまたは状況に使用します。ユーザーがすばやく対応する必要のある重大なイベントまたは状況の場合は、他の UI (モーダル ダイアログ ボックスなど) を使用します。
- 3.. 通知を使用する場合は、快適なユーザー エクスペリエンスをもたらすようにします。通知の確認は強制しないでください。ユーザーが作業に没頭しているときに通知に気付かなければ、設計は成功です。

## 対話操作

以下の記事では、Windows® ベースのアプリケーションにおけるユーザーの対話操作のガイドラインについて説明しています。

- キーボード
- マウスとポインター
- タッチ
- ペン
- アクセシビリティ

## キーボード

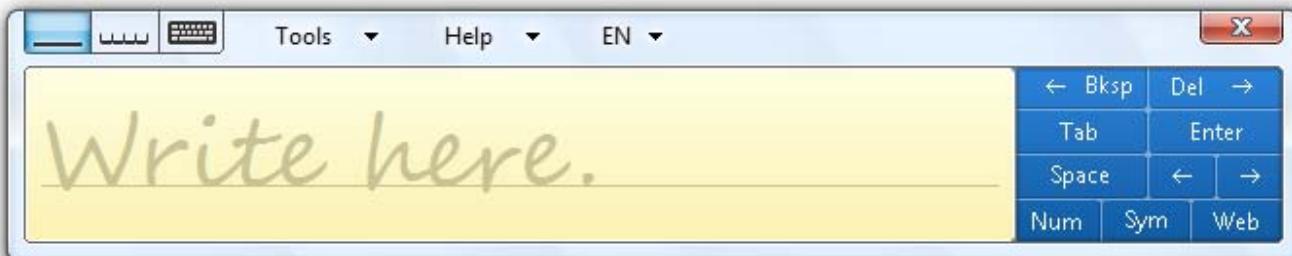
- [デザインコンセプト](#)
- [ガイドライン](#)
- [対話操作](#)
- [キーボードナビゲーション](#)
- [アクセスキー](#)
- [メニューアクセスキー](#)
- [ダイアログボックスアクセスキー](#)
- [ショートカットキー](#)
- [ショートカットキーの選択](#)
- [ショートカットキーの選択\(禁止事項\)](#)
- [キーボードとマウスの併用](#)
- [ドキュメント](#)
- [Windows のキーボードショートカットキー](#)

"キーボード" は、Microsoft® Windows® のテキスト入力に使用する主な入力デバイスです。アクセシビリティと効率化のため、ほとんどの操作をキーボードでも実行することができます。

キーボードという語は、タブレットベースのコンピューターなど、物理的なキーボードのないコンピューターで使用される、仮想的なスクリーンキーボードや手書きパッドを指すこともあります。



Windows Tablet とタッチ テクノロジーのスクリーンキーボード

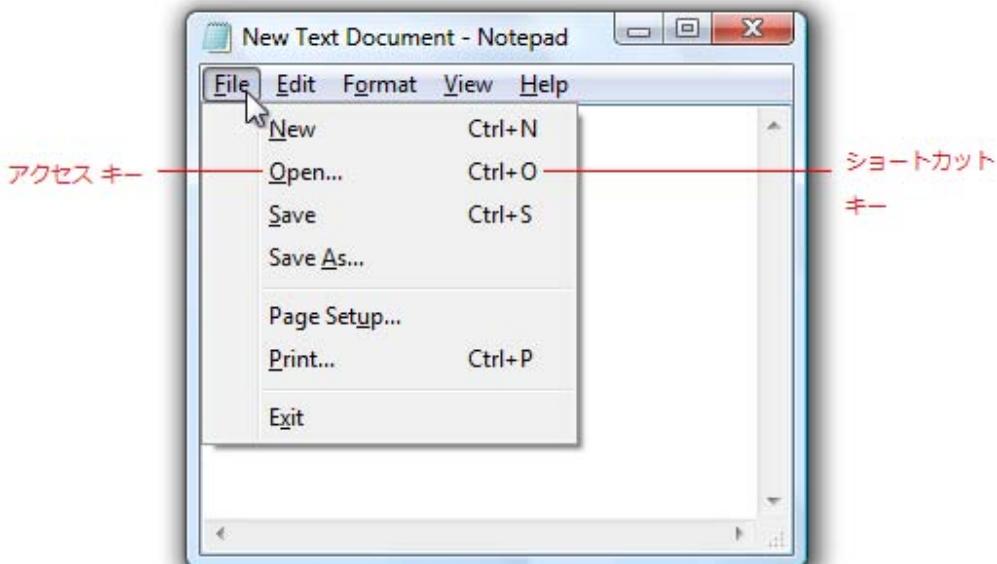


Windows Tablet とタッチ テクノロジーの手書きパッド

次に示すとおり、基本となるキーが 6 種類あります。

- "文字キー" は、表示どおりの文字を入力フォーカスのあるウィンドウに送ります。
- "修飾キー" (Ctrl キー、Alt キー、Shift キー、Windows ロゴキーなど) は別のキーと組み合わせて使用し、そのキーの意味を変えます。
- "ナビゲーションキー" キーは、方向キー、Home キー、End キー、PageUp キー、PageDown キーです。
- "編集キー" は、Ins キー、BackSpace キー、Del キーです。
- "ファンクションキー" は、F1 キーから F12 までのキーです。
- "システムキー" は PrintScreen キー、CapsLock キー、NumLock キーなど、システムを特定のモードに設定する、またはシステムタスクを実行するものです。

"アクセスキー" は、キーボードを使用してすべてのコントロールまたはメニュー項目を操作できるように用意された、アクセシビリティのためのキーまたはキーの組み合わせです。"ショートカットキー" は、使用頻度の高いコマンドを効率よく実行できるように用意された、上級ユーザー向けのキーまたはキーの組み合わせです。Windows では、アクセスキー割り当てに下線を付与してアクセスキーを示しています。



この例には、アクセスキーとショートカットキーの両方が示されています。

Windowsでは見た目が煩雑にならないように、既定でアクセスキーの下線が非表示にされており、Altキーが押されるとはじめて表示されます。Windowsとの一貫性を保つために、UXガイド内の画像も、その内容がアクセスキーに関連する場合を除き、アクセスキーの下線を非表示にしています。

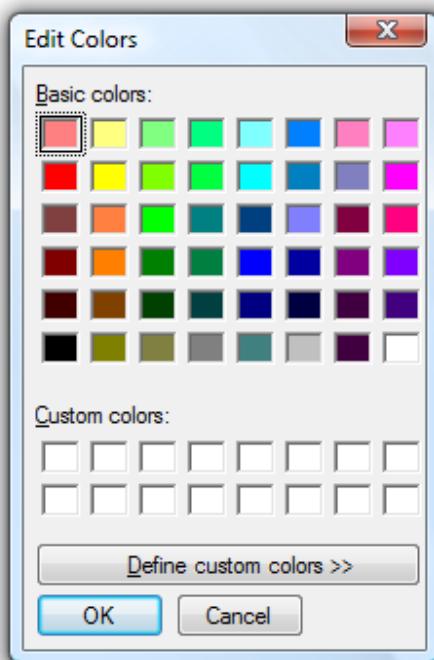
開発プロセス全体でプログラム内のアクセスキー割り当てに対する意識を高めるために、常に表示するように設定することができます。コントロールパネルからコンピューターの簡単操作センターを開き、[キーボードを使いやすくします]をクリックし、[ショートカットキーとアクセスキーに下線を表示します]チェックボックスをオンにします。

**注:** アクセシビリティに関するガイドラインは、別の項目として記載しています。

## デザイン コンセプト

### キーボード ナビゲーションの要素

ユーザーは、キーボードを使用してコントロールに移動し、選択を行い、コマンドを実行することによってウィンドウと対話します。この対話を実現するため、以下の要素も同時に機能しています。



以下の一覧でキーボードナビゲーションの要素を説明するにあたって、このダイアログボックスを使

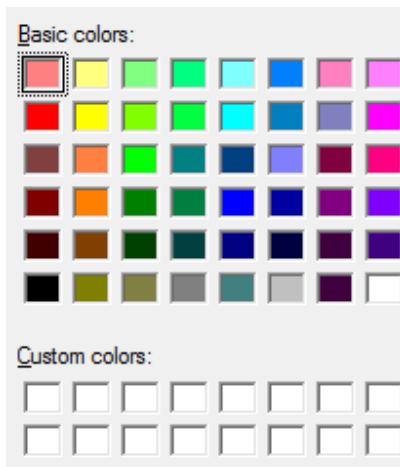
用します。

- 入力フォーカス。入力フォーカスのあるコントロールは、ほとんどのキーボード入力を受け取ります。入力フォーカスは、フォーカス長方形と呼ばれる点線の四角形で示されます。キーボード入力には後で説明するように、入力フォーカスのないコントロールに送られるものもあります。



最初の [基本色] コントロールに、点線の四角形からわかるように入力フォーカスがあります。

- Tab キーとタブストップ。Tab キーはウィンドウ内のナビゲーションにおける重要な機構です。Tab キーはタブストップが設定されているコントロールにのみ停止します。すべての対話型コントロール (グループ内にあるコントロールを除く) にタブストップを設定する必要がありますが、ラベルのような非対話型のコントロールには設定しません。
- タブオーダー。タブオーダーに従って、タブストップが設定されたすべてのコントロールに停止します。Tab キーを押すと、入力フォーカスがタブオーダーの次のコントロールに移動し、Shift キーを押しながら Tab キーを押すと、入力フォーカスが前のコントロールに移ります。
- コントロール グループ。関連のあるいくつかのコントロールをグループにまとめ、タブストップを 1 つ割り当てるすることができます。コントロール グループはラジオ ボタンなど、単一のコントロールのように動作するコントロールのセットに使用します。コントロールの数が多すぎて Tab キー単独では効率よく移動できない場合にも使用できます。



[基本色] および [作成した色] はコントロール グループであり、このダイアログ ボックスには 5 つのタブストップが設定されています。多数のコントロールが存在するので、コントロール グループを使用しないと効率よく移動できません。

- 方向キー。方向キーは、グループ内のコントロール間で入力フォーカスを移動させます。→ キーを押すと、入力フォーカスがタブオーダーの次のコントロールに移動し、← キーを押すと、入力フォーカスが前のコントロールに移ります。Home キー、End キー、上方向キー、下方向キーにもそれぞれ、グループ内で期待される動きが割り当てられています。方向キーではコントロール グループの外部に移動することはできません。
- 既定のボタン。コマンド ボタンおよびコマンド リンクが含まれるウィンドウには、強調表示された枠線で示された既定のボタンが 1 つあり、Enter キーが押されたときにこのボタンがクリックされます。既定では、既定のコマンド ボタンまたはコマンド リンクが 1 つ割り当てられています。ただし、既定のボタンは、ユーザーが別のコマンド ボタンまたはコマンド リンクに移動するのに伴って動きます。このため、入力フォーカスのあるコマンド ボタンまたはコマンド リンクが常に既定のボタンになります。



[OK] ボタンは、通常、強調表示された枠線で示されているとおり、既定のボタンです。ただし、ユーザーが [キャンセル] ボタンにフォーカスを移動するとそれが既定のボタンになり、Enter キーによってアクティビ化されます。

- Space キー、Enter キー、Esc キー。Space キーは入力フォーカスのあるコントロールをアクティビ化するのに対し、Enter キーは既定のボタンをアクティビ化します。Esc キーを押すと、ウィンドウが取り消されるか、閉じられます。

- ・ アクセスキー。アクセスキーは、**Tab**キーで移動せずに直接コントロールを操作する際に使用します。アクセスキーは**Alt**キーと共に使用するものであり、ラベル内の下線付き文字によって示されています。
- ・ アクセスキーのラベル。コマンドボタン、チェックボックス、ラジオボタンのようにそれぞれ独自のラベルを持つコントロールもありますが、リストボックスやツリービューのように外部ラベルを持つコントロールもあります。外部ラベルの場合、アクセスキーはラベルに割り当てられているので、呼び出しの際もタブオーダーの次のコントロールに移動します。**[OK]**、**[キャンセル]**、**[閉じる]**のラベルの付いたボタンは**Enter**キーまたは**Esc**キーによって実行されるので、アクセスキーを割り当てません。



**Alt**キーを押しながら**B**キーを押すと、選択されている基本色に移動し、**Alt**キーを押しながら**D**キーを押すと、[色の作成]ボタンがクリックされます。**Enter**キーによって**[OK]**ボタンが実行され、**Esc**キーによって[bキャンセル]ボタンが実行されます。

- ・ アクセスキーの動作。アクセスキーを呼び出したとき、それが一意に割り当てられたものである場合は、関連付けられたコントロールがクリックされます。割り当てが一意ではない場合は、関連付けられたコントロールに入力フォーカスが与えられます。再びそのアクセスキーを入力すると、タブオーダーに従って同じ割り当てを持つ次のコントロールに入力フォーカスが与えられます。

この機構はかなり複雑ですが、同時にきわめて直感的です。ユーザーはこのような細かな動きを正確に説明することはできなくても、その大部分をすぐに把握することができます。

#### アクセシビリティと上級ユーザーのためのキーボードサポート

Windowsでのキーボードデザインとは、優れたデザインのキーボードナビゲーション、アクセシビリティのためのアクセスキー、上級ユーザーのためのショートカットキーに集約されます。

障害のある方も含め、幅広いユーザーがプログラムの機能を簡単に利用できるようにするには、すべての対話型ユーザーインターフェイス(UI)要素にキーボードでアクセスできることが必要です。一般的に、最も使用頻度の高いUI要素には単一のアクセスキーまたはキーの組み合わせを使用してアクセスし、使用頻度がそれほど高くない要素にはこれらのキーに加えて**Tab**キーまたは方向キーを使用して移動できるようにします。障害のあるユーザーには、包括性の方が一貫性よりも重要です。

プログラムの機能を上級ユーザーにとって効率よくするには、使用頻度の高いUI要素にはショートカットキーも割り当て、直接キーボードでアクセスできるようにする必要があります。上級ユーザーは、多くの場合、キーボードベースのコマンドの方がすばやく入力でき、キーボードから手を離す必要がないので、キーボードの使用を好みます。こうしたユーザーには、効率と一貫性が重要であり、包括性は最も使用頻度の高いコマンドにのみ重要です。

この2つのグループに対しては、キーボードアクセスのデザインに微妙な違いが存在します。Windowsで直接的なキーボードアクセス機構を2つに分けて提供しているのはこのためです。アクセスキーとショートカットキーを効果的に使用することによって、効率性、一貫性、包括性のあるキーボードアクセスをプログラムで使用できるようになり、あらゆるユーザーにメリットをもたらすことができます。

#### アクセスキー

アクセスキーには次の特徴があります。

- ・ **Alt**キーを押しながら英数字キーを押す。
- ・ 主としてアクセシビリティを目的としている。
- ・ すべてのメニューと大部分のダイアログボックスコントロールに割り当てられている。
- ・ 記憶して使用するものではなく、UIに直接示されている(コントロールラベルの該当文字に下線が付けられています)。
- ・ 現在のウィンドウでのみ効果があり、対応するメニュー項目またはコントロールに移動する。
- ・ 一貫性を持たせて割り当てられていない(常に存在するわけではないため)。ただし、**使用頻度の高いコマンド**(特にコミットボタン)には一貫性を持たせてアクセスキーを割り当てる必要があります。
- ・ ローカライズされる。

アクセスキーは記憶して使用するものではないため、ラベルの後半にキーワードがある場合でも、ラベルの前の方に出現する文字に割り当て、見つけやすいようにします。

正しい例:

- [Don't use automatic learning, and delete any previously collected data](#)

間違った例:

- [Don't use automatic learning, and delete any previously collected data](#)

正しい例では、アクセスキーがラベルの前の方の文字に割り当てられています。

ショートカットキー

一方、ショートカットキーには次の特徴があります。

- 主として **ctrl** キーとファンクションキーを組み合わせて使用する (Windows システムのショートカットキーでも、**Alt** キーと英数字以外のキー組み合わせおよび Windows ロゴキーが使用されます)。
- 主として上級ユーザーの効率を上げることを目的としている。
- 最も使用頻度の高いコマンドにのみ割り当てられている。
- 記憶して使用するものであり、メニュー、ツールヒント、ヘルプ内にのみ記されている。
- プログラム全体で効果があるが、該当しない場合は効果がない。
- 記憶して使用され、直接示されないので、一貫して割り当てる必要がある。
- ローカライズされない。

ショートカットキーは記憶して使用するものであるため、最も使用頻度の高いショートカットキーには先頭の文字またはコマンドキーワード内の最も記憶しやすい文字を使用するのが理想的です。たとえば、コピー (Copy) には **Ctrl + C** を、要求 (Request) には **Ctrl + Q** を割り当てます。

よく知られているショートカットキーの意味に一貫性がないと、ストレスやエラーの原因になります。

間違った例:

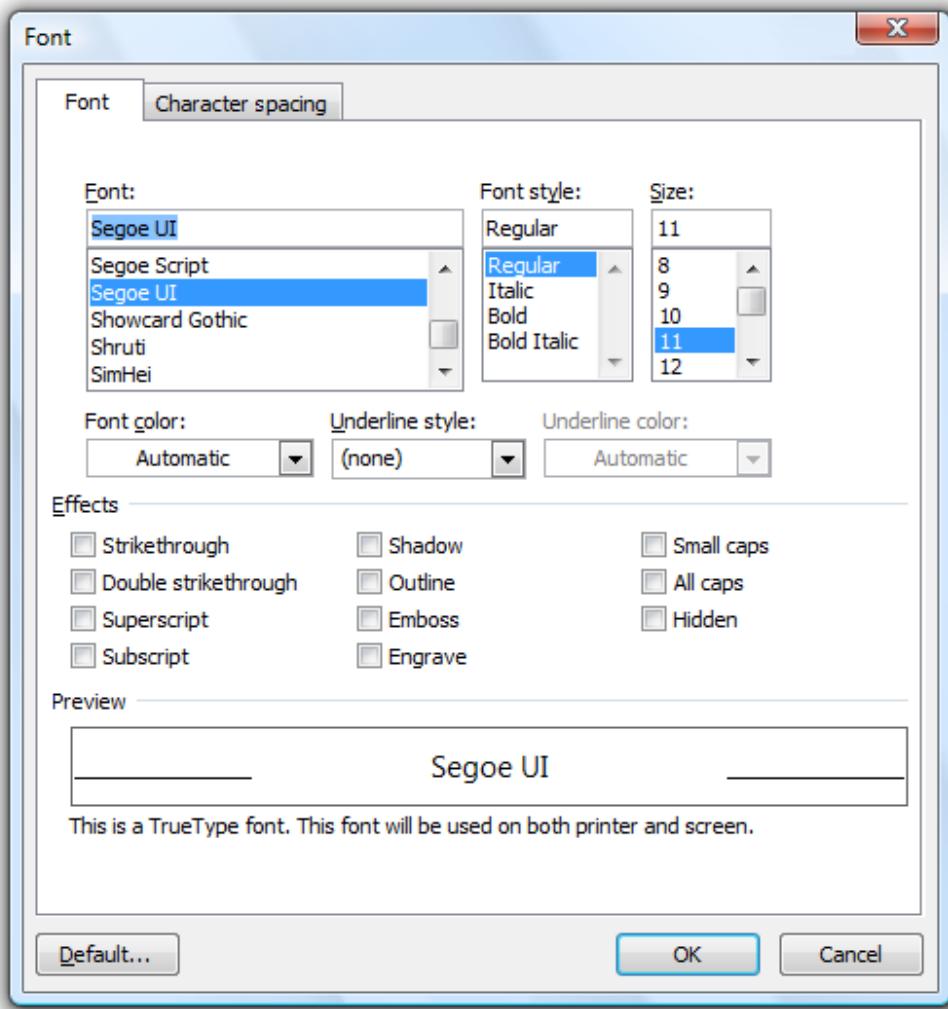


この例では、**Ctrl + F** は [検索](Find) の標準のショートカットキーであるため、[転送](Forward) に割り当てるなど、ストレスやエラーの原因になります。**Ctrl + W** を割り当てる方が適切で記憶しやすくなります。

ショートカットキーは記憶して使用するものであるため、アプリケーションに特有のショートカットキーが意味を持つのは、使用するユーザーが記憶できるほど頻繁に実行されるプログラムと機能に限られます。使用頻度の低いプログラムと機能に、ショートカットキーは必要ありません。たとえば、セットアッププログラムや大半のウィザード、生産性アプリケーション内の使用頻度の低いコマンドには、特別なショートカットキー割り当ては必要ありません。

ダイアログボックス内でのアクセスキーの割り当て

通常はアクセスキーを割り当てないものを除き、できる限りすべての対話型コントロールに一意のアクセスキーを割り当てます。ただし、英語には 26 文字しかないように、どのラベルにも出現しない文字があったり、すべてのラベルに特徴的な文字が存在するとは限らないので、この文字数はさらに少なくなります。また、ローカライズを容易にするために、いくつかの文字を未割り当てにすることを考える必要があります。その結果、1つのダイアログボックスで割り当て可能な一意のアクセスキーはわずか 20 個前後になります。



この例では、コントロールの数が多すぎるため、アクセスキーを一意に割り当てることができません。

対話型コントロールが 20 個以上含まれるダイアログ ボックスがある場合、一部のコントロールにアクセスキーを割り当てないか、まれなケースとしては重複するアクセスキーを割り当てます。

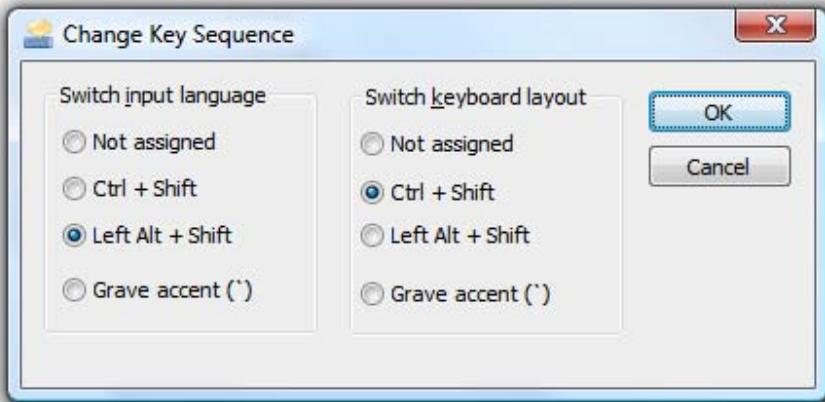
アクセスキーの割り当てでは、以下の一般的な手順を使用します。

- 最初に、アクセスキーを [コミット ボタン](#)とコマンドリンクに割り当てます。該当する場合は[標準のアクセスキー割り当て表](#)を使用します。または、最初の文字を使用します。
- アクセスキーを割り当てないコントロールは除外します。
- 以下のようにして、残りのコントロールに一意のアクセスキーを割り当てます(最も使用頻度の高いものから開始します)。
  - 可能な場合は、標準のアクセスキー割り当て表に従ってアクセスキーを割り当てます。
  - または:
    - できるだけラベルの前の方に出現する文字を使用します。1番目または2番目の語の先頭文字を使用するのが理想的です。
    - できるだけ特徴のある子音または母音を使用します。たとえば、"Exit"(終了)の"x"などを使用します。
    - できるだけ幅の広い文字を使用します。たとえば、w、m、大文字などを使用します。
    - 下線が見えにくくなる文字は使用しないようにします。たとえば、1ピクセルしか幅のない文字、ディセンダーのある文字、ディセンダーのある文字の隣にある文字は避けます。
- すべてのコントロールに一意のアクセスキーを割り当てることができない場合は、以下のようにします(最も使用頻度の低いものから始めます)。
  - 次のような関連のあるコントロールのグループが存在する場合
    - ラジオ ボタンの集合
    - 関連のあるチェック ボックスの集合

- グループ ボックス内の関連のあるコントロールの集合

アクセスキーを、個々のコントロールではなくグループのラベルに割り当てます。通常はこの逆です。(そのためには、このようなコントロールに、定義済みの[コントロール グループ](#)が存在しているようにします。)

- それでも、すべてのコントロールに一意のアクセスキーが割り当てられていない場合は、以下のようにします。
  - 以下の場合は、一意ではないアクセスキーを割り当てることができます。
  - 他の方法ではコントロールにアクセスすることが困難である。
  - 一意ではないアクセスキーが、使用頻度の高いコントロールのアクセスキーと重複していない。
  - または、**Tab**キーと方向キーを使用して残りのコントロールにアクセスすることができます。



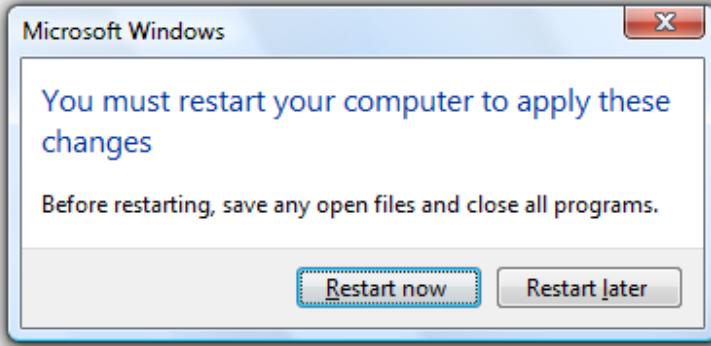
この例では、同じコントロールが2回出現するため、ラジオボタンのグループにアクセスキーが割り当てられています。

#### 意図しないコマンド実行の防止

コンテキストの外部に表示されたウィンドウ(ユーザー開始ではないもの)に入力フォーカスが移動した場合、このウィンドウが、他のウィンドウに対する入力を受け取る可能性があります。さらに、そのダイアログ ボックスにテキスト入力を受け付けるコントロール(テキスト ボックス、リストなど)が存在しない場合、**Alt**キーを押さなくてもアクセスキーが機能します。次の例では、「r」を押すと[今すぐ再起動]ボタンがアクティブになります。

このような入力が、意図しない重大な結果をもたらす可能性があることは明らかです。

間違った例:

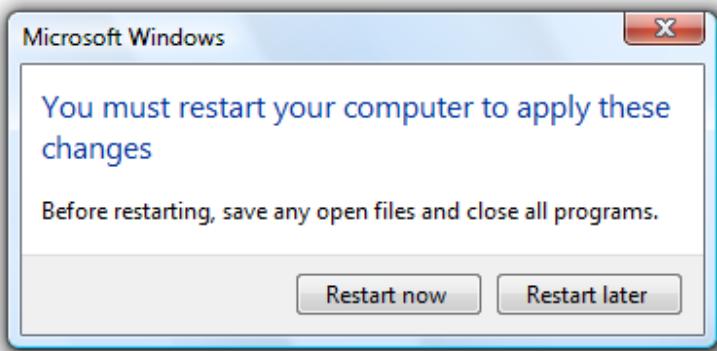


この例では、スペースまたは「r」を入力するか**Enter**キーを押すと、意図せずにWindowsが再起動されます。

当然ながら、この問題に最適の解決策は、入力フォーカスを移動しないことです。その代わりに、プログラムの[タスクバー](#)ボタンを点滅させるか、ユーザーの注意を喚起する通知を表示します。

ただし、このようなウィンドウの表示が必要な場合は、既定のボタンまたはアクセスキーを割り当てないで、最初の入力フォーカスをコミットボタン以外のコントロールに与えるのが最適です。

正しい例:



この例では、意図しない Windows の再起動が起こりにくくなっています。

## 6つの重要な点

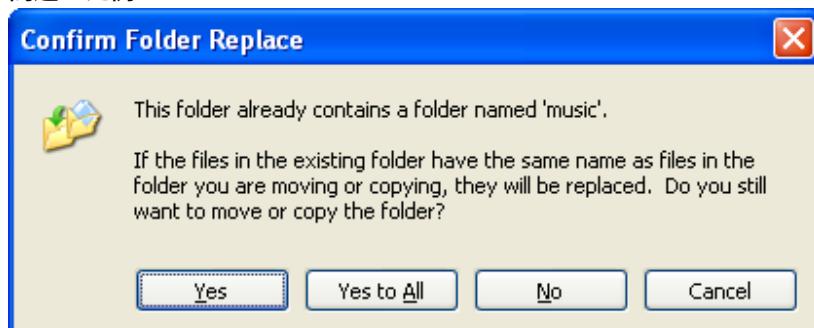
- 適切なタブ オーダーとコントロール グループ、最初の入力フォーカス、および既定のボタンを設定して、適切なキーボード ナビゲーションをデザインします。
- アクセスキーはすべてのメニューと大部分のコントロールに割り当てます。
- アクセスキーは、ラベルの前の方に出現する文字に割り当て、見つけやすくします。
- ショートカットキーは、最も使用頻度の高いコマンドに割り当てます。
- ショートカットキーは、先頭文字またはキーワード内の最も印象的な文字に割り当てるようにします。
- よく知られているショートカットキーには、一貫した意味を割り当てます。

## ガイドライン

### 対話操作

- Shift キーを使用して、メニューまたはダイアログ ボックス内のコマンドを変更することはしません。これは発見も予測もできない用法です。

#### 間違った例:



Windows XP のこの例では、Shift キーを押すことによって、[すべてはい] が [すべていいえ] に変更されます。

- 入力フォーカスのあるコントロールを無効にしません。この場合、ウィンドウがキーボード入力を受け付けなくなる可能性があります。入力フォーカスのあるコントロールを無効にする前に、入力フォーカスを別のコントロールに移します。
- ウィンドウがコンテキストの外部に表示され、ユーザーを驚かせる可能性がある場合は、以下の事項を実践し、意図されない重大な結果が引き起こされないようにする必要があります。
  - 既定のボタンは割り当てません。
  - アクセスキーは割り当てません。
  - 最初の入力フォーカスは、コミット ボタン以外のコントロールに設定します。

### キーボード ナビゲーション

- 常に、入力フォーカス インジケーターを表示します。例外: 以下の場合は、一時的に入力フォーカス インジケーターを非表示にすることができます。
  - 入力フォーカス インジケーターがあると目障りになる(詳細ビュー内ではなく、大きなリスト ビューなど)。

Enter キーの使用前に、Alt キーや方向キーのような他のキーボード入力が想定される。

- 入力フォーカス インジケーターがキーボード入力と一緒に表示される。

- ユーザーが最初に操作する可能性の高いコントロールに最初の入力フォーカスを割り当てます。多くの場合、最初の対話型コントロールに割り当てます。最初の対話型コントロールが適していない場合、ウィンドウのレイアウトを変更することを検討します。
- 読み取り専用の編集ボックスを含む、すべての対話型コントロールにタブストップを割り当てます。次の場合は例外です。
  - 1つのコントロールとして動作する、関連するコントロールのグループ(ラジオ ボタンなど)。このようなグループには1つのタブストップが割り当てられます。
  - 方向キーでグループ内を前後に移動したときに、フォーカスがグループ内から外に出ないように、グループを適切に設定します。
- タブオーダーは左から右、上から下の順序にします。一般的に、タブオーダーは読む順序と同じにします。よく使用的なコントロールは、例外的にタブオーダーの早い方に設定することを検討します。Tab キーでは、止まることなく両方向に、すべてのタブストップを循環するようにします。グループ内では、タブオーダーに例外を設けず、順番どおりにします。
- タブストップ内では、方向キーは、左から右、上から下へ移動するようにします。例外は設けません。方向キーでは、止まることなく両方向に、すべてのアイテムを循環するようにします。
- 次の順番でコミット ボタンを配置します。
  - [OK]/"実行する"/[はい]
  - "実行しない"/[いいえ]
  - [キャンセル]
  - [適用] (該当する場合)

"実行する" および "実行しない" には、メイン指示テキストに対する具体的な応答が入ります。

- 最も安全(データの消失やシステムアクセスが失われることを防ぐため)で、かつ最もセキュリティの高いコマンドボタンまたはコマンドリンクが既定になるように選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用される応答または最も便利な応答を選択します。
- キーボードナビゲーションによってコントロールの値が変更されたり、エラーメッセージが表示されないようにします。ナビゲーション中にコントロールの初期値を変更するようにユーザーに要求しないようにします。その代わりに、有効な値かどうかを終了時に検証するようにコントロールを初期設定し、コントロールの値が変更された場合にのみ検証します。

## アクセスキー

- できる限り次の表に従って、よく使用されるコマンドにアクセスキーを割り当てます。アクセスキーを常に同じように割り当てるることはできません。ただし、特に、頻繁に使用されるコマンドには一貫性を持たせることをお勧めします。

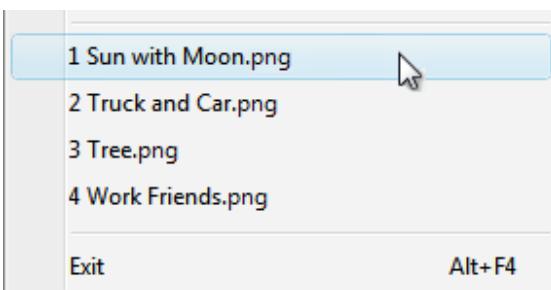
バージョン情報(A)	ファイル(E)	次へ(N)	再開(R)
常に手前に表示(A)	検索(E)	いいえ(N)	再試行(R)
適用(A)	次を検索(N)	開く(O)	実行(R)
戻る(B)	フォント(E)	プログラムから開く(W)	上書き保存(S)
太字(B)	進む(E)	オプション(O)	名前を付けて保存(A)
参照(B) または参照(R)	ヘルプ(H)	ページ設定(U)	すべて選択(A)
閉じる(C)	ヘルプトピック(I)	貼り付け(P)	送る(N)
コピー(C)	非表示(H)	リンク貼り付け(L)	表示(S)
ここにコピー(C)	挿入(I)	ショートカットの貼り付け(S)	サイズ(S)
ショートカットの作成(S)	オブジェクトの挿入(O)	形式を選択して貼り付け(S)	分割(P)
ここにショートカットを作成(S)	斜体(I)	一時停止(P)	停止(S)
切り取り(I)	ここにリンク(L)	再生(P)	ツール(I)

削除(D)	最大化(X)	印刷(P)	下線(U)
今後、この<アイテム>を表示しない(D)	最小化(N)	ここに印刷(P)	元に戻す(U)
編集(E)	詳細(M)	プロパティ(R)	表示(V)
終了(X)	移動(M)	やり直し(R)	ウィンドウ(W)
探索(E)	ここに移動(M)	繰り返し(R)	はい(Y)
簡易表示(E)	新規作成(N)	復元(R)	

- できるだけ幅の広い文字を使用します。たとえば、w、m、大文字などを使用します。
- できるだけ特徴のある子音または母音を使用します。たとえば、"Exit"(終了)の"x"などを使用します。
- 下線が見えにくくなる文字を使用しないようにします。以下に、問題点が大きいものから順に示します。
  - 1ピクセルしか幅のない文字(i、lなど)。
  - ディセンダーのある文字(g、j、p、q、yなど)。
  - ディセンダーのある文字の隣にある文字。
- ウィザードページでアクセスキーを割り当てる場合、"B"は[戻る]、"N"は[次へ]のために残しておきます。
- プロパティページでアクセスキーを割り当てる場合、"A"は[適用]のために残しておきます(使用する場合)。

## メニュー アクセスキー

- すべてのメニュー項目にアクセスキーを割り当てます。例外はありません。
- 動的なメニュー項目(最近使用されたファイルなど)については、数値的にアクセスキーを割り当てます。



この例では、Windows のペイント プログラムで、数字のアクセスキーを最近使用されたファイルに割り当てています。

- メニュー階層内で一意となるアクセスキーを割り当てます。別のメニュー階層ではアクセスキーを再び使用することができます。
- 次のようにして、アクセスキーを見つけやすくなります。
  - 最も使用頻度が高いメニュー項目については、ラベルの1番目または2番目(できれば1番目)の単語の先頭文字を選択します。
  - 使用頻度の低いメニュー項目については、ラベル内の特徴的な子音または母音を選択します。

## ダイアログ ボックス アクセスキー

- 可能な限り、すべての対話型コントロールまたはそのラベルに一意のアクセスキーを割り当てます。**読み取り専用のテキストボックス**は対話型コントロールである(ユーザーがスクロールし、テキストをコピーできる)ため、アクセスキーを割り当てるメリットがあります。以下には、アクセスキーを割り当てないでください。
  - [OK]、[キャンセル]および[閉じる]の各ボタン。EnterキーおよびEscキーが、それぞれのボタンのアクセスキーに使用されています。ただし、"OK"または"キャンセル"を意味する異なるラベルのコントロールには、アクセスキーを常に割り当てます。



この例では、肯定的なコミットボタンにアクセスキーを割り当てています。

- グループラベル。通常、グループ内の各コントロールにはアクセスキーが割り当てられているため、グループラベルに割り当てる必要はありません。ただし、アクセスキーが不足している場合は、グループラベルにアクセスキーを割り当て、各コントロールに割り当てないようにします。
- 汎用的なヘルプボタン。ヘルプには F1 キーを押してアクセスします。
- リンクラベル。リンクが多すぎて一意のアクセスキーを割り当てることができなかったり、リンクの下線によってアクセスキーを示す下線がわからなくなることがあります。代わりに、Tab キーを使用してリンクにアクセスするようにします。
- タブ名。タブ間を順番に移動するには、Ctrl + Tab キーおよび Ctrl + Shift + Tab キーを使用します。
- 「...」というラベルが付けられている参照ボタン。これらにはアクセスキーを一意に割り当てることができません。
- ラベルのないコントロール。スピンコントロール、グラフィックコマンドボタン、ラベルのない段階的表示コントロールなどがあります。
- ラベルのない静的テキストまたは対話型ではないコントロールのラベル。進行状況バーなどがあります。
- 標準的なアクセスキー割り当てになるように、最初にコミットボタンにアクセスキーを割り当てます。標準的なアクセスキー割り当てが存在しない場合は、最初の文字を使用します。たとえば、[はい] コミットボタンおよび [いいえ] コミットボタンのアクセスキーは、ダイアログボックスの他のコントロールに関係なく、常に "Y" および "N" にする必要があります。
- "Don't (~しない)" が使用された否定的なコミットボタン ([キャンセル] を除く) については、"Don't" の "n" にアクセスキーを割り当てます。"Don't (~しない)" が使用されていない場合は、標準のアクセスキー割り当てを使用するか、最初の文字を割り当てます。こうすることで、すべての否定的なコミットボタンのアクセスキー割り当てに一貫性が保たれます。
- アクセスキーを簡単に見つけることができるよう、ラベルの前の方の文字(最初の文字が理想的)をアクセスキーに割り当てます。ラベルの後半にキーワードが含まれている場合も同様です。
- アクセスキーの割り当ては最大 20 個とします。これにより、未割り当ての文字がいくつか残り、ローカライズでの作業に役立ちます。
- 対話型コントロールの数が多すぎるために一意のアクセスキーを割り当てることができない場合は、一意ではないアクセスキーを割り当てるすることができますが、以下の場合に限られます。
  - 他の方法ではコントロールにアクセスすることが困難である。
  - 一意ではないアクセスキーが、使用頻度の高いコントロールのアクセスキーと重複していない。
- ダイアログボックスではメニューバーを使用しません。ダイアログボックスのコントロールとメニュー項目で同じ文字が使用されるため、この場合は一意のアクセスキーを割り当てることが難しくなります。

## ショートカットキー

- ショートカットキーを最もよく使用するコマンドに割り当てます。使用頻度の低いプログラムや機能の場合、代わりにアクセスキーが使用されるので、ショートカットキーは必要ありません。
- ショートカットキーがタスクを実行する唯一の方法にならないようにします。マウスやキーボードの Tab

キー、方向キー、アクセスキーも使用できるようにする必要があります。

- よく知られているショートカットキーに別の機能を割り当てないようにします。よく知られているショートカットキーはユーザーに記憶されているので、機能に貫性がないと、ストレスやエラーの原因になります。Windowsプログラムで使用され、よく知られているショートカットキーについては、「[Windowsのキーボードショートカットキー](#)」を参照してください。
- システム全体に作用するショートカットキーをプログラムに割り当てないようにします。プログラムのショートカットキーは、プログラムに入力フォーカスがある場合にのみ効果を発揮します。
- ショートカットキーはすべてドキュメントに記載します。ショートカットは、メニューバーの項目、ツールバーのツールヒント、使用されるショートカットキーがすべて記載された単一のヘルプ記事に記載します。こうすると、ユーザーがショートカットキー割り当てを確認するのに役立ちます。ショートカットを非公開にしないでください。
  - 例外: コンテキストメニュー内にはショートカットキーの割り当てを表示しません。コンテキストメニューでは効率が最優先されるので、ここにはショートカットキーの割り当てを表示しません。



ショートカットキーがツールヒントに記載されています。

- プログラムで多数のショートカットキーを割り当てる場合、割り当てのカスタマイズ機能を用意します。そうすると、ユーザーが他製品と競合するショートカットキーを割り当て直すことができ、他製品からの移行が実現します。ほとんどのプログラムでは、この機能が必要になるほど多くのショートカットキーを割り当てません。

## ショートカットキーの選択

- よく知られているショートカットキーについては、標準の割り当てを使用します。Windowsプログラムの一般的なショートカットキーについては、「[Windowsのキーボードショートカットキー](#)」を参照してください。
- 非標準のキー割り当てを行う場合、使用頻度の高いコマンドには次の推奨ショートカットキーを使用します。これらのショートカットキーは、一般的なショートカットと競合することができないうえに、簡単に押すことができるという理由で推奨されています。
  - Ctrl + G、J、K、L、M、Q、R、T
  - Ctrl + 任意の番号
  - F7、F8、F9、F12
  - Shift + F2、F3、F4、F5、F7、F8、F9、F11、F12
  - Alt + 任意のファンクションキー (F4 以外)
- 使用頻度の低いコマンドには、次の推奨ショートカットキーを使用します。これらのショートカットキーには競合がありませんが、両手を使う必要があることが多く、押しにくいという難点があります。
  - Ctrl + 任意のファンクションキー (F4 および F6 以外)
  - Ctrl + Shift + 任意の文字または数字
- 頻繁に使用されるショートカットキーは次のようにして、記憶しやすくします。
  - 数字またはファンクションキーではなく、文字を使用します。
  - 先頭語の文字またはコマンドキーワード内の最も印象的な文字を使用するようにします。
- 選択されたオブジェクトに適用されるコマンドなど、小規模の効果を発揮するコマンドにはファンクションキーを使用します。たとえば、F2キーを押すと選択項目の名前を変更できます。
- ドキュメント全体に適用されるコマンドなど、大規模な効果を発揮するコマンドにはCtrlキーと組み合わせて使用します。たとえば、Ctrlキーを押しながらSキーを押すと、現在のドキュメントが保存されます。
- 標準のショートカットキーの操作を拡張または補完するコマンドにはShiftキーと組み合わせて使用します。たとえば、Altキーを押しながらTabキーを押すと、開いているメインウインドウが順に表示されますが、AltキーとShiftキーを押しながらTabキーを押すと、逆の順序で表示されます。同じように、F1キーを押すとヘルプが表示されますが、Shiftキーを押しながらF1キーを押すと、コンテキストに応じたヘルプが表示されます。

- 方向キーを使用してアイテムの移動やサイズ変更を行う場合、**Ctrl** キーを押しながら方向キーを押すとさらに微細にコントロールできます。

#### ショートカット キーの選択 (禁止事項)

- キーの位置の違いを区別しません。たとえば Windows では、左右の Shift キー、Alt キー、Ctrl キー、**Windows ロゴキー**、**アプリケーションキー**が認識され、数字キーパッド上のキーも認識されます。1つのキー位置にのみ動作を割り当てるることは混乱を招く用法であり、予測もできません。
- Windows** ロゴ修飾キーは、プログラムのショートカットキーに使用しません。Windows ロゴキーは Windows 用に確保されています。ある Windows ロゴキーの組み合わせが現時点で使用されていなくても、将来使用される可能性があります。
- アプリケーションキーは、ショートカットキーの修飾キーとして使用しません。代わりに、Ctrl キー、Alt キー、Shift キーを使用します。
- Windows で使用されるショートカットキーは、プログラムのショートカットキーに使用しません。使用すると、プログラムに入力フォーカスがあるとき、Windows システムのショートカットキーと競合します。Windows のショートカットキーについては、「[Windows のキーボードショートカットキー](#)」を参照してください。
- Alt + 英数字キーの組み合わせは、ショートカットキーに使用しません。このようなショートカットキーはアクセスキーと競合する可能性があります。
- ショートカットキーに次の文字を使用しないでください。@ £ \$ {} [] \ ~ | ^ ' < >。これらの文字は、言語ごとに異なるキーの組み合わせが必要か、ロケールに特有のものです。
- キーの組み合わせを複雑にしないようにします。たとえば、キーを3個以上組み合わせたり (Ctrl + Alt + Space など)、キーボード上の離れた位置にあるキーを組み合わせません (Ctrl + F5 など)。使用頻度の高いコマンドには、簡単なショートカットキーを使用します。
- Ctrl キーと Alt キーの組み合わせは使用しないでください。Windows では、一部の言語バージョンでこの組み合わせを AltGR キーと解釈し、英数字を生成するためです。

#### キーボードとマウスの併用

- リンクの場合、新しいウィンドウを使用してアクセスするには Shift キーを押しながらクリックし、新しいタブを使用してアクセスするには Ctrl キーを押しながらクリックします。この方法には、Windows Internet Explorer® と一貫性があります。

#### ドキュメント

キーボードに言及するときは、以下のこと留意します。

- "スクリーンキーボード" は、ユーザーがタッチして文字を入力する、画面上のキーボードに言及するときに使用します。
- キーボードの組み合わせでは、修飾キーから始まるように指定します。修飾キーは次の順序で表します。Windows ロゴキー、アプリケーションキー、Ctrl キー、Alt キー、Shift キー。テンキーの修飾キーを使用する場合、修飾先のキーの直前にそれを配置します。
- キーボード上のキーを表すのにすべて大文字の表記は使用しません。標準キーボードで使用される大文字化に従うか、キーボード上にラベルが付いていないキーには小文字を使用します。
  - アルファベットキーの組み合わせには、大文字を使用します。
  - "PageUp"、"PageDown"、"PrintScreen"、"ScrollLock" と表記します。
  - "正符号 (+)"、"マイナス記号 (-)"、"ハイフン (-)"、"ピリオド (.)"、"コンマ (,) " と表記します。
  - 方向キーについては、"← キー"、"→ キー"、"↑ キー"、"↓ キー" を使用します。方向キーにグラフィックラベルは使用しません。
  - アイコンラベル付きのキーに言及するには、"Windows ロゴキー" または "アプリケーションキー" を使用します。これらのキーにグラフィックラベルは使用しません。

#### 正しい例:

Space、Tab、Enter、PageUp、Ctrl + Alt + Del、Alt + W、Ctrl + 正符号 (+)

#### 間違った例:

SPACE、TAB、ENTER、PG UP、CTRL + ALT + DEL、Alt + w、Ctrl + +

キーの組み合わせを示すにはプラス記号 (+) を使用し、スペースは挿入しません。

正しい例:

Ctrl+A、Shift+F5

間違った例:

Ctrl-A、Shift + F5

- キーの組み合わせに、疑問符など Shift キーの使用を必要とする句読点が含まれることを示すには、その組み合わせに Shift キーを追加し、シフト先のキーの名前または記号を指定します。\$ ではなく # を使用するなど、シフト元のキーの名前を使用するとユーザーが混乱し、適切でないこともあります。たとえば、? と / は、どのキーボードでも必ず入れ替えが行われるとは限りません。

正しい例:

Ctrl + Shift + ?、Ctrl + Shift + \*、Ctrl + Shift + コンマ

間違った例:

Ctrl + Shift + /、Ctrl + ?、Ctrl + Shift + 8、Ctrl + \*

- 最初に言及する際、必要な場合はキー名に "キー" を付けて明確にします。たとえば、"F1 キー" のようにします。それ以降は、"F1 キーを押します" のようにキーの名前だけで言及します。
- プログラミングおよびその他の技術文書では、"アクセスキー" および "ショートカットキー" と具体的に記述します。"アクセラレータ"、"ニーモニック"、"ホットキー" は使用しません。それ以外のドキュメント(特にユーザー向けマニュアル)では、"キーボードショートカット" を使用します。

対話操作に言及するときは、以下のことに留意します。

- キーを押して直ちに離すと、プログラム内でアクションが起動されたり、ドキュメント内または UI 内の移動が生じる場合は "押す" を使用し、"押し下げる"、"たたく"、"打つ"、"入力する" は使用しません。
- ユーザーにテキスト入力を指示するには、"記入する" ではなく "入力する" を使用します。
- 方向キーまたはファンクションキーのようなキーの種類に言及する場合など、"押す" では混乱を招く可能性がある場合は "使用する" を使用します。このような場合に "押す" を使用すると、ユーザーはすべてのキーを同時に押す必要があると受け取る可能性があります。
- 修飾キーのようにキーを押したままにする場合は、"押しながら" を使用します。
- "押す" は、"クリック" の同義語として使用しません。

例:

- 名前を入力し、Enter キーを押します。
- Ctrl キーを押しながら F キーを押し、検索するテキストを入力します。
- ファイルを保存するには、Y キーを押します。
- 挿入ポイントを移動するには、方向キーを使用します。

# Windows のキーボードショートカットキー

## キーボード

次の表は、標準的な Microsoft® Windows® のショートカットキーの割り当てをまとめたものです。

次のショートカットはすべてのプログラムで使用できますが、以下の特定の意味を持たせる必要があります。

### 一般的なショートカット

キー	意味
F1	状況に応じたヘルプを表示します。
Shift + F1	状況に応じたヘルプを表示します (F1と同じです)。
F2	選択した項目の名前を変更します。
F3	次を検索します。
F5	アクティブ ウィンドウを更新します。
F10	アクティブ プログラムでメニュー バーをアクティブ化します。
Alt + Enter	選択した項目のプロパティ ダイアログ ボックスを表示します。
Ctrl + A	すべてを選択します。
Ctrl + C、 Ctrl + Ins	コピーします。
Ctrl + X、 Ctrl + Del	切り取ります。
Ctrl + V、 Shift + Ins	貼り付けます。
Ctrl + Y、 Alt + Shift + BackSpace	やり直します。
Ctrl + Z、 Alt + BackSpace	元に戻します。
Ctrl + F	[検索] ダイアログ ボックスを開きます。
Ctrl + H	[置換] ダイアログ ボックスを開きます。
Ctrl + N	新しい空白の文書や [新規作成] ダイアログ ボックスを開きます。
Ctrl + O	[開く] ダイアログ ボックスを開きます。
Ctrl + P	[印刷] ダイアログ ボックスを開きます。
Ctrl + S	作業中のドキュメントを保存します (通常、[名前を付けて保存] ダイアログ ボックスは開きません)。
Shift + F10	選択した項目のコンテキストメニューを表示します。

以下は、Windows コントロールの標準的なショートカットです。カスタム コントロールではショートカットを一貫して使用する必要があります。

### リスト ビューのショートカット

キー	意味
Home	最初の項目を選択します。
End	最後の項目を選択します。
Ctrl + Home	選択せずにフォーカスを最初の項目に移動します。
Ctrl + End	選択せずにフォーカスを最後の項目に移動します。
Ctrl + 方向キー + Space キー	連続していない複数の項目を選択できます。

1つまたは複数の印字されているキー	ラベルの先頭の文字が一致する項目に選択を移動します。
Ctrl + Shift + ← キー、 Ctrl + Shift + → キー	列の幅を変更します。

## ツリービューのショートカット

テンキー上のアスタリスク ク (*)	選択した項目のすべてのサブ項目を表示します。
テンキー上の正符号 (+)	選択した項目のすぐ下にあるサブ項目を表示します。
テンキー上のマイナス記号 (-)	選択した項目グループのすぐ下にある項目を折りたたみます。
← キー	現在の選択を折りたたみ(展開されている場合)、グループリーフのルートにフォーカスを移動します。
→ キー	選択した項目のすぐ下にあるサブ項目を表示します(折りたたまれている場合)。
Alt + ← キー	前のグループを表示します。
Alt + → キー	次のグループを表示します。
Ctrl + ↑ キー	選択を変更せずにビューをスクロールします。
Ctrl + ↓ キー	選択を変更せずにビューをスクロールします。
1つまたは複数の印字されているキー	ラベルの先頭の文字列が一致する項目に選択を移動します。

## 検索ボックスのショートカット

Ctrl + E	検索ボックスを選択します。
Alt + Enter	ローカル検索を使用して、入力された用語を検索します。
Shift + Enter	インターネット ブラウザーを使用して、入力された用語を検索します。
Ctrl + Shift + Enter	スタートメニューから使用している場合は昇格したプログラムを起動します。

## その他のコントロールのショートカット

F4、Alt + ↓ キー、Alt + ↑ キー	アクティブなドロップダウンリストまたはコンボ ボックスの項目を表示または非表示にします。
Ctrl + Tab、Ctrl + PageDown	タブ コントロールを順に移動します。
Ctrl + Shift + Tab、Ctrl + PageUp	タブ コントロールを逆順に移動します。
Ctrl + → キー	挿入ポイントを次の単語の先頭に移動します。
Ctrl + ↓ キー	挿入ポイントを前の単語の先頭に移動します。
Ctrl + Shift + いづれかの方向キー	テキストの範囲を選択します。
Alt + Shift + ↑ キー	選択した項目を上に移動します。
Alt + Shift + ↓ キー	選択した項目を下に移動します。

以下は、Windows で使用するために予約されているショートカットです。

## Windows のショートカット

キー	意味

Alt	メニュー バーをアクティブまたは非アクティブにします。
Alt + Esc	開いた項目の順に移動します。
Alt + Shift + Esc	開いた項目の逆順に移動します。
Alt + ハイフン (-)	アクティブな子ウィンドウ (マルチドキュメント インターフェイス (MDI) アプリケーション) のコンテキスト メニューを表示します。
アプリケーション キー	選択した項目のコンテキスト メニューを表示します。
PrintScreen	画面のイメージ全体をクリップボードにコピーします。
Alt + PrintScreen	アクティブ ウィンドウのイメージをクリップボードにコピーします。
Alt + Space キー	アクティブ ウィンドウのシステム メニューを表示します。
Alt + Tab	開いているメイン ウィンドウを順に移動します。
Alt + Shift + Tab	開いているメイン ウィンドウを逆順に移動します。
Ctrl + Alt + Tab	メニューを閉じずに、開いているメイン ウィンドウを順に移動します。
Ctrl + Alt + Shift + Tab	メニューを閉じずに、開いているメイン ウィンドウを逆順に移動します。
Ctrl + Esc	スタート メニューを表示します。
Ctrl + Shift + Esc	タスク マネージャーを起動します。
Ctrl + Alt + Del	Windows セキュリティ 画面を表示します。
Shift + <タスク バーのプログラムをクリック>	プログラムを起動します (Windows 7 の場合)。
Ctrl + Shift + <タスク バーのプログラムをクリック>	昇格したプログラムを起動します (Windows 7 の場合)。

#### ナビゲーションのショートカット

Alt + F4	アクティブ ウィンドウまたはアクティブ プログラムを閉じます。
Ctrl + F4	複数のドキュメントを開くことのできるプログラムで、作業中のドキュメント を閉じます。
Ctrl + Tab、 F6	プログラム内の次のウィンドウまたはパレットに移動します。
Ctrl + Shift + Tab、 Shift + F6	プログラム内の前のウィンドウまたはパレットに移動します。
Ctrl + F6	関連 ウィンドウ (または MDI ドキュメント ウィンドウ) グループ内の次の ウィンドウに移動します。
Ctrl + Shift + F6	関連 ウィンドウ (または MDI ドキュメント ウィンドウ) グループ内の前の ウィンドウに移動します。

#### Windows キーのショートカット

Windows ロゴ キー	スタート メニューを表示または非表示にします。
Windows ロゴ キー + ← キー	アクティブ ウィンドウを画面の左半分に配置します (Windows 7 の場合)。
Windows ロゴ キー + → キー	アクティブ ウィンドウを画面の右半分に配置します (Windows 7 の場合)。
Windows ロゴ キー + ↑ キー	アクティブ ウィンドウを最大化します (Windows 7 の場合)。
Windows ロゴ キー + ↓ キー	アクティブ ウィンドウを元のサイズに戻す、または最小化します (Windows 7 の場合)。

Windows ロゴ キー + Shift + ↑ キー	幅を維持した状態でアクティブ ウィンドウを縦方向に最大化します (Windows 7 の場合)。
Windows ロゴ キー + Shift + ↓ キー	幅を維持した状態でアクティブ ウィンドウを縦方向に元のサイズに戻す、または最小化します (Windows 7 の場合)。
Windows ロゴ キー + Shift + ← キー	ウィンドウをモニターの左側に移動します (Windows 7 の場合)。
Windows ロゴ キー + Shift + → キー	ウィンドウをモニターの右側に移動します (Windows 7 の場合)。
Windows ロゴ キー + Space キー	一時的にデスクトップを表示します (Windows 7 の場合)。
Windows ロゴ キー + P	プロジェクトオプションを表示します。方向キーを使用してオプションを選択します (Windows 7 の場合)。
Windows ロゴ キー + T	フォーカスをタスク バーに移動します (Windows 7 の場合)。
Windows ロゴ キー + Home	非アクティブな背面ウィンドウをすべて最小化します (Windows 7 の場合)。
Windows ロゴ キー + <数値>	タスク バーの特定の位置 (Windows 7) またはクリック起動 (Windows Vista) にあるプログラムの新しいインスタンスを起動します (例: Windows ロゴ キー + 1 で最初のプログラムを起動します)。
Windows ロゴ キー + B	通知領域にフォーカスを設定します。
Windows ロゴ キー + Break	[システムのプロパティ] ダイアログ ボックスを表示します。
Windows ロゴ キー + D	デスクトップを表示します。
Windows ロゴ キー + E	Windows エクスプローラーで [コンピューター] を開きます。
Windows ロゴ キー + F	ファイルまたはフォルダーを検索します。
Windows ロゴ キー + Ctrl + F	コンピューターを検索します (ネットワークに接続している場合)。
Windows ロゴ キー + G	サイドバー ガジェット間を順に移動します。
Windows ロゴ キー + L	コンピューターをロックするか (ネットワーク ドメインに接続されている場合)、ユーザーを切り替えます (ネットワーク ドメインに接続されていない場合)。
Windows ロゴ キー + M	すべてのウィンドウを最小化します。
Windows ロゴ キー + Shift + M	最小化されたウィンドウを元のサイズに戻します。
Windows ロゴ キー + R	[ファイル名を指定して実行] ダイアログ ボックスを開きます。
Windows ロゴ キー + Space キー	Windows サイドバーを前面に表示します。
Windows ロゴ キー + T	タスク バーにフォーカスを設定し、プログラム間を順に移動します。
Windows ロゴ キー + Tab	Windows フリップ 3D を使用して、タスク バーのプログラム間を順に移動します。

Windows ロゴ キー + Shift + Tab	Windows フリップ 3D を使用して、タスクバーのプログラム間を逆順に移動します。
Windows ロゴ キー + Ctrl + Tab	Windows フリップ 3D を使用して、タスクバーのプログラム間を方向キーを押して移動します。
Windows ロゴ キー + U	[コンピューターの簡単操作センター] を開きます。
Windows ロゴ キー + X	[Windows モビリティ センター] を開きます。
Windows ロゴ キー + F1	[Windows ヘルプとサポート] を開きます。
Windows ロゴ キー、プログラム名を選択、Enter	プログラムを起動します。

### アクセシビリティのショートカット

Alt (左側) + Shift (左側) + PrintScreen	ハイコントラストのオン/オフを切り替えます。
Alt (左側) + Shift (左側) + NumLock	マウス キー機能のオン/オフを切り替えます。
Shift キーを 5 回押す	固定キー機能のオン/オフを切り替えます。
Shift キー (右側) を 8 秒間押し続ける	フィルター キー機能のオン/オフを切り替えます。
NumLock キーを 5 秒間押し続ける	切り替えキー機能のオン/オフを切り替えます。

次のショートカットはすべてのプログラムで使用できますが、以下のように一貫した意味を持たせる必要があります。

### Windows エクスプローラーのショートカット

キー	意味
F3、または Ctrl + F	ファイルまたはフォルダーを検索します。
F4	アドレスバーの一覧を表示します。
Alt + D	アドレスバーを選択します。
Alt + ↑ キー	1 つ上の階層のフォルダーを表示します。
Alt + ← キーまたは Backspace	前に表示した場所に戻ります。
Alt + → キー	次に表示した場所に進みます。
Alt + F4 または Ctrl + W	アクティブな項目を閉じたり、アクティブ プログラムを終了します。
Ctrl + D または Del	選択した項目を削除し、ごみ箱に移動します。
Shift + Del	選択した項目をごみ箱に移動せずに削除します。
Ctrl + N	現在の場所を新しいウィンドウで開きます。
Ctrl + Shift を押しながら項目をドラッグ	選択した項目のショートカットを作成します。
Alt (左側) + Shift	入力言語やキーボード レイアウトを切り替えます(ユーザーが複数のキーボード レイアウトをインストールした場合に使用、構成できます)。
Ctrl + Shift	キーボード レイアウトや入力言語を切り替えます(ユーザーが複数のキーボード レイアウトをインストールした場合に使用、構成できます)。
Ctrl または Alt (左側) + Shift + ~ (ティルダ)	入力言語のショートカット キーです(ユーザーが複数の

記号) キー、数字(0~9) キー、またはア クサングラーブ(‘) キー	キーボードレイアウトをインストールした場合に使用、 構成できます。
Esc	現在のタスクまたは検索を取り消します。

## Windows Internet Explorer® のショートカット

### 一般的なショートカット

キー	意味
F11、Alt + Enter	全画面表示モードを有効または無効にします。
Ctrl + A	ページ上のすべての項目を選択します。
Ctrl + F	ページ上の単語または語句を検索します。
Ctrl + L	[開く] ダイアログ ボックスを開きます。
Ctrl + N	現在の Web ページを新しいウィンドウで開きます。
Ctrl + P	ページを印刷します。
Ctrl + 正符号 (+)	拡大表示します。
Ctrl + マイナス記号 (-)	縮小表示します。
Ctrl + 0	表示倍率を 100% にします。
Tab	アドレスバー、[更新] ボタン、検索ボックス、Web ページ上の項目間を順に移動します。

### ナビゲーションのショートカット

F5	ページを更新します。
Ctrl + F5	ページとキャッシュを更新します。
Alt + Home	ホーム ページに移動します。
Alt + ← キー	戻ります。
Alt + → キー	進みます。
Esc	ページのダウンロードを中止します。

### お気に入りセンターのショートカット

Ctrl + B	お気に入りを整理します。
Ctrl + D	現在のページを [お気に入り] に追加します。
Ctrl + H	履歴を開きます。
Ctrl + Shift + H	ピン モードで履歴を開きます。
Ctrl + I	お気に入りを開きます。
Ctrl + Shift + I	ピン モードでお気に入りを開きます。
Ctrl + J	フィードを開きます。
Ctrl + Shift + J	ピン モードでフィードを開きます。

### タブのショートカット

Ctrl + マウスの左ボタン	新しいタブを背面に作成してリンクを開きます。
Ctrl + Shift + マウスの左ボタン	新しいタブを前面に作成してリンクを開きます。

Ctrl + F4、 Ctrl + W	タブを閉じます (タブが 1 つしか開いていない場合はウィンドウを閉じます)。
Ctrl + Q	タブのサムネイルを表示する [クイック タブ] ビューを開きます。
Ctrl + T	新しいタブを開きます。
Ctrl + Shift + Q	開いているタブの一覧を表示します。
Ctrl + Tab	次のタブに切り替えます。
Ctrl + Shift + Tab	前のタブに切り替えます。
Ctrl + (数字)	指定のタブ番号に切り替えます。

#### アドレスバーのショートカット

F4	これまでに入力したアドレスの一覧を表示します。
Alt + D	アドレスバーを選択します。
Alt + Enter	アドレスバーに入力したアドレスの Web サイトを新しいタブで開きます。
Ctrl + Enter	アドレスバーのテキストの先頭に "http://www." を追加し、末尾に ".com" を追加します。
Ctrl + Shift + Enter	アドレスバーのテキストの先頭に "http://www." を追加し、末尾に指定した Web サイト アドレスの接尾辞を追加します。

#### 検索バーのショートカット

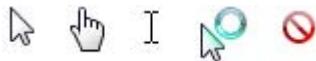
Alt + Enter	検索結果を新しいタブで開きます。
Ctrl + ↓ キー	検索プロバイダーの一覧を表示します。

# マウスとポインター

[デザインコンセプト](#)  
[ガイドライン](#)  
[クリックアフォーダンス](#)  
[標準的なマウス ボタン操作](#)  
[マウス操作](#)  
[マウス ホイール](#)  
[ポインターの非表示](#)  
[アクティビティ ポインター](#)  
[キャレット](#)  
[アクセシビリティ](#)  
[ドキュメント](#)

"マウス" は、Microsoft® Windows® 内のオブジェクトを操作する際に使用する、主要な入力デバイスです。マウスという用語は、ノート型コンピューターに組み込まれているトラックボール、タッチパッド、ポインティングスティックや、Windows Tablet とタッチテクノロジーおよびタッチスクリーンで使用されるペン、ユーザーの指など、その他のポインティングデバイスを指す場合もあります。

マウスを物理的に動かすと、画面上のグラフィック "ポインター" ("カーソル" とも呼ばれます) が移動します。ポインターには、現在の動作を表すさまざまな形があります。

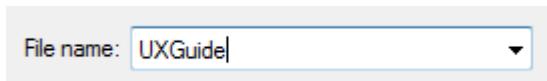


通常のマウス ポインター

今日の一般的なマウスには、主ボタン(通常は左ボタン)および副ボタン(通常は右ボタン)と、この2つのボタンに挟まれたマウス ホイールが備わっています。ポインターを移動してマウスの主ボタンや副ボタンをクリックすることにより、ユーザーはオブジェクトを選択してそれに対するアクションを行うことができます。ほとんどの操作において、マウス ボタンを押すことはターゲットの選択を意味し、ボタンを離すことはアクションの実行を意味します。

ビギナー ポインターを除くすべてのポインターには単一ピクセルの "ホットスポット" があり、これによってマウス ポインターの画面上の正確な位置が定められます。ホットスポットによって、マウス操作の影響を受けるオブジェクトが決まります。ホットスポットがオブジェクト上にあると判断する領域を "ホットゾーン" といい、各オブジェクトに定義されています。通常、ホットゾーンはオブジェクトの境界と一致しますが、対話操作を容易にするために、それより広い範囲に設定されている場合もあります。

"キャレット" は、ユーザーがテキスト ボックスまたはその他のテキスト エディターで入力を行うときに表示される、点滅する垂直のバーです。既定ではキャレットとポインターは別個のものです。ユーザーが入力を行っている間、Windows ではポインターが表示されません。



キャレット

注: [アクセシビリティ](#) および [タッチ](#) に関するガイドラインは、それぞれ別の項目として記載しています。

## デザイン コンセプト

マウスは直感的である

マウスが入力デバイスとして成功してきたのは、一般的な人間の手で使いやすいからです。ポインター ベースの操作が成功してきたのは、直感的で、多種多様なエクスペリエンスが可能になるためです。

優れたデザインのユーザーインターフェイス (UI) オブジェクトのことを、アフォーダンスがあると言います。アフォーダンスとは、オブジェクトの使用方法を示す視覚的、動作的な特性です。ポインターは手の代わりとして動作し、これによってユーザーは、物理的なオブジェクトを操作するのとまったく

同じように画面上のオブジェクトを操作できます。私たち人間は、人間の手がどのように動くか、生まれもって理解しています。したがって、押せるように見えるものがあれば押そうとし、つかめそうに見えればつかもうとします。したがって、アフォーダンスが強いオブジェクトであれば、ユーザーはそれを見て試すだけで使用方法を理解できます。



これらのオブジェクトには、強いアフォーダンスがあります。

一方、アフォーダンスの弱いオブジェクトは、よりわかりにくくなります。そのようなオブジェクトは、ほとんどの場合、説明のためのラベルや指示テキストが必要です。



これらのオブジェクトは、アフォーダンスが不足しています。

マウス使用には直感的ではない面もある

右クリック、ダブルクリック、Shift修飾キーまたはCtrl修飾キーを押しながらのクリックの3つは、直感的でないマウス操作です。これらは、現実世界の動作を模倣したものではありません。キーボードショートカットやアクセスキーとは異なり、これらのマウス操作は、通常、UIのどこにも記述されていません。これは、基本的なタスクの実行において、右クリック、ダブルクリック、および修飾キーが、特に初心者ユーザーにとって、必要になるべきではないことを意味します。また、これらの高度な操作を効果的に使用するには、動作が一貫しており、予測可能である必要があることも意味しています。

#### シングルクリックかダブルクリックか

ダブルクリックは、Windowsデスクトップで非常に広範囲に使用されているため、高度な操作には見えないかもしれません。たとえば、Windowsエクスプローラーのファイルウィンドウでフォルダー、プログラム、またはドキュメントを開く操作は、ダブルクリックを使用して実行します。Windowsデスクトップ上のショートカットを開く操作でも、ダブルクリックを使用します。一方、スタートメニューでフォルダーやプログラムを開く場合はシングルクリックを使用します。クイック起動バーからプログラムを起動する場合も同様です。

選択可能なオブジェクトは、シングルクリックを使用して選択するので、開く場合にはダブルクリックが必要です。これに対して、選択不可のオブジェクトはシングルクリックだけで開きます。多くのユーザーは、この違いを理解していません。その結果、一部のユーザーは必要な操作ができるまでアイコンをクリックし続けることになります。

#### 直接操作

オブジェクトに直接影響を及ぼす操作を、直接操作と呼びます。一般的な直接操作としては、ポイント、クリック、選択、移動、サイズ変更、分割、スクロール、パン、ズームなどがあります。これに対して、プロパティ ウィンドウやその他のダイアログ ボックスを介して行うオブジェクト操作を、間接操作と呼ぶことができます。

ただし、直接操作を行う場面では、操作ミスが起こる可能性があり、そのために寛容性が必要になります。“寛容性”とは、好ましくない操作を簡単に元に戻したり、修正したりできることです。元に戻す機能を提供し、優れた視覚的なフィードバックを提示して、ユーザーが誤りを簡単に修正できるようにすることによって、直接操作に寛容性を持たせます。寛容性に関連して、好ましくない操作が実行されることを最初の段階で防ぐこともできます。これは、意図しない結果を招く可能性がある危険な操作やコマンドに対して、制限付きのコントロールと確認を使用することで実現できます。

#### 標準的なマウス ボタン操作

標準的なマウス操作は、さまざまな要因によって決まります。これらの要因には、クリックされるマウ

スキー、クリック回数、クリック中の位置、キーボード修飾キーが押されたかどうか、などがあります。これらの要因が一般に操作に与える影響を以下にまとめます。

- ほとんどのオブジェクトでは、左ボタンをダブルクリックすると、左ボタンのシングルクリックが実行され、既定のコマンドが実行されます。既定のコマンドは、コンテキストメニューによって特定されます。
- 一部の種類の選択可能なオブジェクトでは、クリックするたびにクリックの効果が拡大します。たとえばテキストボックスでは、シングルクリックによって入力位置が設定され、ダブルクリックによって単語が選択され、トリプルクリックによって文または段落が選択されます。
- 右クリックすると、オブジェクトのコンテキストメニューが表示されます。
- オブジェクトをポイントしたままになると、ホバー状態になります。
- マウスボタンを押しているときにマウスを静止させた状態を保つと、クリックおよび單一オブジェクトの選択になります。マウスの移動は、移動、サイズ変更、分割、ドラッグ、複数オブジェクトの選択を示します。
- Shiftキーは、選択範囲を連続的に拡張します。
- Ctrlキーは、クリックした項目の選択状態を切り替えることで、他のオブジェクトの選択に影響することなく選択範囲を拡張します。

### 一般的なポインター形状

以下の表では、一般的なポインター形状と、その操作および効果について説明します。

#### 単純なマウス操作

単純なアクション	対話操作	通常の効果
ポイント	ポインターを特定のオブジェクトに移動します。マウスボタンのクリックは行いません。	対象がポイント状態で表示され、動的なアフォーダンスがあれば表示されます。
ホバー	ポインターを特定のオブジェクトに移動します。マウスボタンのクリックは行わず、少なくとも1秒間は移動しません。	対象のツールヒント、情報ヒント、またはそれと同等のものが表示されます。
クリック	特定の選択不可オブジェクトにポインターを移動し、そこから移動させることなくマウスボタンを押して離します。クリックは、マウスボタンを離したときに効果が発生します。これは、マウスポインターを対象から別の場所に移動させることによって、ユーザーがクリックを取り消すことができるようになります。したがって、マウスボタンを押しただけでは、選択された対象が示されるだけです。	主ボタンのシングルクリックの場合、オブジェクトがアクティブ化されます。主ボタンのダブルクリックの場合、オブジェクトがアクティブ化され、既定のコマンドが実行されます。副ボタンの場合、オブジェクトのコンテキストメニューが表示されます。
選択	特定の選択可能なオブジェクトにポインターを移動し、マウスボタンを押して離します。	主ボタンのシングルクリックの場合、オブジェクトが選択されます。ユーザーがドラッグした場合、連続した範囲のオブジェクトが選択されます。主ボタンのダブルクリックの場合、オブジェクトが選択され、既定のコマンドが実行されます。  テキストの場合、主ボタンをクリックすると挿入ポイントが設定され、2度目のクリックで挿入ポイントにある単語が選択され、3度目のクリックで文または段落が選択され

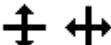
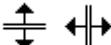
		ます。
押下	特定のオブジェクトにポインターを移動させ、マウス ボタンを押したまま離しません。	自動繰り返し機能(スクロール矢印を押して継続的にスクロールするなど)の場合は、繰り返しアクティブ化されます。それ以外の場合は、移動せずに離す操作を行わない限り、移動、サイズ変更、分割、またはドラッグの開始を示します。
ホイール操作	マウス ホイールを動かします。	マウス ホイールと同じ方向にウィンドウが垂直スクロールします。

## ポインター

形状	名前	使用される時
	通常の選択	ほとんどのオブジェクトの選択に使われます。
	リンクの選択	テキスト リンクやグラフィック リンクはアフォーダンスが弱いため、その選択に使用されます。
	テキストの選択	テキストで文字間の位置を示すために使われます。
	精密な選択	グラフィックやその他の 2D 操作に使われます。

## マウスの複合操作

複合アクション	対話操作	通常の効果	ポインター
移動	<p>コマンドを実行して移動モードに入ってから行う移動の場合は、移動モードに移行し、ポインターを移動可能なオブジェクトに合わせて、マウス ボタンを押し、マウスを動かしてからマウス ボタンを離します。この場合、モードを示すためにポインターの形状が変化します。</p> <p>それ以外の場合は、ポインターを移動可能オブジェクトのグラバーに合わせ、マウス ボタンを押し、マウスを動かしてからマウス ボタンを離します。この場合、ポインターの形状が変化する必要はありません。</p>	<p>オブジェクトは、ポインターが移動する方向に移動します。</p>	 ワインドウを任意の方向に移動するために使用します。   オブジェクトをウィンドウ内で任意の方向に移動するために使用します。
サイズ変更	サイズ変更可能な境界線またはサイズ変更ハンドルにポインターを合わせ、マウス ボタンを押して、マウスを動かしてからマウス ボタンを離します。	オブジェクトのサイズは、ポインターが移動する方向に変化します。	 垂直および水平のサイズ変更   1 方向のサイズを変更するために使用します。   斜め方向のサイズ変更   同時に 2 方向のサイズを変更するため使用します。

			行および列のサイズ変更  グリッド内の行または列のサイズを変更するために使用します。
分割	スプリッターにポインターを合わせ、マウスボタンを押して、マウスを動かしてからマウスボタンを離します。	分割ウィンドウの境界線は、ポインターが移動する方向に移動します。	ウィンドウスプリッター  分割ウィンドウを垂直または水平にサイズ変更するために使用します。
ドラッグアンドドロップ	ドラッグ可能なオブジェクトにポインターを合わせ、マウスボタンを押して、マウスポインターをドロップターゲットまで移動してからマウスボタンを離します。	オブジェクトは、ドロップターゲットに移動またはコピーされます。	通常の選択  有効なドラッグターゲットに対して使用されます。特定の効果を示すための情報ヒントがある場合もあります。 使用不可  この画面が有効なドロップターゲットではないことを示すために使用されます。

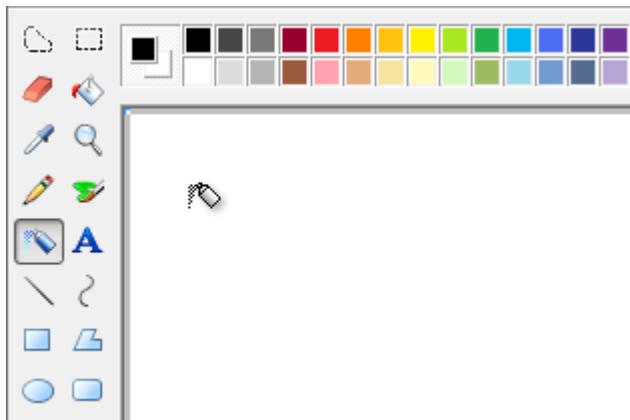
### アクティビティ インジケーター

完了するまでに数秒以上かかるアクションを実行するときに、ユーザーに対して表示されるポインターを、次の表に示します。

形状	名前	使用される時
	ビジュー ポインター	ウィンドウが応答可能になるまで待機するときに使用されます。
	バックグラウンドで作業中ポインター	バックグラウンドでタスクを処理しているときにポイント、クリック、プレス、選択を行うために使用されます。

### マウスのカスタム形状

ユーザーが標準以外の直接操作を複数実行できるアプリケーションでは、直接操作モードのパレット ウィンドウが提供されます。



この例では、ペイントには直接操作モードのパレット ウィンドウがあります。

#### 手の形のポインター

テキスト リンクおよびグラフィック リンクはアフォーダンスが弱いため、手の形の ("リンク選択") ポ

インター (人差し指を立てた手の形 ) が使用されます。リンクがあることを示す他の視覚的な手掛け (下線や特殊な配置方法など) を使用することもできますが、ポイント時に手の形のポインターを表示することが、リンクを示す最も確実な方法です。

混乱を避けるために、その他の目的では手の形のポインターを絶対に使用しないでください。たとえば、コマンド ボタンには既に強いアフォーダンスがあるので、手の形のポインターは必要ありません。手の形のポインターに、"このターゲットはリンクです" という以外の意味があつてはなりません。

#### フィットの法則

フィットの法則とは、グラフィカル ユーザー インターフェイスのデザインの人間工学における有名な法則で、基本的に以下のことを述べています。

- ターゲットが遠くなればなるほど、マウスで捕捉するための時間が長くなる。
- ターゲットが小さくなればなるほど、マウスで捕捉するための時間が長くなる。

したがって、ターゲットは大きい方がよいことになります。ターゲット領域全体がクリック可能になるようにしてください。

#### 間違った例:

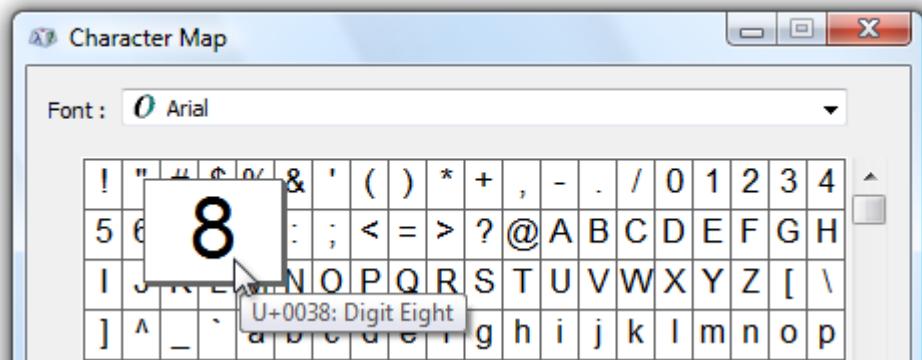


#### 正しい例:



正しい例では、ターゲット全体がクリック可能です。

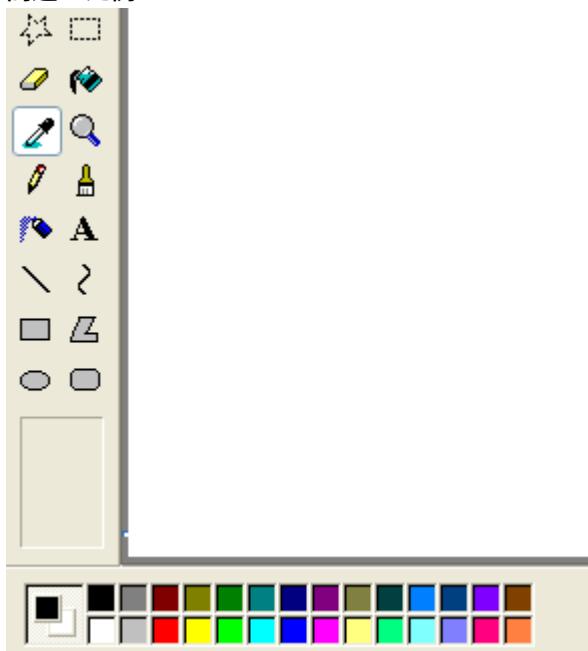
ポイント時のターゲットのサイズを動的に変更して、捕捉しやすくすることができます。



この例では、捕捉しやすくするために、ユーザーがポイントしたときにターゲットが大きくなります。

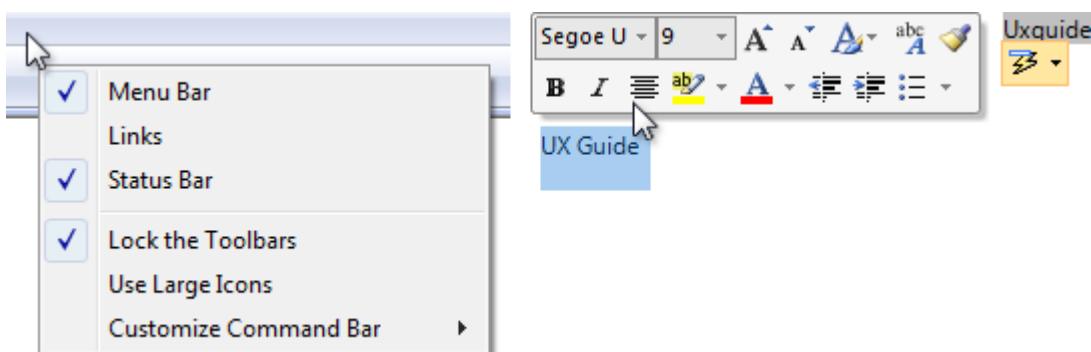
また、ターゲットは近くにあることも重要です。クリック可能な項目は、使用される可能性が高い場所の近くに配置します。

間違った例:



この例では、カラー パレットの場所が、使用される場所から離れすぎています。

ユーザーの現在のポインター位置がターゲットに近ければ近いほど、捕捉が簡単になるということを考慮します。したがって、コンテキストメニューは、Microsoft Office で使用されるミニ ツールバー やスマート タグと同様、フィットの法則を十分に活かしています。



現在のポインターの位置は常に、最も簡単に使用できる場所にあります。

また、オブジェクトのサイズを決めるときは、代替の入力デバイスを考慮します。たとえば、タッチに推奨する最小のターゲット サイズは 23 × 23 ピクセル (13 × 13 DLU) です。

#### マウスのない環境

すべての Windows 環境にマウスがあるとは限りません。たとえば、キオスクではマウスがあることは

稀で、通常は代わりにタッチスクリーンが使用されています。これは、ユーザーが左クリックや、おそらくドラッグ アンド ドロップなどの単純な操作を実行できることを意味します。ただし、ホバー、右クリック、またはダブルクリックはできません。これらの制限事項は、通常、前もって知ることができるために、このような状況のためのデザインは簡単です。

マウスを使用するには微細運動能力が必要なので、すべてのユーザーがマウスを使うことができるわけではありません。できるだけ多くのユーザーが使用できるソフトウェアにするためには、微細運動能力が必須でないすべての操作を、代わりにキーボードでも実行できるようにします。

その他の情報とガイドラインについては、「[アクセシビリティ](#)」を参照してください。

#### 4つの重要な点

1. マウス操作の動作は標準の効果と矛盾しないようにし、標準のポインターが適切である場合は必ずこれを使用します。
2. 高度なマウス操作(右クリック、複数クリック、または修飾キーが必要な操作)は、上級ユーザー向けの高度なタスクに限定します。
3. 高度なマウス操作には、効果的に使用できるように、一貫性のある予測可能な動作を割り当てます。
4. 好ましくないアクション(特にリスクのあるコマンド)を元に戻すまたは修正する機能をプログラムで提供します。直接操作では、操作ミスが起こる可能性が高くなります。

## ガイドライン

### クリック アフォーダンス

- オブジェクトがクリック可能かどうかを判断するために、実際にクリックすることをユーザーに要求しないようにします。オブジェクトを見ただけで、クリックできるかどうかが判断できるようにします。
  - プライマリ UI(コミット ボタンなど)には、静的なクリックアフォーダンスを設定します。ユーザーは、プライマリ UI を判断するのにマウス ポインターを合わせる必要がなくなります。
  - セカンダリ UI(副コマンドや段階的表示コントロールなど)では、ポイント時にクリックアフォーダンスを表示できます。
  - テキストリンクはリンクテキストを静的に示唆し、ポイントされたときにクリックアフォーダンス(下線やその他の表示の変更、および手の形のポインター)を表示するようにします。
  - グラフィックリンクでは、ポイント時に手の形のポインターを表示するだけにします。
- 手の形の(または"リンク選択"の)ポインターは、テキストリンクおよびグラフィックリンクに対してのみ使用します。このようにしないと、ユーザーはオブジェクトがリンクであるかどうかを判断するために実際にクリックしてみなければならなくなります。

### 標準的なマウス ボタン操作

次の表に、ほとんどの場合に当たるマウス ボタン操作を要約します。

対話操作	効果
ホバー	対象のツールヒント、情報ヒント、またはそれと同等のものが表示されます。
左ボタンをシングルクリック	オブジェクトがアクティブ化または選択されます。テキストの場合は、挿入ポイントが設定されます。
右ボタンをシングルクリック	オブジェクトが選択され、コンテキストメニューが表示されます。
左ボタンをダブルクリック	オブジェクトがアクティブ化または選択され、既定のコマンドが実行されます。テキストの場合は、挿入ポイントにある単語が選択されます(3回目のクリックでは、文または段落が選択されます)。
右ボタンをダブルクリック	右ボタンのシングルクリックと同じです。
Shift キーを押しながら左ボタンをクリック	選択可能なオブジェクトの場合は、選択範囲が連続的に拡張されます。それ以外の場合、左ボタンのシングルクリックと同様です。ただし、効果が異なる可能性があります。

タンをシングルクリック	ます。たとえば、ペイントでは、Shift修飾キーを押しながら橙円を描くと、円が描画されます。
Shiftキーを押しながら右ボタンをシングルクリック	Shiftキーを押しながら左ボタンをシングルクリックした場合と同じです。
Shiftキーを押しながら左ボタンをダブルクリック	Shiftキーを押しながら左ボタンをシングルクリックした場合と同様で、選択範囲全体に対して既定のコマンドが実行されます。
Shiftキーを押しながら右ボタンをダブルクリック	Shiftキーを押しながら左ボタンをシングルクリックした場合と同じです。
Ctrlキーを押しながら左ボタンをシングルクリック	選択可能なオブジェクトの場合、他のオブジェクトの選択状態に影響を与えることなく、クリックされた項目の選択状態を切り替えることによって、選択範囲を拡張します(したがって、非連続的な範囲を選択可能です)。それ以外の場合は、左ボタンをシングルクリックした場合と同じです。
Ctrlキーを押しながら右ボタンをシングルクリック	Ctrlキーを押しながら左ボタンをシングルクリックした場合と同じです。
Ctrlキーを押しながら左ボタンをダブルクリック	Ctrlキーを押しながら左ボタンをシングルクリックした場合と同様で、選択範囲全体に対して既定のコマンドが実行されます。
Ctrlキーを押しながら右ボタンをダブルクリック	Ctrlキーを押しながら左ボタンをシングルクリックした場合と同じです。

## マウス操作

- クリックターゲットは $16 \times 16$ ピクセル以上にして、どんな入力デバイスでも簡単にクリックできるようにします。タッチの場合、コントロールの推奨される最小サイズは $23 \times 23$ ピクセル( $13 \times 13$ DLU)です。小さいターゲットの場合は、捕捉しやすくするために、ユーザーがポイントしたときにサイズを動的に変更することを検討します。

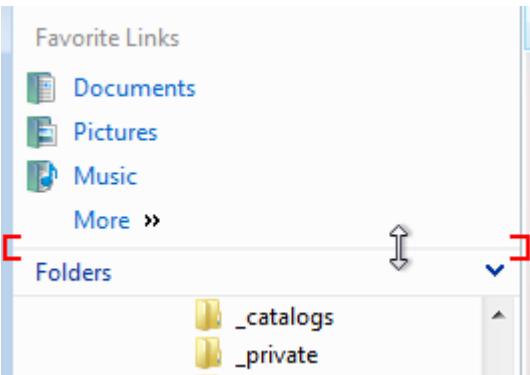
間違った例:



この例では、タッチパネルやペンを使用するにはスピンコントロールのボタンが小さすぎます。

- スプリッターの幅は5ピクセル以上にして、どんな入力デバイスでも簡単にクリックできるようにします。小さいターゲットの場合は、捕捉しやすくするために、ユーザーがポイントしたときにサイズを動的に変更することを検討します。

間違った例:



この例では、Windows エクスプローラーのナビゲーション ウィンドウのスプリッターの幅が狭すぎて、マウスやペンで効果的に使用できません。

- ユーザーに対して、空間的な誤差の範囲を許可します。ユーザーがマウス ボタンを離すときは、多少のマウス移動(例: 3 ピクセル)を許容します。ユーザーがマウス ボタンを離すとき、微妙にマウスを動かしてしまうことがあるため、ボタンを離した直後よりも、離す直前のマウス位置の方が、ユーザーの意図がより反映されているといえます。
- ユーザーに対して、時間的な誤差の範囲を許可します。シングルクリックとダブルクリックを判別するには、システムのダブルクリック速度を使用します。
- クリックは、ボタンが離されたときに有効になるようにします。ユーザーがマウス ボタンを離す前に、有効なターゲットからマウス ポインターを外すことによって、マウス アクションを破棄できるようにします。ほとんどのマウス操作において、マウス ボタンを押すことはターゲットの選択だけを意味し、ボタンを離すことでのアクションが実行されます。自動繰り返し機能(スクロール矢印を押して継続的にスクロールするなど)は例外です。
- 選択、移動、サイズ変更、分割、およびドラッグの場合は、マウスをキャプチャします。
- 移動、サイズ変更、分割、およびドラッグなど、マウスの複合操作は、Esc キーを使用すると破棄できるようになります。
- オブジェクトはダブルクリックに対応していないのに、ユーザーは対応していると考える可能性がある場合は、"ダブルクリック"をシングルクリックと解釈するようにします。ユーザーが 2 回ではなく 1 回のアクションを意図したと考えます。

間違った例:



ユーザーは、タスクバーのボタンがダブルクリックに対応していると考える可能性があるため、"ダブルクリック"をシングルクリックとして処理する必要があります。Windows Vista® では、ウィンドウが最小化されたときに正しく動作するようになっています。

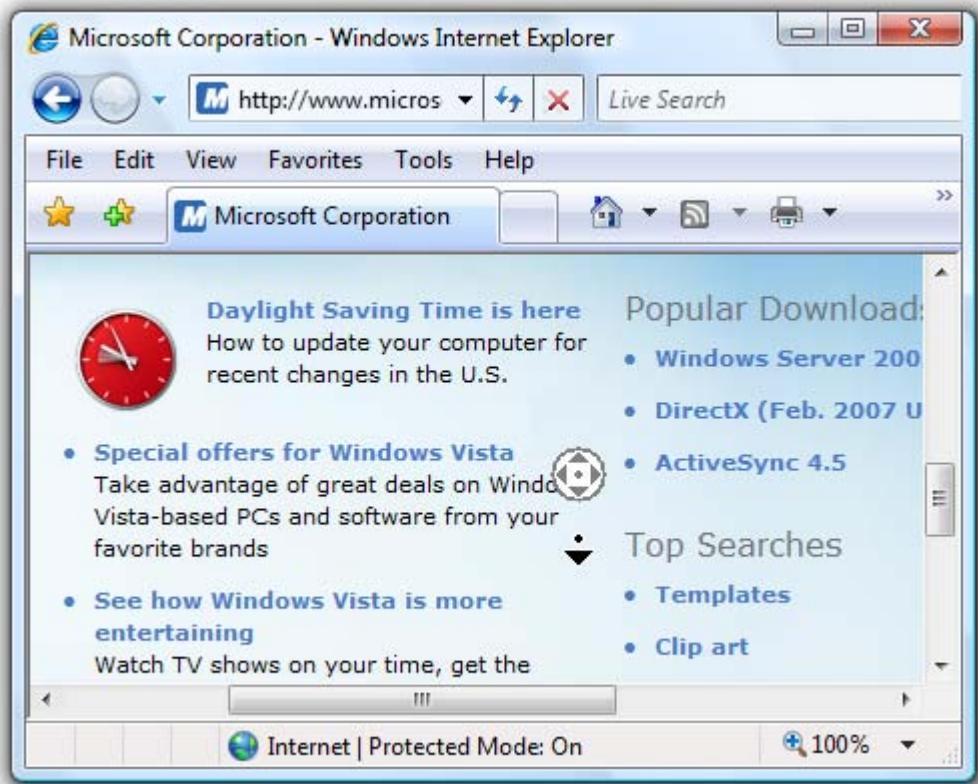
- プログラムがアクティブでないときは、不必要的マウス クリックを無視します。たとえば、プログラムがアクティブでないときにユーザーがマウス ボタンを 10 回クリックした場合は、それをシングルクリックと解釈します。
- ダブルドラッグまたはダブルコードは使用しません。ダブルドラッグとは、ダブルクリックで開始するドラッグ アクションのことです。ダブルコードとは、複数のマウス ボタンを同時に押すことです。これらの操作は標準でない上、わかりづらく、実行が難しいため、偶発的に実行される可能性が高くなります。
- マウス操作の修飾キーとして、Alt キーは使用しません。Alt キーは、ツールバーへのアクセスおよびアクセス キー用に予約されています。
- マウス操作の修飾キーとして、Shift + Ctrl は使用しません。この修飾キーは難しすぎて使用できません。
- ホバーを唯一の操作にしません。タッチ可能なプログラムにするには、ホバーをフルに活用しますが、その場合はアクションを実行するためにホバーを必要としないようにします。これは、通常、アクションがクリックでも実行できることを意味しますが、厳密に同じ方法である必要はありません。ホバーはほとんどのタッチ テクノロジーではサポートされていないため、そうしたタッチスクリーンを使用しているユーザーは、ホバーを必要とするタスクを実行できません。

## マウス ホイール

- マウス ホイールは、現在ポインターが指しているコントロール、ペイン、またはウィンドウに作用するよう

にします。このようにすることで、予期しない結果を防ぐことができます。

- マウス ホイールは、クリックまたは入力フォーカスがなくても有効になるようにします。ホバーで十分です。
- マウス ホイールは、最も具体的な範囲でオブジェクトに作用するようにします。たとえば、スクロール可能なウィンドウ内のスクロール可能なペインにあるスクロール可能なリスト ボックス上にポインターがある場合、マウス ホイールはリスト ボックス コントロールに作用します。
- マウス ホイールを使用するときに、入力フォーカスを変更しないでください。
- マウス ホイールは、以下のとおりに動作するようにします。
  - スクロール可能なウィンドウ、ペイン、コントロールの場合:
    - マウス ホイールを回転させると、オブジェクトが垂直にスクロール(上に回転させれば上へスクロール)します。ホイールの対応関係が自然になるように、マウス ホイールを回転したときは、決して水平にはスクロールしません。水平にスクロールすることは予期されおらず、わかりづらいからです。
    - Ctrl キーが押された場合、マウス ホイールを回転させるとオブジェクトが拡大/縮小します。上に回転すると拡大し、下に回転すると縮小します。
    - マウス ホイールを傾けると、オブジェクトが水平にスクロールします。
  - 拡大/縮小可能なウィンドウおよびペイン(スクロールバーなし)の場合:
    - マウス ホイールを回転させると、オブジェクトが拡大/縮小します。上に回転すると拡大し、下に回転すると縮小します。
    - マウス ホイールを傾けても何も起りません。
  - タブの場合:
    - マウス ホイールを回転させると、現在のタブを変更できます。タブの向きは関係ありません。
    - マウス ホイールを傾けても何も起りません。
  - Shift キーおよび Alt キーが押されている場合、マウス ホイール操作は無効です。
- 垂直スクロール サイズ(回転用)および水平スクロール サイズ(チルト用)には、Windows のシステム設定を使用します。これらの設定は、[マウス] コントロール パネル アイテムで構成できます。
- マウス ホイールを速く回転させたときに、スクロールも速くなるようにします。このようにすることで、ユーザーは膨大なドキュメントをより効率的にスクロールできます。
- スクロール可能なウィンドウの場合は、マウス ホイール ボタンをクリックしたときにウィンドウを "リーダー モード" にすることを検討します。リーダー モードでは、スクロール始点を示す特殊なアイコンが配置され、このスクロール始点に対応する方向と速度にウィンドウがスクロールします。



この例の *Windows Internet Explorer*® は、リーダー モードに対応しています。

#### ポインターの非表示

- ポインターは非表示にしません。次の場合は例外です。
  - フルスクリーンのプレゼンテーション モードで実行中のプレゼンテーション アプリケーションでは、ポインターを非表示にできます。ただし、ユーザーがマウスを動かしたときは、直ちにポインターを表示する必要があります。2秒間非アクティブな状態が続けば再度非表示にできます。
  - マウスのない環境(キオスクなど)では、ポインターを完全に非表示にできます。
- 既定では、ユーザーがテキスト ボックスに入力している最中は Windows によってポインターが非表示にされます。この Windows システム設定は、[マウス] コントロール パネル アイテムで構成できます。

#### アクティビティ ポインター

Windows におけるアクティビティ ポインターは、ビジー ポインター( )およびバックグラウンドで作業中ポインター( )です。

- ビジー ポインターは、アクションが完了するまでユーザーが1秒以上待機しなければならないときに表示します。ビジー ポインターにはホット スポットがないので、このポインターが表示されている間、ユーザーは何もクリックできません。
- バックグラウンドで作業中ポインターは、アクションが完了するまでユーザーは1秒以上待機しなければならないが、プログラムは応答可能であり、アクションが未完了であることを示す視覚的なフィードバックが他にないときに表示します。
- アクティビティ ポインターは、進行状況バーや進行状況アニメーションと一緒に使用しないでください。

#### キャレット

- キャレットは、テキスト入力ウィンドウまたはテキスト入力コントロールに入力フォーカスが来るまでは表示しません。キャレットは、ユーザーに対して入力フォーカスを示しますが、ウィンドウやコントロールには入力フォーカスがなくてもキャレットを表示できます。コンテキスト外のダイアログ ボックスにキャレットが表示できるように、入力フォーカスは取得しないでください。

間違った例:



コンテキストと関係なく Windows 資格情報マネージャーが表示され、キャレットが表示されていますが、入力フォーカスはありません。その結果、ユーザーは予期しない場所にパスワードを入力することになります。

- キャレットは、ユーザーが最初に入力すると思われるところに配置します。通常、これはユーザーが最後に入力していた場所か、またはテキストの末尾です。

## アクセシビリティ

- マウスをまったく使用できないユーザーに対しては、マウスをキーボードで置き換えるようにします。
  - ユーザーは、絵を描く、ゲームで遊ぶ、といった微細運動能力が必要な動作を除き、マウスで実行できるすべての操作をキーボードでも実行できる必要があります。
  - ユーザーは、効率的なテキスト入力を除き、キーボードで実行できるすべての操作をマウスでも実行できる必要があります。
- マウスの使用が困難なユーザーの場合は、以下のようにします。
  - 操作を実行する方法を、ダブルクリックとドラッグのみに限定しないようにします。

その他の情報とガイドラインについては、「[アクセシビリティ](#)」を参照してください。

## ドキュメント

マウスに言及するときは、以下のことに留意します。

- 複数形の "mice" は使用しません (英語の場合)。複数のマウスに言及する必要がある場合は、"複数のマウスデバイス" と表現します。
- マウスの左ボタンを示すときは、"マウス ボタン" を使用します。"マウスの主ボタン" という表現は使用しません。同様に、"マウスの副ボタン" ではなく "マウスの右ボタン" と呼びます。正確さに関係なく、ユーザーはこれらの用語を理解できるので、ボタンを再プログラムするユーザーは、考え方を切り替えます。
- マウス ホイールの回転部分を "ホイール"、そのクリック可能な部分を "ホイール ボタン" と呼びます。
- マウス アクションに言及するときは、"クリックする"、"ポイントする"、"ドラッグする" などの動詞を使用します。ユーザーはホイールを垂直に "回転" させ、水平に "チルト" し、ホイール ボタンを "クリック" します。
- ドキュメントまたはフォルダーを移動するアクションのことは、"ドラッグ アンド ドロップ" ではなく "ドラッグ" と呼びます。"フォルダーの移動は、ドラッグ アンド ドロップ操作です" のように、"ドラッグ アンド ドロップ" を形容詞として使用することは可能です。
- "ダブルクリック"、および "右クリック" は、動詞としてハイフンでつなぎます (英語の場合)。
- "～の上をクリックする" ではなく、"～をクリックする" という表現を使用します。"～内をクリックする" ("ウィンドウ内をクリックする" など) という表現は問題ありません。

マウス ポインターに言及するときは、以下の点に留意します。

- マウス ポインターのことは "ポインター" と呼びます。"カーソル" という用語は技術文書でのみ使用します。
- アクティビティ インジケーター付きのポインターについては、アクティビティ インジケーターのみでできているポインターを "ビジー ポインター"、ポインターとアクティビティ インジケーターの組み合わせを "バックグラウンドで作業中ポインター" と呼びます。
- その他のタイプのポインターの場合は、ポインターに言及するときに説明的な名前を使用しないでください。必要な場合は、グラフィックを使用して、マウス ポインターが画面上にどのように表示されるかを説明します。

例:

- ウィンドウの境界線をポイントします。
- マウスを使用して、最小化ボタンをクリックします。
- Shift キーを押しながら、マウスの右ボタンをクリックします。
- ポインターが  の形になったら、ポインターをドラッグして分割線を移動します。

## タッチ

すべての Microsoft® Windows® アプリケーションには優れたタッチ エクスペリエンスを持たせる必要があります。これは、思いのほか簡単に実現できます。

[デザインコンセプト](#)

[ガイドライン](#)

[コントロールの使い方](#)

[コントロールのサイズ](#)

[コントロールのレイアウトと間隔](#)

[対話操作](#)

[Windows タッチ ジェスチャー](#)

[寛容性](#)

[ドキュメント](#)

"タッチ" とは、Windows で実現されている、指を使ってコンピューターと直接対話する方法です。多くの場合、タッチはマウス、キーボード、またはペンを使用するよりも自然で、魅力的で、便利です。



タッチを使用しているところ。

多くのタッチ操作は、ジェスチャー や フリックを使って行います。"ジェスチャー" とは、画面上で行われる 1 本以上の指のすばやい動きです。コンピューターではこの操作がマウスの動き、書き込み、描画ではなくコマンドとして解釈されます。Windows 7 には、パン、ズーム、回転、マルチ タップ、プレス アンド タップといった新しいマルチタッチ ジェスチャーがあります。ジェスチャーのうち、最も高速かつ簡単に実行されるものにフリックがあります。"フリック" とは、ナビゲーションやコマンドの編集を行うための簡単なジェスチャーです。ナビゲーション フリックには、上へドラッグ、下へドラッグ、戻る、戻る、進むがあり、編集 フリックには、コピー、貼り付け、元に戻す、削除があります。

"操作" とは、オブジェクトのリアルタイムで物理的な処理のことです。操作がジェスチャーと異なるのは、現実世界の操作に対するオブジェクトの自然な反応に、入力が直接対応している点です。たとえば、写真閲覧アプリケーションでは、イメージを移動、ズーム、サイズ変更、および回転させて写真を操作できるようになっています。"マルチタッチ操作" では、同時に複数の接触ポイントを使用します。

細かい動きやホバーが必要なタスクの場合、Windows Vista® では "タッチ ポインター" を使用できます。タッチ ポインターは、マウスのような外観の画面上の移動ポインターです。タッチ ポインターは直接入力に比べると使いにくいため、タッチ可能なプログラムでタッチ ポインターを多用することは避けます。Windows 7 の既定では、タッチ ポインターは無効になっています。

Windows のタッチ機能のみがサポートされ、最も重要なタスクのコントロールが簡単にタッチできるプログラムは、"タッチ可能" と見なされます(爪を使わなければならないようなタスクは、タッチに適しているとはいえません)。タッチ可能なコントロールは、指を使った対象の指定が簡単に行えるよう十分な大きさでなければなりません)。実際には、次の場合が該当します。

- プログラムの対話型コントロールは、簡単にタッチできる十分な大きさ (23 × 23 ピクセルまたは 13 × 13 ダイアログ ユニット (DLU) 以上) です。
- フリック、マルチタッチ ジェスチャー、ドラッグ アンド ドロップなどの関連するシステム ジェスチャーが機能するように、プログラムでキーボードおよびマウスが適切にサポートされています。
- タスクにホバー操作やタッチ ポインターが必要ありません。
- すべてのコントロールで Microsoft Active Accessibility (MSAA) を使用し、支援テクノロジーにプログラムによる UI アクセスを提供しています。

プログラムが主なタスクでのタッチ用にデザインされている場合、"タッチ対応"と見なされます。通常は、次の場合が該当します。

- 使用頻度の最も高いコントロールが  $40 \times 40$  ピクセル ( $23 \times 22$  DLU) 以上です。
- 関連するジェスチャー(パン、ズーム、回転、マルチタップ、プレスアンドタップなど)がサポートされ、接触時にその効果が現れます。
- プログラムが、パン、ズーム、回転中に視覚的にスムーズに反応するため、対話性が高いと感じさせます。

プログラムが特にタッチ用にデザインされている場合、"タッチに最適化されている"と見なされます。通常は、次の場合が該当します。

- 最も頻繁に実行するコマンドを、ドロップダウンメニューではなく、直接 UI やコンテンツに表示することで、簡単にタッチできるようにタスクがデザインされています。
- プログラム専用のエクスペリエンスが、マルチタッチ操作を使ったり、弾みや抵抗などの現実世界の物理的な特性で反応するなど、(おそらく生のタッチ入力データを使用する)没入型のタッチエクスペリエンスになるようにデザインされています。
- タスクに寛容性があり、タッチやドラッグによって、ミスを簡単に修正したり、間違いを処理したりできます。
- 大量のテキスト入力や厳密な選択の必要性をなくしたり、軽減したりするようにタスクがデザインされています。

注: マウス、ペン、アクセシビリティに関するガイドラインは、それぞれ別の項目として記載しています。

## デザイン コンセプト

入力にタッチを使用することには次の特徴があります。

- 自然で直感的です。指でポイントしたり、物を触ったりする方法はだれでも理解できます。オブジェクトの対話操作は、現実の世界でユーザーがオブジェクトを操作する方法と同じになるように設計されています。
- それほど邪魔になりません。タッチは静かに使用できるので、特に会議のような社会的な状況ではタイピングやクリックほど気が散ることがありません。ペンに比べて、指は特に便利です。指を使用すれば、ペンを探したり、持ち上げたりする必要がないからです。
- 持ち運びできます。キーボード、マウス、タッチパッドなしでほとんどのタスクを実行できるので、タッチ対応のコンピューターの方がコンパクトです。操作領域を必要としないため、柔軟性に優れ、コンピューター使用の新しい場所やシナリオに対応できます。
- 直接的で魅力的です。タッチには、画面上のオブジェクトを直接操作する感覚があります。マウスやタッチパッドの場合は、手の動きをそれぞれの画面上のポインターの動きと連係させる必要があるため、間接的に感じます。
- 正確さが低下します。タッチでは、マウスやペンほどオブジェクトを正確に対象として指定できません。このため、ユーザーが小さなオブジェクトをタップしたり、操作したりすることを期待できません。

タッチでは、現実世界にいるような感覚で自然に操作できます。この感覚は、直接操作やアニメーションによって、オブジェクトが現実的で動的な動きや反応をすることによって得られます。たとえば、トランプゲームについて考えてみましょう。指を使用すると、トランプを引くことが便利で簡単になるだけではありません。まったく本物のトランプのように、トランプを配ったり、滑らせたり、回転させたり、跳ねさせたりできると、エクスペリエンスが魅力的な現実世界の感覚を帯びてきます。また、移動できないトランプを移動させようとした場合、操作は認識されたが、実行できないことを明確に示すために、トランプが動かないのではなく、動きに抵抗を示すようにし、トランプを放すと元の位置に収まるようにすると、エクスペリエンスが向上します。

## すべての Windows プログラムをタッチ可能にする

タッチは従来 Tablet PC に関連するものですが、通常のコンピューターでも一般的になります。Windows Tablet とタッチテクノロジーは Windows Vista および Windows 7 の標準コンポーネントであり、適切なハードウェアさえあれば、互換性のあるすべてのコンピューターでタッチを利用することができます。その結果、コンピューターの製造元により、通常のラップトップやデスクトップモニターにもタッチスクリーンが搭載されるようになりました。

タッチが Tablet PC 以外の種類のコンピューターにまで広がると、ソフトウェアプログラムの開発者や

デザイナーの間でも、タッチをサポートすることがますます重要と考えられるようになります。すべての Windows プログラムには優れたタッチ エクスペリエンスを持たせる必要があります。指を使用して、プログラムの最も重要なタスクを効率よく実行できる必要があります。タイピングや詳細なピクセル操作などタッチ操作には適していない一部のタスクであっても、タッチ可能にしておく必要はあります。

さいわいにも、プログラムが既に適切にデザインされていれば、優れたタッチ エクスペリエンスを実現することは簡単です。優れたタッチ エクスペリエンスを提供できるプログラムは、次のような点で適切にデザインされています。

- 適切なマウス サポート。対話型コントロールのアフォーダンスがはっきりと表示されます。標準のマウス操作(左シングルクリック、左ダブルクリック、右クリック、ホバー)に対して、オブジェクトに標準の動作が用意されています。
- 適切なキーボード サポート。標準のショートカット キー割り当てを提供し、特にナビゲーションや編集コマンドでショートカット キーを使用して(タッチ ジェスチャーを使用することもできます)作業の効率を向上させることができます。
- タッチに適した大きなコントロール。コントロールの最小サイズは  $23 \times 23$  ピクセル ( $13 \times 13$  DLU)、最もよく使用されるコントロールは  $40 \times 40$  ピクセル ( $23 \times 22$  DLU) 以上です。無応答をなくすために、ターゲットの間隔が狭くならないようにします。UI 要素の間隔は、隣のターゲットと接触しているか、5 ピクセル (3 DLU) 以上空いているかのどちらかにする必要があります。
- 必要に応じて、スムーズに応答するパンやズームを実行。イベントのパンやズームの速度に十分対応する速さで再描画されるため、ジェスチャー中に対話的な感覚がもたらされます。
- アクセシビリティ。Microsoft Active Accessibility (MSAA) を使用し、支援テクノロジーにプログラムによる UI アクセスを提供しています。プログラムがテーマやシステムのメトリクスの変更に適切に応答します。
- 120 dpi (ドット/インチ) ディスプレイで正常に機能し、適切に表示。120 dpi は、Windows Touch 対応コンピューターに推奨される既定の解像度です。
- コモン コントロールの使用。ほとんどのコモン コントロールは、適切なタッチ エクスペリエンスをサポートするようデザインされています。プログラムに必要な場合は、ターゲット指定や対話型操作を容易にするためにデザインされたカスタム コントロールが適切に実装されています。
- 有効な値の範囲のみ受け付けるコントロールの使用。簡単にタッチでターゲット指定できるデザインにするには、テキスト ボックスなどのコントロールよりも、リストやスライダーなどの選択範囲が限定されたコントロールの方が、テキスト入力の必要性が低下するので適しています。
- 適切な既定値の設定。最も安全(データの消失やシステム アクセスが失われることを防ぐため)で、最もセキュリティの高いオプションを既定で選択している。安全性とセキュリティを判断要素として考える必要がない場合は、最もよく使用されるオプションまたは最も便利なオプションを選択し、不要な対話操作を省いています。
- テキストのオートコンプリート機能。最もよく使用されるか最近使用された入力値の一覧を提供し、テキスト入力を簡易化しています。

逆に、プログラムが適切にデザインされていない場合、タッチを使用するユーザーに対して、プログラムの欠点が特に浮き彫りになります。

アクセシビリティ対応のソフトウェアはすべてのユーザーに役立つのと同じように、基本的な操作が簡単に実行でき、効率性、応答性、および寛容性に優れている場合、優れたタッチ エクスペリエンスの提供はすべてのユーザーに役立ちます。

#### タッチの対話操作モデル

タッチの使用経験がない場合は、実際に使用して理解するのが一番です。タッチ対応のコンピューターを使用し、マウスとキーボードを取りはずして、普段のタスクをタッチを使用して実行します。タブレット PC を使用する場合、膝の上などに乗せたり、平らなテーブルに置いたり、起立して腕で抱えて試します。縦や横にして使用してみてください。

タッチを使用してみると、次のようなことがわかります。

- 小さなコントロールは使いにくい。コントロールのサイズは対話操作の効率に大きく影響します。指で操作できるのは  $23 \times 23$  ピクセル ( $13 \times 13$  DLU) 以上のコントロールですが、快適に操作できるのは、それよりも大きい  $40 \times 40$  ピクセル ( $23 \times 22$  DLU) 以上のコントロールです。たとえば、スタートメニュー ( $42 \times 35$  ピクセル) は簡単にタッチできますが、[スピンドル](#) ( $15 \times 11$  ピクセル) は小さすぎて、指では操作できません。
- タスクを狭い範囲で実行できるようにする必要がある。画面上でポインターを 30 cm 移動する場合はマウスを 6

cm 動かすだけで済みますが、タッチを使用すると手を 30 cm 動かす必要があります。互いに離れているターゲットの間を何度も移動するのは手間がかかるため、タスクの対話操作をできる限り手に負担がかからない範囲に収めるのが適切です。コンテキストメニューは手の移動が必要ないので便利です。

- ホバーは必ずしも必要ではない。ほとんどのタッチスクリーンテクノロジーでは、ペンによるホバーは検出できても、指によるホバーは検出されません。ホバーに依存するタスクがある場合、タッチを使用して効率的に実行することはできません。
- テキストの入力と選択が難しい。タッチで長文のテキストを入力するのは特に難しいため、オートコンプリート機能や許容できるテキストの既定値があるとタスクが大幅に簡単になります。テキストの選択も非常に難しいため、カーソルを正確に配置する必要がないとタスクが簡単になります。
- ディスプレイの縁端付近にある小さなターゲットをタッチするのは非常に難しい。一部のディスプレイではベゼル幅が広く、一部のタッチスクリーン技術では縁端部の感度が低いため、縁端付近にあるコントロールが使いにくくなっています。たとえば、ウィンドウを最大化すると、タイトルバーにある最小化、最大化、元のサイズに戻す、閉じるの各ボタンが使いにくくなります。

ここではさまざまな課題がありますが、それらに対処することですべてのユーザーのエクスペリエンスが向上します。

### 基本的なタッチのデザイン原則

それぞれの入力デバイスには、長所もあれば短所もあります。キーボードは、テキスト入力や最小限の手の動きでコマンドを指定するのに最適です。マウスは、効率的で正確なポイントに最適です。タッチは、オブジェクトの操作や簡単なコマンドの指定に最適です。ペンは、手書きや描画などのフリー形式の表現に最適です。

プログラムでのタッチのサポートについて考える場合、次の点に留意します。

- UI がマウスで適切に動作する場合、タッチでも同様だと思い込まない。適切なマウスのサポートは第 1 条件ですが、優れたタッチ エクスペリエンスにはさらにいくつか要件があります。
- UI が指で適切に動作する場合、ペンでも同様と考えることはできる。プログラムをタッチ可能にすることは、ペ็นを適切にサポートするのに大いに役立ちます。主な違いは、指は正確にポイントできないため、より大きな対象が必要になることです。また、ホバーは省略可能にする必要があります。ペン入力のサポートに関するガイドラインについては、「[ペン](#)」を参照してください。
- タッチの UI に関する問題を修正するために、タッチ ポインターに依存しない。タッチ ポインターは直接入力ほど簡単に使用できないため、タッチ ポインターは、タッチ用にデザインされていないプログラムの最後の手段と考えます。

### コントロールのサイズ

フィットの法則では、対象の操作に必要な時間は、対象のサイズとそこまでの距離によって変わります。対象が小さく、距離が離れているほど、ますます操作しにくくなります。しかし、指先の表面の領域が広いために、密接した小さなコントロールを正確に対象指定することが難しくなることもあります。

一般的なルールとしては、 $23 \times 23$  ピクセル ( $13 \times 13$  DLU) のコントロール サイズは、すべての入力デバイスに最適な対話型コントロールの最小サイズです。それに対して、 $15 \times 11$  ピクセルのスピニコントロールは、タッチで効果的に操作するには小さすぎます。

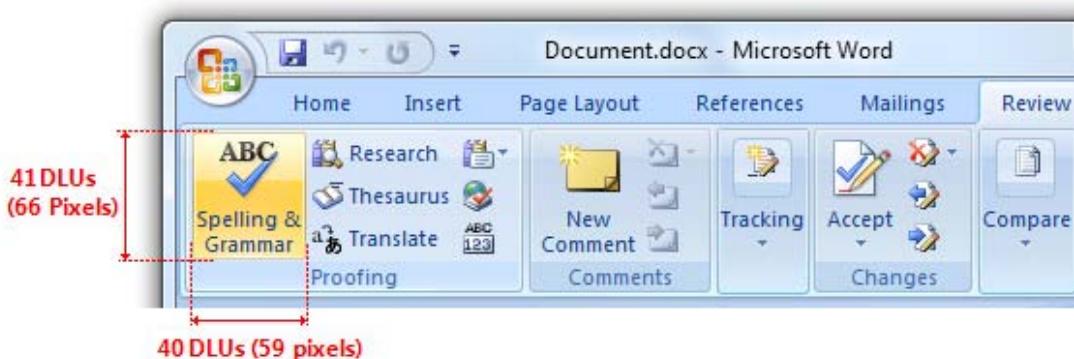


スピニコントロールは、タッチするには小さすぎます。

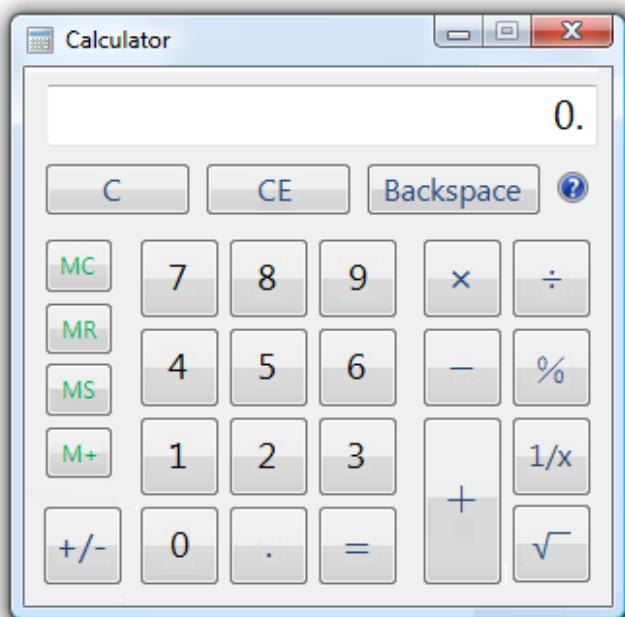
最小サイズは、実際に、ピクセルや DLU などのレイアウト メトリックではなく、物理的領域に基づいています。研究により、指を使用して効率的で正確な操作を行うための最小の対象領域は  $6 \times 6$  mm であることがわかっています。この領域は、次のようなレイアウト メトリックで表されます。

フォント	サイズ (mm)	相対ピクセル	DLU
9 ポイント、 Segoe UI	$6 \times 6$	$23 \times 23$	$13 \times 13$
8 ポイント、 Tahoma	$6 \times 6$	$23 \times 23$	$15 \times 14$

さらに、 $10 \times 10 \text{ mm}$  (約  $40 \times 40$  ピクセル) の最小サイズでは、速度や正確さが向上し、またユーザーがより快適に感じるがことが調査でわかっています。最も重要なコマンドや使用頻度が最も高いコマンドに使用する場合、通常はこの大きいサイズのコマンド ボタンを使用します。



この例では、Microsoft Word で、最も重要なコマンドに  $10 \times 10 \text{ mm}$  より大きいボタンを使用しています。



この電卓のバージョンでは、使用頻度も最も高いコマンドに  $10 \times 10 \text{ mm}$  より大きいボタンを使用しています。

目的は、大きなコントロールを作成することではありません。必要なのは、簡単にタッチできるコントロールにすることだけです。この記事の後半のガイドラインセクションに一覧表示された手法を使用すると、コントロールを過度に大きく見せることなく、簡単にタッチ可能にすることができます。

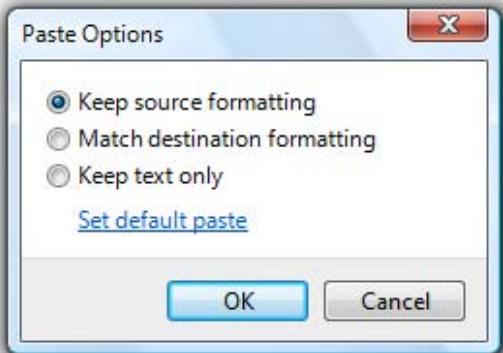
注: コントロールのサイズを指定する場合、96 dpi を対象とします。解像度を上げると、ユーザーのタッチ操作が向上します。

#### コントロールの間隔

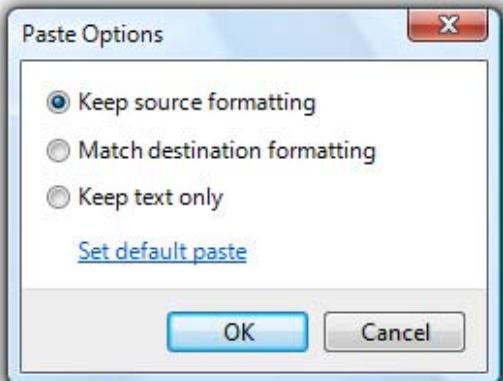
コントロール間の間隔も、コントロールを簡単にタッチ可能にする決定要因です。指をポインティングデバイスとして使用する場合、対象指定はすばやくできても、正確さが劣るため、目的の対象から外れた場所をタップすることがよくあります。対話型コントロールが非常に近くに配置されているものの実際には接触していない場合、ユーザーがそれらのコントロールの間の非アクティブな領域をクリックすることができます。非アクティブな領域をクリックしても効果や視覚的なフィードバックが得られないで、多くの場合ユーザーは何が悪かったのかがわかりません。小さなコントロールが狭すぎる間隔で配置されていると、間違って別のオブジェクトをタップしないように、ユーザーは正確にタップしなければならなくなります。これらの問題に対処するには、対話型コントロールのターゲット領域の間隔は、隣のコントロールのターゲット領域と接触しているか、5 ピクセル (3 DLU) 以上空いているかのどちらかにする必要があります。

コントロール間の垂直方向の間隔を推奨されるサイズより広げると、グループ内のコントロールを区別しやすくなります。たとえば、高さが 19 ピクセルのラジオ ボタンは、推奨される最小サイズの 23 ピクセルよりも低くなっています。垂直方向の領域が空いている場合、標準の 7 ピクセルに 4 ピクセルの間隔を追加することで、推奨サイズとだいたい同じ効果が得られます。

正しい例:



より良い例:



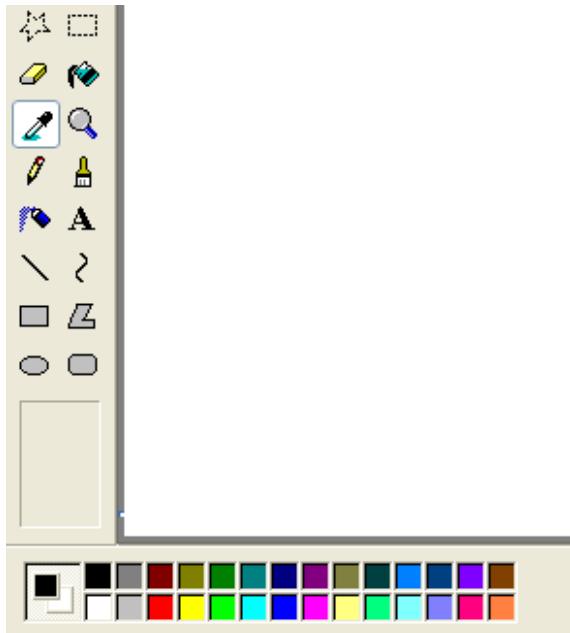
より良い例では、間隔を広げることで、ラジオ ボタンが区別しやすくなっています。

間隔を広げることは、タッチを使用する場合は望ましいが、マウスやキーボードを使用する場合には必要ないという状況もあります。このような状況では、タッチを使用して開始する操作では、間隔の広いデザインのみを使用する必要があります。

コントロールの位置

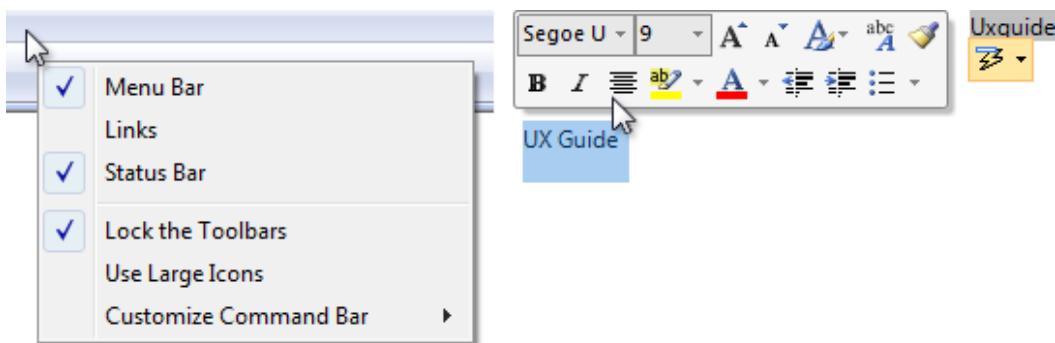
タスクの配置を狭い範囲に收めると、画面を何度も往復する手間が減ります。手の移動をできるだけ少なくするには、最も使用されそうな場所から近いところにコントロールを配置します。

間違った例:



Windows XP のこの例では、カラー パレットが使用されると思われる場所から離れすぎています。

ユーザーの現在の位置からターゲットまでが最も近くなるようにすると、簡単に使用できるようになります。以下に示すように、コンテキストメニューでは Microsoft Office のミニ ツール バーやスマート タグと同じく、フィックスの法則が最大限活用されています。



現在のポインターの位置は常に、最も簡単に使用できる場所にあります。

ディスプレイ縁端付近のターゲットが小さいとタッチが難しいため、小さなコントロールをウィンドウ縁端付近に配置しないようにします。ウィンドウが最大化されているときでも簡単にコントロールをポイントできるようにするには、コントロールのサイズを 23 × 23 ピクセル (13 × 13 DLU) 以上にするか、ウィンドウ縁端部から離れたところにコントロールを配置します。

#### タッチの対話操作

##### システム ジェスチャー

システム ジェスチャーは Windows によって定義され処理されます。このため、すべての Windows プログラムでシステム ジェスチャーを使用できます。これらのジェスチャーには、マウス、キーボード、およびアプリケーションに対応するコマンド メッセージがあります。

システム ジェスチャー	生成される対応メッセージ
ホバー (サポートされている場合)	マウスでのホバー
タップ (軽くたたく)	マウスでの左クリック
ダブルタップ ("軽くたたく" を 2 回)	マウスでの左ダブルクリック
プレス アンド ホールド (押して一時停止し、上げる)	マウスでの右クリック
ドラッグ (押して移動し、上げる)	マウスでの左ドラッグ

プレス アンド ホールド アンド ドラッグ (押して一時停止したのち移動し、上げる)	マウスでの右ドラッグ
選択 (押して選択可能オブジェクト上を移動したのち、上げる)	マウスでの選択

開発者向け情報: 詳細については、[SystemGesture の列挙に関するページ](#)を参照してください。

## フリック

フリックとは簡単なジェスチャーであり、キーボードショートカットとほぼ同等です。ナビゲーションフリックには、上へドラッグ、下へドラッグ、戻る、進むがあり、編集フリックには、コピー、貼り付け、元に戻す、削除があります。フリックを使用するには、プログラムが関連するキーストロークのコマンドに対応していることだけが必要です。または、プログラムでイベントを直接処理できれば問題ありません。



これは、Windows 7 の 8 個のフリック ジェスチャーとその既定の割り当てです。ナビゲーションフリックは、スクロール (オブジェクトがジェスチャーとは反対の方向に移動) ではなく、パン (オブジェクトがジェスチャーに一致する方向に移動) に対応するように変更されています。



これは、Windows Vista の 8 個のフリック ジェスチャーとその既定の割り当てです。

ナビゲーションフリックでは、学習や記憶が容易になるように自然な割り当てが行われています。編集フリックは高い精度が必要な斜め方向への移動で、割り当てはあまり自然ではないため ([ごみ箱] の方向にフリックすると削除され、[戻る] 矢印の方向にフリックすると元に戻す処理が行われます)、既定では有効にされていません。すべてのフリック操作は、コントロールパネル項目のペンと入力デバイスを使用してカスタマイズすることができます。

フリック	生成される対応メッセージ
左へフリック	[進む] コマンド (Windows Vista では [戻る] コマンド)
右へフリック	[戻る] コマンド (Windows Vista では [進む] コマンド)
上へフリック	キーボードでの下へスクロール
下へフリック	キーボードでの上へスクロール
左上へフリック	キーボードでの削除
左下へフリック	キーボードでの元に戻す
右上へフリック	キーボードでのコピー
右下へフリック	キーボードでの貼り付け

## アプリケーション ジェスチャー

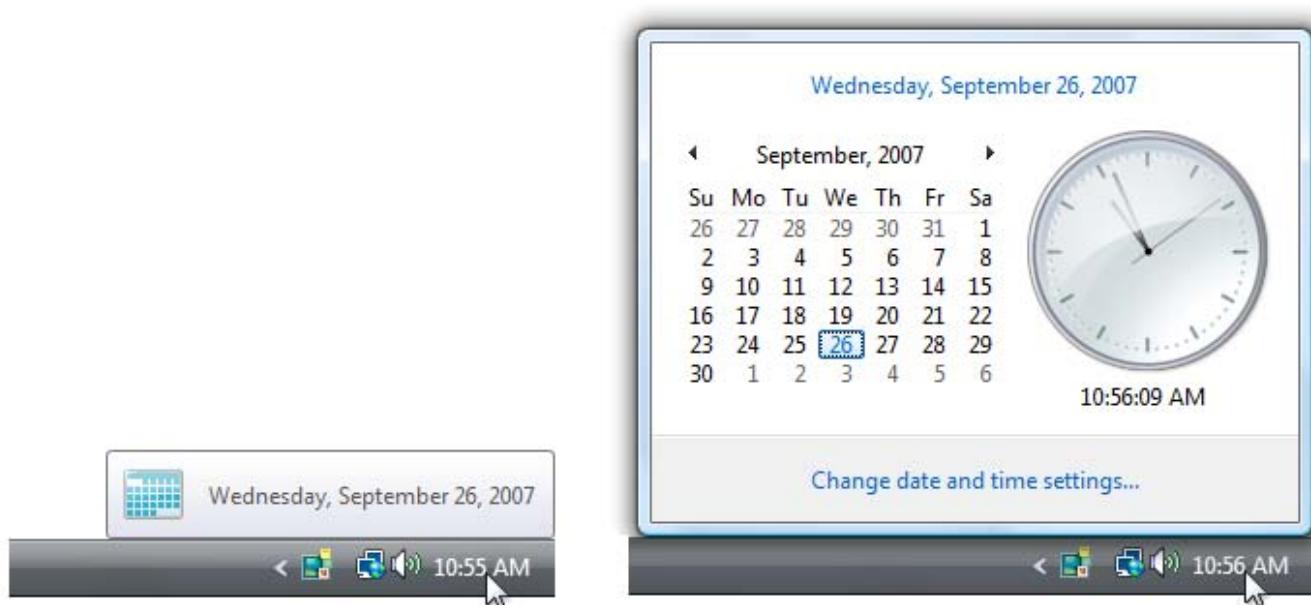
アプリケーションでは、他のジェスチャーも定義し処理することができます。Microsoft Gesture Recognizer では、[40 ジェスチャー](#)以上認識することができます。アプリケーション ジェスチャーを使

用するには、プログラムで認識するジェスチャーを定義し、発生したイベントを処理する必要があります。

## ホバー

ホバーは、操作を開始する前にヒントによって追加情報を得られるため、便利な操作です。ヒントによって追加情報を得ることで、ユーザーは自信が付き、ミスが少なくなります。

残念なことに、ホバーはタッチ テクノロジーではサポートされていないため、指を使ってホバーを行うことはできません。この問題の簡単な解決策は、ホバーを十分に活用することですが、それは操作を実行する必要がない場合に限ります。これは、通常、実際にはこの操作はクリックでも実行でき、必ずしもまったく同じ方法で実行するとは限らないということです。



この例では、今日の日付をホバー、クリックのいずれかで確認できます。

## 応答性と一貫性

応答性は、直接的で魅力的なタッチ エクスペリエンスを作り出すのに不可欠です。直接的な感じを与えるには、ジェスチャーが即座に効果を現し、ジェスチャーの間、ユーザーの指でオブジェクトの接点が滑らかに保たれることが必要です。操作の効果は、ユーザーの動きに直接対応している必要があります。たとえば、ユーザーが指を 90°回転させると、オブジェクトも 90°回転するようにします。応答が遅れる、安定しない、接点が失われる、正確に応答しないなどが起こると、直接操作の感覚も魅力もなくなります。

一貫性は、自然で直感的なタッチ エクスペリエンスを作り出すのに不可欠です。ユーザーが標準のジェスチャーを習得した場合、そのジェスチャーがすべてのアプリケーション プログラムで同じ効果を生むと考えます。混乱や不満をなくすため、標準のジェスチャーには非標準の意味を割り当てないでください。プログラムに特有の対話操作にはカスタム ジェスチャーを使用します。

## 寛容性

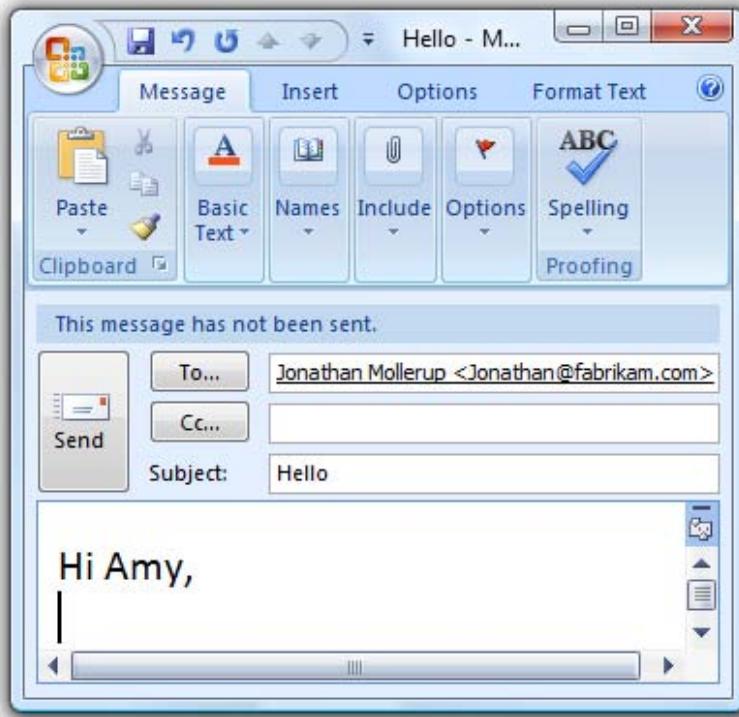
タッチが自然で、表現力豊かで、効率的で、魅力的なものである要因は、その直接性にあります。実際、タッチ操作は直接操作と言われることがよくあります。ただし、直接操作を行う場面では、操作ミスが起こる可能性があり、そのために寛容性が必要になります。

"寛容性" とは、好ましくない操作を簡単に元に戻したり、修正したりできることです。タッチ エクスペリエンスで寛容性を確保するには、元に戻す操作を行う、適切な視覚的フィードバックを行う、使用頻度の高いコマンドと [リスクのあるコマンド](#)との間の物理的な区切りをはっきりさせる、ユーザーがミスを簡単に修正できるようにする、といった方法があります。寛容性に関連して、好ましくない操作が実行されることを最初の段階で防ぐこともできます。これは、意図しない結果を招く可能性がある危険な操作やコマンドに対して、制限付きのコントロールと確認を使用することで実現できます。

## テキストの編集

テキストの編集は、指を使用する場合、最も困難な操作の1つです。選択範囲が限定されたコントロール、適切な既定値、オートコンプリート機能を使用すると、テキスト入力の必要性がなくなるか減ります。ただし、プログラムにテキストの編集が含まれる場合、タッチ使用時に既定で自動的に入力 UI を最大 150% 拡大すると、ユーザーの生産性を上げることができます。

たとえば電子メール プログラムでは標準のタッチ可能なサイズで UI を表示しますが、メッセージ作成時に入力 UI を 150% に拡大することができます。



この例では、入力 UI が 150% に拡大されています。

## 6つの重要な点

1. Windows プログラムに適切なタッチ エクスペリエンスが備わっているようにします。ユーザーが指を使用し、プログラムの最も重要なタスクを効率よく実行できることが必要です(最低でも、そうしたタスクで多量のタイピングや精緻なピクセル操作を必要としないようにします)。
2. コモンコントロールについては、[標準的なコントロールのサイズ](#)を使用します。その他のコントロールでは、静的な外観が非常に小さい場合でも、クリック対象領域が 23 × 23 ピクセル (13 × 13 DLU) 以上になるようにします。
3. ホバーを使用しますが、これが操作を実行する唯一の方法にならないようにします。ホバーは、ほとんどのタッチスクリーン テクノロジーでサポートされていません。
4. 直接的で魅力的なエクスペリエンスを作り出すには、ジェスチャーが即座に反映され、ジェスチャーの間、ユーザーの指で接点が滑らかに保たれ、ジェスチャーの効果が直接ユーザー動作に割り当てられるようにします。
5. 自然で直感的なエクスペリエンスを作り出すには、適切な標準ジェスチャーをサポートし、それぞれの標準の意味を割り当てます。プログラムに特有の対話操作にはカスタム ジェスチャーを使用します。
6. プログラムで、特にリスクのあるコマンドなどの好ましくない操作を元に戻したり、修正したりできるようにします。タッチ操作では、操作ミスが起こる可能性が高くなります。

## ガイドライン

### コントロールの使い方

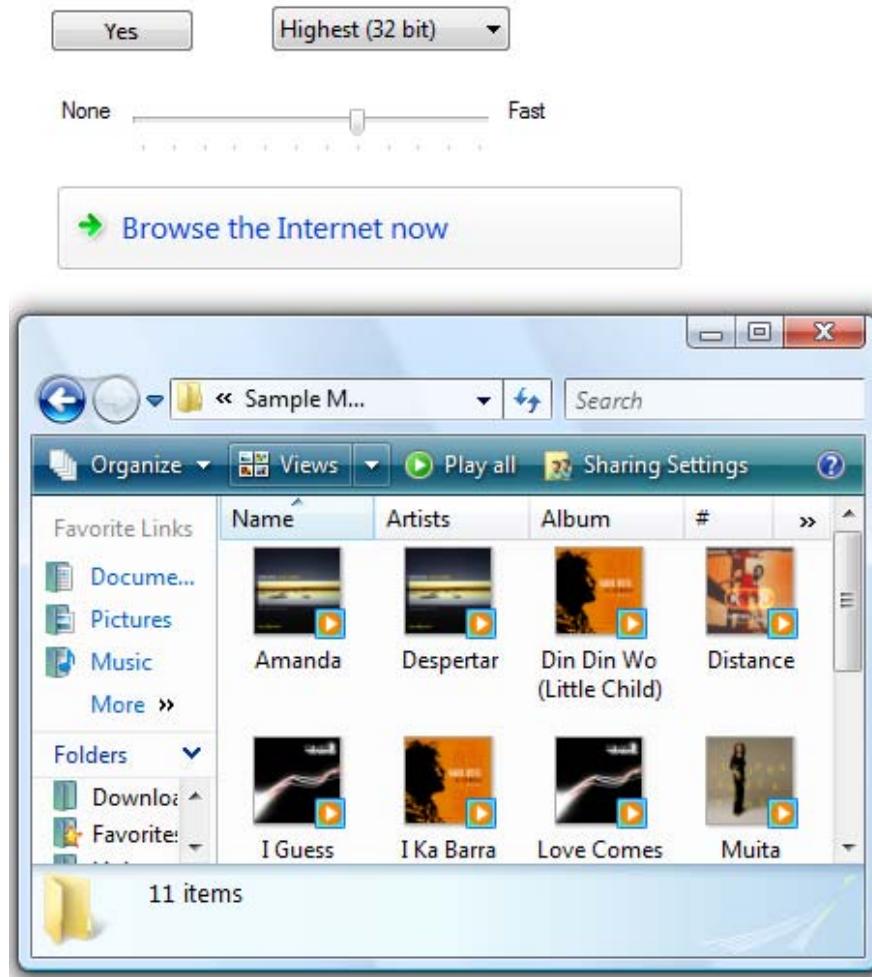
- コモンコントロールを優先して使用します。ほとんどのコモンコントロールは、適切なタッチ エクスペリエンスをサポートするようデザインされています。
- タッチをサポートするようにデザインされたカスタムコントロールを選択します。プログラムに特有のエクスペリエンスをサポートするカスタムコントロールが必要になることがあります。次のようなカスタムコントロールを選択します。

簡単にタッチできる十分な大きさのサイズに設定できる。

- 弾みや抵抗などを付けることによって、操作時に現実世界のオブジェクトのように移動したり反応したりする。
- ユーザーが簡単にミスを修正できるようにして寛容性を確保している。
- クリックやドラッグによる間違いの寛容性を確保している。移動先の近くにドロップされたオブジェクトが正しい位置にドロップされるようにします。
- 指がコントロールの上に重なっている場合でも、波及効果のようにはっきりとフィードバックが見える。
- できるだけ制約付きコントロールを使用します。テキスト入力の負担を減らすために、テキストボックスのような制約のないコントロールではなく、リストやスライダーといった制約付きコントロールをできる限り使用します。
- 適切な既定値を設定します。最も安全(データの消失やシステムアクセスが失われることを防ぐため)で、最もセキュリティの高いオプションを既定で選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用されるオプションまたは最も便利なオプションを選択し、不要な対話操作を省きます。
- テキストのオートコンプリート機能を実装します。最もよく使用されるか最近使用された入力値の一覧を提供し、テキスト入力を簡易化します。
- 複数の選択項目を使用する重要なタスクについては、[標準的な複数選択リスト](#)が通常使用される場合は代わりに[チェックボックスリスト](#)を使用するオプションを提供します。

#### コントロールのサイズ

- コモンコントロールについては、[コントロールに推奨されるサイズ](#)を使用します。推奨されるコントロールのサイズは、23 × 23 ピクセル(13 × 13 DLU)の最小サイズ以上です。ただし、チェックボックスやラジオボタン(テキストボックスの幅でいくらかサイズが埋まります)、スピンドルコントロール(タッチでは操作できませんが、冗長になっています)、およびスプリッターを除きます。



推奨されるコントロールのサイズは簡単にタッチできます。

- 最も重要なコマンドや最も使用頻度の高いコマンドに使用するコマンド ボタンでは、実際に使用する場合、必ず最小サイズ  $40 \times 40$  ピクセル ( $23 \times 22$  DLU) を使用します。そうすることで速度や正確さが向上し、ユーザーがより快適に感じるようになります。



実際に使用する場合、重要なコマンドや使用頻度の高いコマンドには大きめのコマンド ボタンを使用します。

- その他のコントロールの場合、次の点に留意します。

- 大きめのクリック対象を使用する。小さいコントロールでは、対象のサイズを静的に表示されている UI 要素よりも大きくなります。たとえば、 $16 \times 16$  ピクセルのアイコン ボタンには、 $23 \times 23$  ピクセルのクリック対象ボタンを設定し、テキスト要素には、テキストよりも幅が 8 ピクセル、高さが 23 ピクセル大きい選択範囲を設定できます。

正しい例:



間違った例:



正しい例:



正しい例では、クリック対象が静的に表示されている UI 要素よりも大きくなっています。

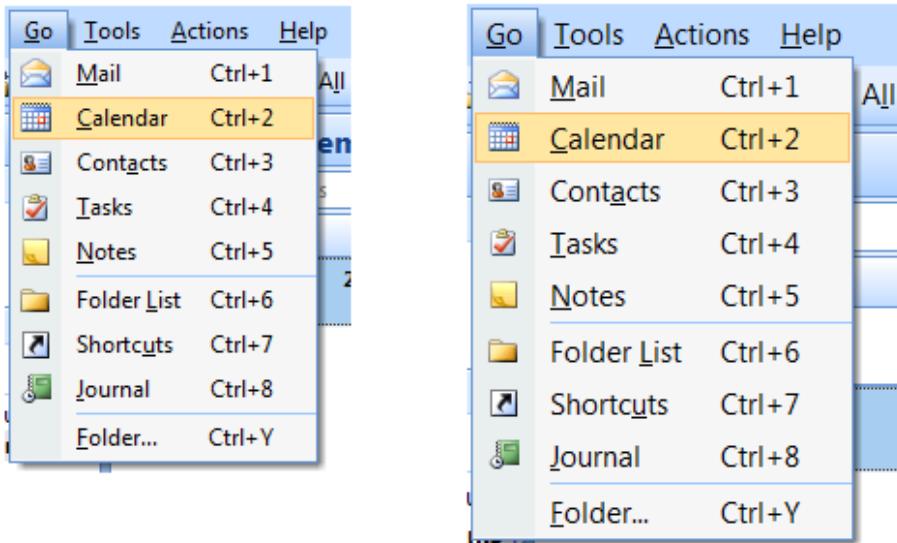
- 冗長なクリック対象を使用する。コントロールに冗長な機能がある場合、クリック対象が最小サイズよりも小さくなってもかまいません。

たとえば、ツリービューコントロールで使用される段階的表示の三角形は  $6 \times 9$  ピクセルしかありませんが、その機能は関連するアイテム ラベルと重複しています。



ツリービューの三角形は小さすぎて簡単にタッチできませんが、関連する大きなラベルと機能において冗長になっています。

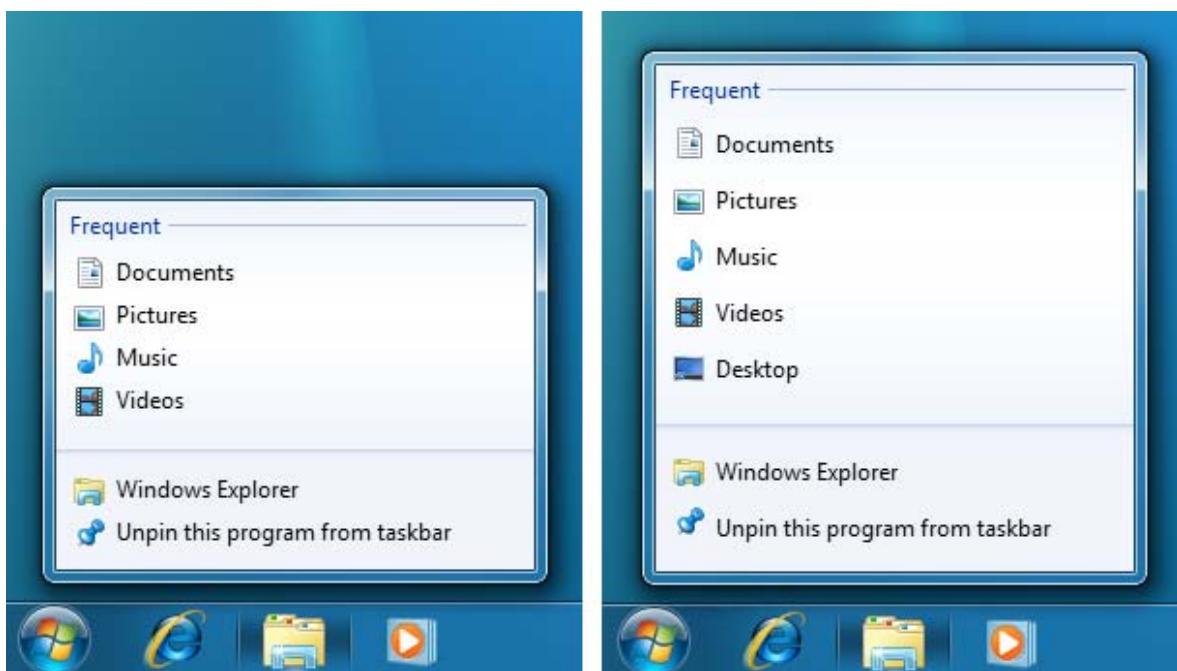
- システムのメトリクスを尊重します。すべてのサイズでシステムのメトリクスを使用し、ハードウェア組み込みのサイズは使用しません。必要な場合は、ユーザーがシステムのメトリクスまたは dpi を適宜変更することができます。ただし、通常はユーザー側でシステム設定を調整して UI を使用可能にすることは望ましくないので、これは最終的な手段にします。



この例ではメニューの高さに関して、システムのメトリクスが変更されています。

#### コントロールのレイアウトと間隔

- 最も使用されると思われる場所に近いところにコントロールを配置するレイアウトを選択します。できる限り狭い範囲にタスクの対話操作を収めます。コモンタスクとドロップでは特に、手の移動距離が長くならないようにします。
- 推奨される間隔**を使用します。推奨される間隔はタッチに適しています。
- 対話型コントロールの間隔は、隣のコントロールと接触しているか、5ピクセル(3 DLU)以上空いているかのどちらかにする必要があります。そうすることで、ユーザーが目的のターゲットの外側をタッチしても混乱しません。
- コマンドリンク、チェックボックス、ラジオボタンなどのコントロールのグループ内およびグループ間では、推奨される上下の間隔よりも広く間隔を空けます。そうすることで、差異化しやすくなります。
- タッチを使って操作を開始する場合、垂直方向の推奨される間隔を動的に広げることを検討します。間隔を広げることで、オブジェクトを区別しやすくなりますが、キーボードやマウスを使用する場合、間隔を広げなくても簡単に区別できます。間隔は、通常サイズの3分の1または8ピクセル以上追加します。



この例では、Windows 7 のタスクバーのジャンプリストは、タッチを使って表示するときに間隔が広がります。

#### 対話操作

- ホバーを唯一の操作にしません。ホバーを利用しますが、操作を実行する必要がない場合に限ります。これは、通常、実際にはこの操作はクリックでも実行でき、必ずしもまったく同じ方法で実行するとは限らないということです。ホバーはほとんどのタッチテクノロジーではサポートされていないため、そうしたタッチスクリーンを使用しているユーザーは、ホバーを必要とするタスクを実行できません。
- テキスト入力が必要なプログラムでは、次の方法で、タッチキーボード機能を完全に統合します。
  - ユーザー入力のための適切な既定値を用意する。
  - 必要に応じて、オートコンプリートの候補を用意する。

**開発者向け情報:** タッチキーボードの統合の詳細については、[ITextInputPanel インターフェイスに関するページ](#)を参照してください。

- プログラムにテキスト編集を必要とするタスクがある場合は、ユーザーがコンテンツ UI を拡大できるようにします。タッチ使用時は自動的に 150% に拡大されるようにします。
- 必要に応じて、スムーズに応答するパンやズームを実行します。応答し続けるために、パンやズームの後にしばらく再描画します。これは、直接操作が真に直接的な感覚をもたらすために必要です。
- パンまたはズーム中、ジェスチャー全体にわたって接触ポイントが指の下から放れないようにします。そうしないと、パンやズームを制御しにくくなります。
- ジェスチャーは記憶されるため、プログラム全体で一貫して意味を割り当てます。確定した意味を持つジェスチャーに別の意味を割り当てないでください。適切なプログラム特有のジェスチャーを使用します。

## Windows タッチ ジェスチャー

次のジェスチャーをプログラムで使用できる場合、必ず使用します。これらは最も便利で自然なジェスチャーです。

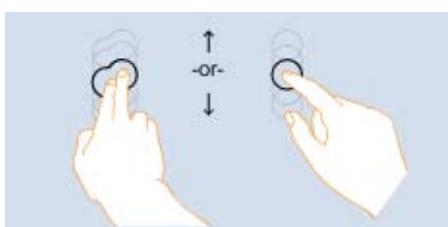
### • パン

入力状態: 1 本または 2 本の指で画面に触れます。

動作: 互いの指が同じ位置をキープしたままドラッグします。

終了状態: 最後の指を画面から放すと、ジェスチャーが終了します。

効果: 指が動くと、直ちに下のオブジェクトが移動します。ジェスチャー全体にわたって、接触ポイントが指の下から離れないようにします。



パンのジェスチャー。

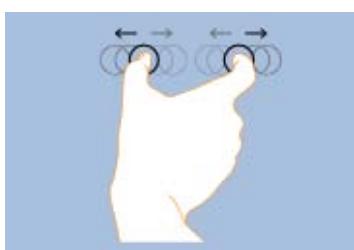
### • ズーム

入力状態: 同時に 2 本の指で画面に触れます。

動作: 2 本の指を軸に沿って引き離すか、合わせ(つまみ)ます。

終了状態: 指を画面から放してジェスチャーを終了させるか、指で軸を中断します。

効果: 指が軸から離れたり、軸に近づいたりすると、直ちに下のオブジェクトが拡大したり、縮小したりします。ジェスチャー全体にわたって、接触ポイントが指の下から離れないようにします。



ズームのジェスチャー。

綿密にアニメーション化を行う場合、パン中にズームできるようになると、効果的かつ効率的に操作できます。

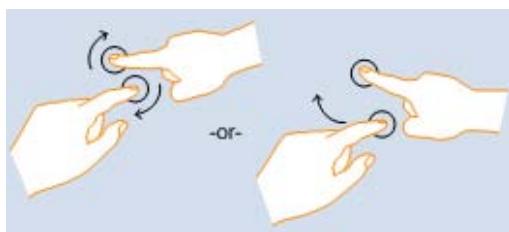
### • 回転

入力状態: 同時に 2 本の指で画面に触れます。

動作: 1本または2本の指を、2本の指をつなぐ直線に対して垂直に動かし、他方を軸にして回転させます。

終了状態: 指を画面から放すと、ジェスチャーが終了します。

効果: 指の回転と同じだけ下のオブジェクトが回転します。ジェスチャー全体にわたって、接触ポイントが指の下から離れないようにします。



回転のジェスチャー。

回転は、一定の種類のオブジェクトでのみ有効であるため、Windows システムの操作には対応しません。

回転は、ユーザーによってさまざまな動作になることがあります。軸指の周りで指を回転させる場合もあれば、両方の指を円を描くように回転させる場合もあります。ほとんどの場合、2本を組み合わせて、1本の指を他方よりも動かして回転させます。任意の角度へのスムーズな回転は、写真の閲覧などの多くのコンテンツで最も効果的な操作ですが、ユーザーが手を放すと、90°回転した状態に落ち着くのが最適な操作です。写真の編集では、写真のゆがみを修正するために小さな回転を使用できます。

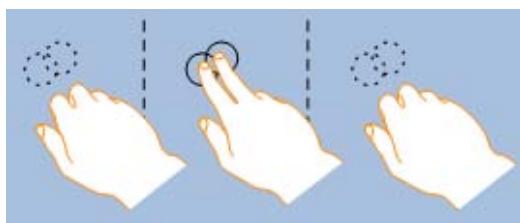
- マルチ タップ

入力状態: 同時に2本の指で画面に触れます。

動作: 動作はありません。

終了状態: 指を画面から放すと、ジェスチャーが終了します。

効果: 指の間にあるオブジェクトの既定の表示をズームしたり、復元したりすることもできます。



マルチ タップのジェスチャー。

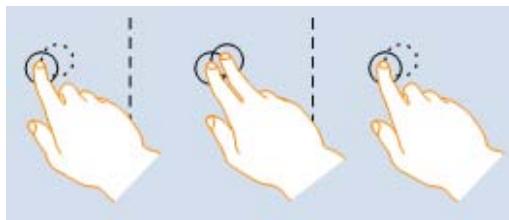
- プレス アンド タップ

入力状態: 1本の指で画面に触れ、その後2本目の指で触れます。

動作: 動作はありません。

終了状態: 2本目の指を画面から放すと、ジェスチャーが終了します。

効果: 最初の指の下にあるオブジェクトの右クリックを実行します。



プレス アンド タップのジェスチャー。

## 寛容性

- [元に戻す] コマンドを用意します。すべてのコマンドに元に戻す簡単な方法を用意するのが理想的ですが、プログラムにはコマンドの効果を元に戻すことができないものが含まれていることがあります。
- 実用的な場合は必ず、指が触れていることの適切なフィードバックを提供しますが、指が離れるまでは操作が反映されないようにします。こうすると、ユーザーは事前に間違いを修正できます。
- 実用的な場合は必ず、ユーザーが簡単に間違いを修正できるようにします。指を離したときに操作が反映される場合は、指が触れている間にスライドによって間違いを修正できるようにします。
- 実際の操作では、動作に抵抗を示すことで直接操作が実行できないことを示します。動作が起こるようにして

も、オブジェクトを放すと元の位置に戻るようにして、操作が認識されても実行できないということを明確に示します。

- 頻繁に使用されるコマンドとリスクのあるコマンドを、物理的にはっきり見分けられるようにします。そうでない場合、リスクのあるコマンドを誤ってタッチするおそれがあります。その結果が広範囲におよび、簡単に元に戻せないか、結果がすぐには表れない場合は、リスクのあるコマンドと見なされます。
- [リスクのある操作](#)のコマンドまたは[意図しない結果](#)をもたらすコマンドを確認します。このためには、確認ダイアログ ボックスを使用します。
- タッチを使ったときに誤って実行しがちで、気付かないままになるか、元に戻すのが難しい他の操作を確認することを検討します。通常、これを[日常的な操作に対する確認](#)といいますが、このようなコマンドはマウスやキーボードでは誤って発行されることがあまりないことを想定し、推奨されていません。不要な確認を行わないようにするには、タッチを使ってコマンドが開始された場合にのみ、確認ダイアログ ボックスを表示します。



日常的な操作に対する確認は、ユーザーがタッチを使ってよく誤って実行する操作で利用できます。

開発者向け情報: マウス イベントとタッチ イベントは、[GetMessageExtraInfo](#) API を使用して判別できます。

## ドキュメント

タッチに言及するときは、以下のことに留意します。

- 入力デバイスとして使用する場合のユーザーの手は、1本または複数本の "指" とします。
- キーボード、マウス、トラックボール、ペン、指を総称して "入力デバイス" と呼びます。
- 指またはペンの使用に特定した手順を記す場合は、"クリック" ではなく "タップ" (および "ダブルタップ") を使用します。タップとは、画面を押し、ホールド タイム前に持ち上げることを意味します。マウス クリックの生成に使用することが許可されることもあれば、禁止されることもあります。指またはペンを使用しない対話操作の場合は、引き続き "クリック" を使用します。
- "タッチスクリーン" と "タッチパッド" は、単一の複合語です。"タッチスクリーン" や "タッチパッド" としないでください。

# ペン

すべての Microsoft® Windows® アプリケーションはペン対応にする必要があります。これは、思いのほか簡単に実現できます。

デザインコンセプト

ガイドライン

コントロールの使い方

コントロールのサイズ、レイアウト、間隔

対話操作

利き手

寛容性

ドキュメント

"ペン入力" とは、Windows で実現されている、ペンを使用してコンピューターと直接対話する方法です。ペンはポイントとジェスチャー、単純なテキスト入力に使用するほか、デジタルインクを使用して自由に考えを記入することができます。

入力に使用する "ペン" の先端は滑らかで微細であるため、ポイント、書き込み、インクによる描画を高精細に行うことができます。ペンには、オプションで "ペンボタン" (右クリック用) と "消しゴム" (インク消去用) が備わっていることがあります。ほとんどのペンでは、ホバー機能がサポートされています。



典型的なペン

ペンを手書きに使用する場合、"手書き認識" を使用してユーザーのストロークをテキストに変換することができます。ストロークは実際に書き込まれた状態を保持でき、バックグラウンドで実行される認識では、テキストとしての検索機能とコピー機能がサポートされます。このような無変換のストロークは、デジタル "インク" と呼ばれます。



## インク入力の例

ほとんどの Windows プログラムは、マウスの代わりにペンを使用でき、ほとんどの重要なタスクと対話操作で円滑に機能します。また、プログラムがジェスチャーに応答するという意味では既に "ペンに対応" しています。プログラムで手書きのテキスト入力が支援されている場合は、"手書きに対応" したプログラムになります。ペンストロークを同等のテキストまたはマウス動作に変換する必要がなく、インクを直接扱うことができるようになると、"インク対応" になります。これにより、ユーザーは高品質のデジタルインクを使用して自由にコメントを書き込んだり、描画、追加できるようになります。インクの方がマウスよりも高い解像度とサンプルレートが必要であり、また、インクには筆圧や傾きの微妙な違いも加わるため、インクの取得はマウスイベントの取得とは異なります。手書きに適したプログラムとインク対応のプログラムの作成方法については、[インクの統合に関するページ](#)および[ペンを使用したテキスト入力に関するページ](#)を参照してください。

ペンの位置を特定するとき、ペンの先端自体がポイント先なのでカーソルはあまり必要ありません。ただし Windows では、ターゲット指定を支援するために小さな "ペン カーソル" を用意し、現在のペンの位置を示しています。ペン カーソルはマウスポインターとは異なり、ペンがディスプレイに接近している場合を除いては必要ないため、非アクティブな状態が数秒続いた後に非表示になり、情報がよく見えるようになります。

ペンに適したほとんどのプログラムでは、ジェスチャーがサポートされています。"ジェスチャー" とは、画面上で行われるペンのすばやい動きです。コンピューターではこの操作がマウスの動き、書き込み、描画ではなくコマンドとして解釈されます。ジェスチャーのうち、最も高速かつ簡単に実行されるものにフリックがあります。"フリック" とは、ナビゲーションやコマンドの編集を行うための簡単なジェスチャーです。ナビゲーションフリックには、上へドラッグ、下へドラッグ、戻る、進むがあり、編集フリックには、コピー、貼り付け、元に戻す、削除があります。

ビギー ポインターを除くすべてのポインターには単一ピクセルの "ホットスポット" があり、これによってポインターの画面上の正確な位置が定められます。ホットスポットによって、対話操作の影響を受けるオブジェクトが決まります。ホットスポットがオブジェクト上にあると判断する領域を "ホットゾーン" といい、各オブジェクトに定義されています。通常、ホットゾーンはオブジェクトの境界と一致しますが、対話操作を容易にするために、それより広い範囲に設定されている場合もあります。

ペンの方が指よりも精確にポイントできるため、ユーザーインターフェイスでタッチが正常に機能している場合は、ペンも正常に機能します。このため、この記事では、既にタッチ用に設計されているプログラムに、ペンのサポートを追加することを中心に説明します。

注: [マウス](#)、[タッチ](#)、[アクセシビリティ](#)に関するガイドラインは、それぞれ別の項目として記載しています。

## デザイン コンセプト

入力にペンを使用することには次の特徴があります。

- 自然で直感的です。ペンでポイントやタップを行う方法はだれでも理解できます。オブジェクトの対話操作は、現実の世界でユーザーがオブジェクトを操作する方法と同じになるように設計されています。
- 表現力が豊かです。マウスを使用するよりも、ペンのストロークの方が、コントロール、書き込み、描画、スケッチ、ペイント、コメントを簡単に実行できます。
- 個人的な感じが強くなります。手書きのメモまたは署名の方がタイプ入力よりも個人的な感じが強くなるように、デジタルの手書きのメモまたは署名についても同じことがいえます。
- それほど邪魔になりません。ペンは静かに使用できるので、特に会議のような社会的な状況ではタイピングやクリックほど気が散ることがありません。
- 持ち運びできます。キーボード、マウス、タッチパッドなしでほとんどのタスクを実行できるので、ペン対応のコンピューターの方がコンパクトです。操作領域を必要としないため、柔軟性に優れ、コンピューター使用の新しい場所やシナリオに対応できます。
- 直接的で魅力的です。ペンを使用すると、画面上のオブジェクトを直接操作する感覚があります。マウスやタッチパッドの場合は、手の動きをそれぞれの画面上のポインターの動きと連係させる必要があるため、間接的に感じます。

すべての Windows プログラムには適切なペン エクスペリエンスを持たせる必要があります。ペンを使用して、プログラムの最も重要なタスクを効率よく実行できる必要があります。タイピングや精緻なピクセル操作など、一部のタスクはペンに適していませんが、最低限、実行可能にすることは必要です。

都合のよいことに、プログラムが既に適切に設計され、タッチに適している場合、適切なペン サポートを提供することは簡単です。この目的に合うように適切にデザインされているプログラムでは、次のことことが実現されています。

- 適切なマウス サポート。視覚的に表現されたわかりやすいアフォーダンスと、ポインター フィードバック用のホバー状態が対話型コントロールに備わっています。標準のマウス操作(左シングルクリック、左ダブルクリック、右クリック、ホバー)に対して、オブジェクトに標準の動作が用意されています。[ポインターの形](#)は必要に応じて変化し、直接操作の種類を示します。
- 適切なキーボード サポート。標準のショートカット キー割り当てを提供し、特にナビゲーションや編集コマンドでショートカット キーを使用して(ジェスチャーを使用することもできます)作業の効率を向上させることができます。
- タッチに適した大きなコントロール。コントロールの最小サイズは  $23 \times 23$  ピクセル ( $13 \times 13$  ダイアログ ユニット [DLU])、最もよく使用されるコントロールは  $40 \times 40$  ピクセル ( $23 \times 22$  DLU) 以上です。無応答をなくすために、ターゲットの間隔が狭くならないようにします。UI 要素の間隔は、隣のターゲットと接触しているか、5 ピクセル (3 DLU) 以上空いているかのどちらかにする必要があります。
- アクセシビリティ。Microsoft Active Accessibility (MSAA) を使用し、支援テクノロジーにプログラムによる UI アクセスを提供しています。プログラムがテーマやシステムのメトリクスの変更に適切に応答します。
- **120 dpi (ドット/インチ)** ディスプレイで正常に機能し、適切に表示。120 dpi は、ペン対応コンピューターに推奨される既定の解像度です。
- コモン コントロールの使用。ほとんどのコモン コントロールは、適切なペン エクスペリエンスをサポートするようデザインされています。プログラムに必要な場合は、ターゲット指定や対話型操作を容易にするためにデザインされたカスタム コントロールが適切に実装されています。
- 有効な値の範囲のみ受け付けるコントロールの使用。簡単にターゲット指定できるデザインにするには、テキスト ボックスなどのコントロールよりも、リストやスライダーなどの選択範囲が限定されたコントロールの方が、テキスト入力の必要性が低下するので適しています。
- 適切な既定値の設定。最も安全(データの消失やシステム アクセスが失われることを防ぐため)で、最もセキュリティの高いオプションを既定で選択している。安全性とセキュリティを判断要素として考える必要がない場合は、最もよく使用されるオプションまたは最も便利なオプションを選択し、不要な対話操作を省いています。
- テキストのオートコンプリート機能。最もよく使用されるか最近使用された入力値の一覧を提供し、テキスト入力を簡易化しています。

逆に、プログラムが適切にデザインされていない場合、ペンを使用するユーザーに対して、プログラムの欠点が特に浮き彫りになります。

## ペンの対話操作モデル

ペンの使用経験がない場合は、実際に使用して理解するのが一番です。ペン対応のコンピューターを使用し、マウスとキーボードを取りはずして、普段のタスクをペンを使用して実行します。Windows Journal のようなインク対応プログラムと、インク非対応のプログラムの両方を試します。タブレット PC を使用する場合、膝の上などに乗せたり、平らなテーブルに置いたり、起立して腕で抱えて試します。縦方向と横方向の両方で使用したり、ペンでの書き込みとポイントを行ったり、右手と左手の両方でペンを使用してみます。

ペンを使用してみると、次のようなことがわかります。

- 小さなコントロールは使いにくい。コントロールのサイズは対話操作の効率に大きく影響します。サイズが  $10 \times 10$  ピクセルのコントロールはペンでも問題なく機能しますが、コントロールのサイズが大きいほど快適に使用できます。たとえば  $15 \times 11$  ピクセルの[スピンドル コントロール](#)は、ペンで快適に使用するには小さすぎます。
- 利き手が重要になる。確認や操作の対象が手で隠れることがあります。たとえば、ユーザーが右利きの場合、コンテキスト メニューをクリック ポイントの右に表示すると使いにくくなるので、左側に表示するのが適切です。Windows® では、ユーザーがコントロール パネルの Tablet PC 設定で自分の利き手を指定できま

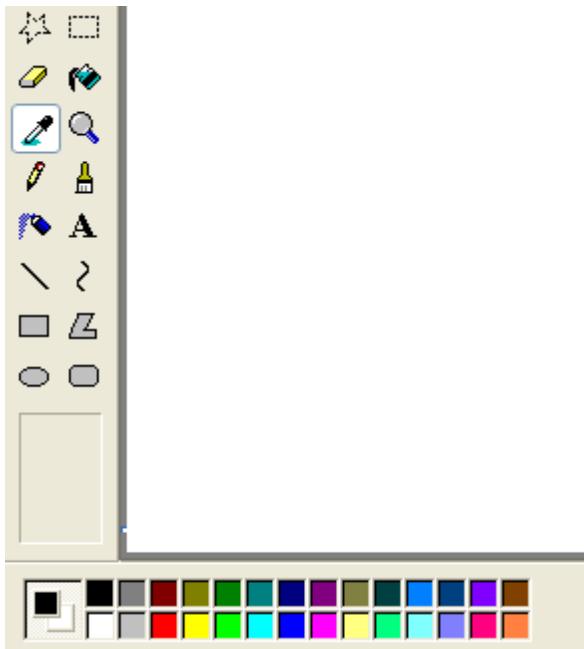
す。

- タスクを狭い範囲で実行できるようにする必要がある。画面上でポインターを 30 cm 移動する場合はマウスを 6 cm 動かすだけで済みますが、ペンを使用すると手を 30 cm 動かす必要があります。互いに離れているターゲットの間を何度も移動するのは手間がかかるため、タスクの対話操作をできる限り手に負担がかからない範囲に収めるのが適切です。コンテキストメニューは手の移動が必要ないので便利です。
- テキストの入力と選択が難しい。ペンで長文のテキストを入力するのは特に難しいため、オートコンプリート機能や許容できるテキストの既定値があるとタスクが大幅に簡単になります。テキストの選択も非常に難しいため、カーソルを正確に配置する必要がないとタスクが簡単になります。
- ディスプレイの縁端付近にある小さなターゲットをタップするのは非常に難しい。一部のディスプレイではベゼル幅が広く、一部のタッチスクリーン技術では縁端部の感度が低いため、縁端付近にあるコントロールが使いにくくなっています。たとえば、ウィンドウを最大化すると、タイトルバーにある最小化、最大化、元のサイズに戻す、閉じるの各ボタンが使いにくくなります。

#### コントロールの位置

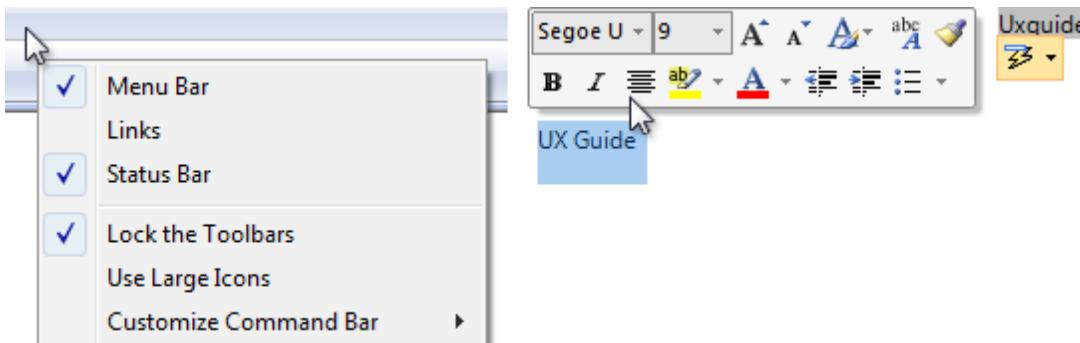
タスクの配置を狭い範囲に収めると、画面を何度も往復する手間が減ります。手の移動ができるだけ少なくするには、最も使用されそうな場所から近いところにコントロールを配置します。

#### 間違った例:



Windows XP のこの例では、カラー パレットが使用されると思われる場所から離れすぎています。

ユーザーの現在の位置からターゲットまでが最も近くなるようにすると、簡単に使用できるようになります。以下に示すように、コンテキストメニューでは Microsoft Office のミニツールバーやスマート タグと同じく、[フィットの法則](#)が最大限活用されています。



現在のポインターの位置は常に、最も簡単に使用できる場所にあります。

ディスプレイ縁端付近のターゲットが小さいとポイントが難しいため、小さなコントロールをウィンドウ縁端付近に配置しないようにします。ウィンドウが最大化されているときでも簡単にコントロールをポイントできるようにするには、コントロールのサイズを 23 × 23 ピクセル (13 × 13 DLU) 以上にする

か、ウィンドウ縁端部から離れたところにコントロールを配置します。

## ペンの対話操作

### システム ジェスチャー

システム ジェスチャーは Windows によって定義され処理されます。このため、すべての Windows プログラムでシステム ジェスチャーを使用できます。これらのジェスチャーには、マウス、キーボード、およびアプリケーションに対応するコマンド メッセージがあります。

システム ジェスチャー	生成される対応メッセージ
ホバー (サポートされている場合)	マウスでのホバー
タップ (軽くたたく)	マウスでの左クリック
ダブルタップ ("軽くたたく" を 2 回)	マウスでの左ダブルクリック
プレス アンド ホールド (押して一時停止し、上げる)	マウスでの右クリック
ドラッグ (押して移動し、上げる)	マウスでの左ドラッグ
プレス アンド ホールド アンド ドラッグ (押して一時停止したのち移動し、上げる)	マウスでの右ドラッグ
選択 (押して選択可能オブジェクト上を移動したのち、上げる)	マウスでの選択

開発者向け情報: 詳細については、[SystemGesture の列挙に関するページ](#)を参照してください。

## フリック

フリックとは簡単なジェスチャーであり、キーボードショートカットとほぼ同等です。ナビゲーションフリックには、上へドラッグ、下へドラッグ、戻る、進むがあり、編集フリックには、コピー、貼り付け、元に戻す、削除があります。フリックを使用するには、プログラムで、関連するキー入力コマンドに応答するようにするだけです。



これは、Windows 7 の 8 個のフリック ジェスチャーとその既定の割り当てです。ナビゲーションフリックは、スクロール (オブジェクトがジェスチャーとは反対の方向に移動) ではなく、パン (オブジェクトがジェスチャーに一致する方向に移動) に対応するように変更されています。



これは、Windows Vista の 8 個のフリック ジェスチャーとその既定の割り当てです。

ナビゲーションフリックでは、学習や記憶が容易になるように自然な割り当てが行われています。編集フリックは高い精度が必要な斜め方向への移動で、割り当てはあまり自然ではないため ([ごみ箱] の

方向にフリックすると削除され、[戻る]矢印の方向にフリックすると元に戻す処理が行われます)、既定では有効にされていません。すべてのフリック操作は、コントロールパネル項目のペンと入力デバイスを使用してカスタマイズすることができます。

フリック	生成される対応メッセージ
左へフリック	[進む] コマンド (Windows Vista では [戻る] コマンド)
右へフリック	[戻る] コマンド (Windows Vista では [進む] コマンド)
上へフリック	キーボードでの下へスクロール
下へフリック	キーボードでの上へスクロール
左上へフリック	キーボードでの削除
左下へフリック	キーボードでの元に戻す
右上へフリック	キーボードでのコピー
右下へフリック	キーボードでの貼り付け

## アプリケーション ジェスチャー

アプリケーションでは、他のジェスチャーも定義し処理することができます。Microsoft Gesture Recognizer では、[40 ジェスチャー](#)以上認識することができます。アプリケーション ジェスチャーを使用するには、プログラムで認識するジェスチャーを定義し、発生したイベントを処理する必要があります。

## 応答性と一貫性

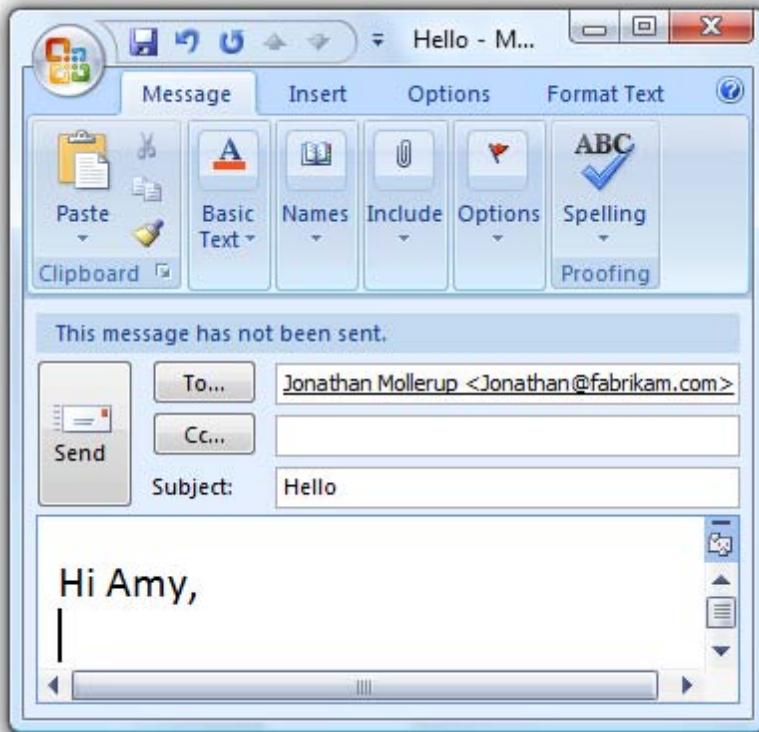
応答性は、直接的で魅力的なペン エクスペリエンスを作り出すのに不可欠です。直接的な感じを与えるには、ジェスチャーが即座に効果を現し、ジェスチャーの間、ペン部分でオブジェクトの接点が滑らかに保たれることが必要です。応答が遅れる、安定しない、接点が失われる、正確に応答しないなどが起こると、直接操作の感覚も魅力もなくなります。

一貫性は、自然で直感的なペン エクスペリエンスを作り出すのに不可欠です。ユーザーが標準のジェスチャーを習得した場合、そのジェスチャーがすべてのアプリケーション プログラムで同じ効果を生むと考えます。混乱や不満をなくすため、標準のジェスチャーには非標準の意味を割り当てないでください。プログラムに特有の対話操作にはカスタム ジェスチャーを使用します。

## インクとテキストの編集

インクとテキストの編集は、ペンを使用する際の最も難しい対話操作の 1 つです。選択範囲が限定されたコントロール、適切な既定値、オートコンプリート機能を使用すると、テキスト入力の必要性がなくなるか減少します。ただし、プログラムにテキストまたはインクの編集が含まれる場合、ペン使用時に既定で自動的に入力 UI を最大 150% 拡大すると、ユーザーの生産性を上げることができます。

たとえば電子メール プログラムでは標準サイズで UI を表示しますが、メッセージ作成時に入力 UI を 150% に拡大することができます。



この例では、入力 UI が 150% に拡大されています。

#### 4 つの重要な点

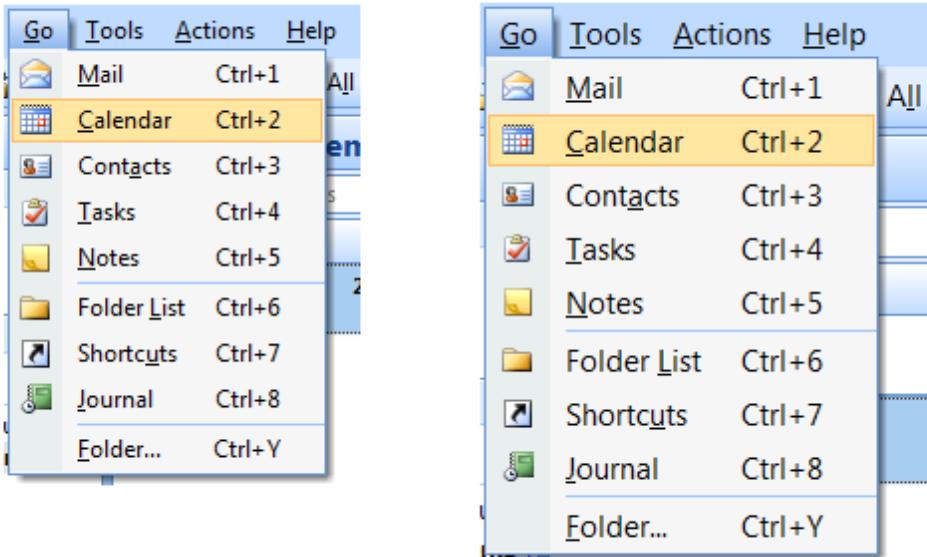
1. Windows プログラムに適切なペン エクスペリエンスが備わっているようにします。ユーザーがペンを使用し、プログラムの最も重要なタスクを効率よく実行できることが必要です(最低でも、そうしたタスクで多量のタイピングや精緻なピクセル操作を必要としないようにします)。
2. 最も適切なシナリオで、インクを使用した直接的な書き込み、描画、コメントのサポートを加えることを検討します。
3. 直接的で魅力的なエクスペリエンスを作り出すには、ジェスチャーが即座に反映され、ジェスチャーの間、ユーザーのペン部分で接点が滑らかに保たれ、ジェスチャーの効果が直接ユーザー動作に割り当てられるようにします。
4. 自然で直感的なエクスペリエンスを作り出すには、適切な標準ジェスチャーをサポートし、それぞれの標準の意味を割り当てます。プログラムに特有の対話操作にはカスタム ジェスチャーを使用します。

## ガイドライン

### コントロールの使い方

- コモン コントロールを優先して使用します。ほとんどのコモン コントロールは、適切なペン エクスペリエンスをサポートするようデザインされています。
- できるだけ制約付きコントロールを使用します。テキスト入力の負担を減らすために、テキスト ボックスのような制約のないコントロールではなく、リストやスライダーといった制約付きコントロールをできる限り使用します。
- 適切な既定値を設定します。最も安全(データの消失やシステム アクセスが失われることを防ぐため)で、最もセキュリティの高いオプションを既定で選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用されるオプションまたは最も便利なオプションを選択し、不要な対話操作を省きます。
- テキストのオートコンプリート機能を実装します。最もよく使用されるか最近使用された入力値の一覧を提供し、テキスト入力を簡易化します。
- 複数の選択項目を使用する重要なタスクについては、[標準的な複数選択リスト](#)が通常使用される場合は代わりに[チェック ボックスリスト](#)を使用するオプションを提供します。
- システムのメトリクスを尊重します。すべてのサイズでシステムのメトリクスを使用し、ハードウェア組み込みのサイズは使用しません。必要な場合は、ユーザーがシステムのメトリクスまたは dpi を適宜変更することができます。ただし、通常はユーザー側でシステム設定を調整して UI を使用可能にすることは望ましく

ないので、これは最終的な手段にします。



この例ではメニューの高さに関して、システムのメトリクスが変更されています。

#### コントロールのサイズ、レイアウト、間隔

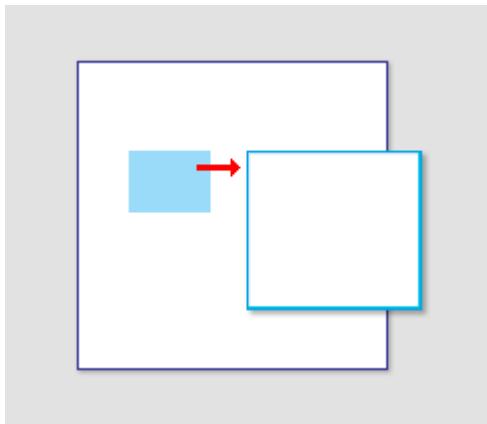
- コモン コントロールについては、[コントロールに推奨されるサイズ](#)を使用します。これらにはスピン コントロール (ペンで使用できませんが、代替手段があります) を除き、適切なペン エクスペリエンスが得られる大きさがあります。
- 最も使用されると思われる場所に近いところにコントロールを配置するレイアウトを選択します。できる限り狭い範囲にタスクの対話操作を収めます。コモンタスクとドラッグでは特に、手の移動距離が長くならないようにします。
- [推奨される間隔](#)を使用します。推奨される間隔はペンに適しています。
- 対話型コントロールの間隔は、隣のコントロールと接触しているか、5 ピクセル (3 DLU) 以上空いているかのどちらかにする必要があります。そうすることで、ユーザーが目的のターゲットの外側をタップしても混乱しません。
- コマンドリンク、チェック ボックス、ラジオ ボタンなどのコントロールのグループ内およびグループ間では、推奨される上下の間隔よりも広く間隔を空けます。そうすることで、差異化しやすくなります。

#### 対話操作

- 手書きを受け入れるように設計されているプログラムの場合、既定のインク機能を有効にします。既定のインク機能を使用すると、タップやコマンド指定などの特別な操作の必要がなく、書き始めることによってインク入力が可能になり、最も自然なペン エクスペリエンスが実現されます。手書きを受け入れるように設計されていないプログラムの場合は、テキスト ボックス内でペンを入力したときに選択部分として処理します。
- プログラムにテキスト編集を必要とするタスクがある場合は、ユーザーがコンテンツ UI を拡大できるようにします。ペン使用時は自動的に 150% に拡大されるようにします。
- ジェスチャーは記憶されるため、プログラム全体で一貫して意味を割り当てます。確定した意味を持つジェスチャーに別の意味を割り当てないでください。適切なプログラム特有のジェスチャーを使用します。

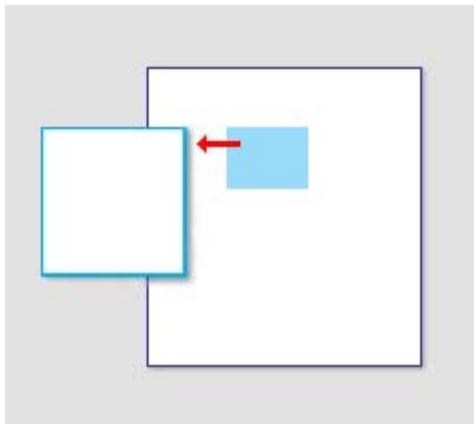
#### 利き手

- コンテキスト ウィンドウの場合は、必ず起動元オブジェクトの近くに表示します。表示するウィンドウで起動元のオブジェクトが隠れることがないように、適切な位置に配置します。
  - マウスを使用して表示する場合、可能であれば、コンテキスト ウィンドウを下方向および右方向にオフセットして表示します。



コンテキスト ウィンドウは、起動元オブジェクトの近くに表示します。

- ペンを使用して表示する場合、可能であれば、ユーザーの手で隠れることができないようにコンテキスト ウィンドウを配置します。右利きのユーザーであれば左側に、左利きのユーザーであれば右側に表示します。



ペンを使用する場合は、コンテキスト ウィンドウがユーザーの手で隠れることができないように配置します。

- 開発者向け情報: マウス イベントとペンイベントは、[GetMessageExtraInfo](#) API を使用して判別できます。また、ユーザーの利き手は、[SystemParametersInfo](#) API と SPI\_GETMENUDROPALIGNMENT を使用して判断できます。

## 寛容性

- [元に戻す] コマンドを用意します。すべてのコマンドに元に戻す機能を用意するのが理想的ですが、プログラムにはコマンドの効果を元に戻すことができないものが含まれていることがあります。
- ホバーの適切なフィードバックを提供します。ペンをクリック可能なターゲットに合わせたときに、わかるように表示します。このようなフィードバックは意図しない操作を防ぐ有用な方法の 1 つです。
- 実用的な場合は必ず、適切なペンダウンフィードバックを提供しますが、移動またはペンアップが発生するまでは操作が反映されないようにします。こうすると、ユーザーは事前に間違いを修正できます。
- 実用的な場合は必ず、ユーザーが簡単に間違いを修正できるようにします。ペンアップ時に操作が反映される場合は、ペンダウンの間にスライドによって間違いを修正できるようにします。

## ドキュメント

ペン入力に言及するときは、以下のことに留意します。

- ペン型のスタイラス入力デバイスを "ペン" と呼びます。初回言及時は、"タブレット ペン" を使用します。
- ペンの側面にあるボタンは "ペン ボタン" と表現し、"バレル ボタン" は使用しません。
- キーボード、マウス、トラックボール、ペン、指を総称して "入力デバイス" と呼びます。
- ペンの使用に特定した手順を記す場合は、"クリック" ではなく "タップ" (および "ダブルタップ") を使用します。タップとは、画面を押し、ホールド タイム前に持ち上げることを意味します。マウス クリックの生成に使用することが許可されることもあります、禁止されることもあります。ペンを使用しない対話操作の場合

は、引き続き "クリック" を使用します。

## アクセシビリティ

デザインコンセプト

ガイドライン

全般

特定の障碍への対応

アクセスキー

メニュー アクセスキー

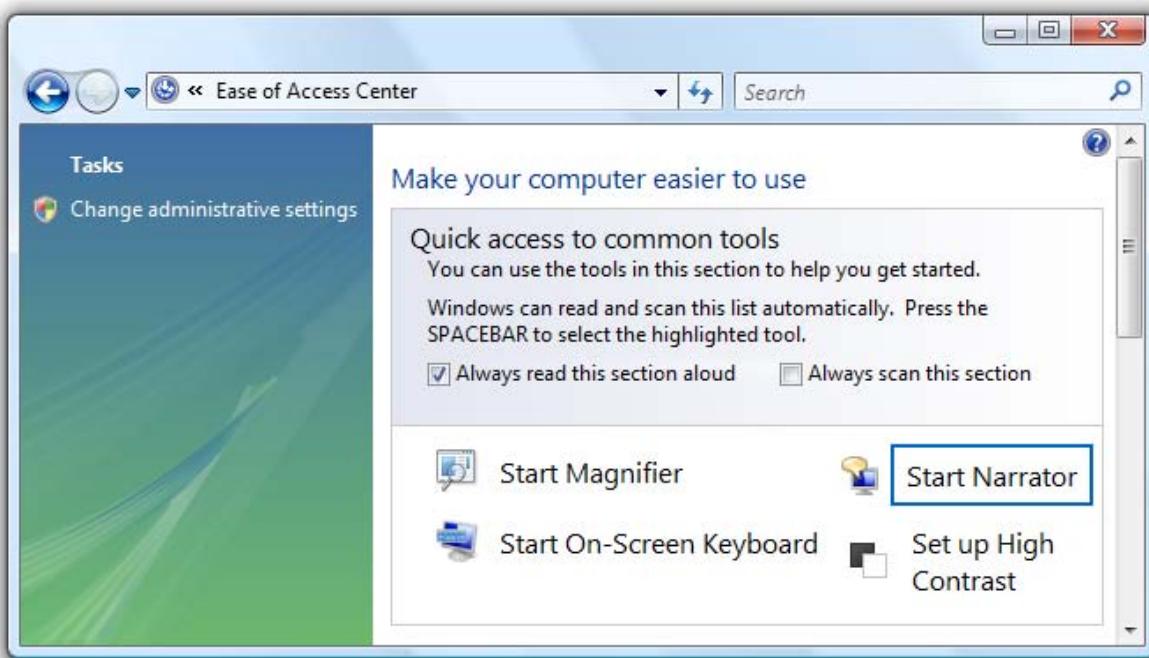
ダイアログボックス アクセスキー

テキスト

ドキュメント

"アクセシビリティ" 対応のソフトウェアをデザインするということは、障害のある方を含め幅広いユーザーがプログラムや機能を簡単に利用できるようにすることを意味します。

アクセシビリティ機能を利用しているユーザーの数は想像以上に多く、米国では、コンピューターユーザーの半数以上がアクセシビリティに関する問題を経験しており、アクセシビリティテクノロジーの利用により恩恵を受ける可能性が高いことが調査の結果わかりました。さらに、アクセシビリティの特徴である柔軟性と包括性を備えたソフトウェアデザインのアプローチにより、ユーザビリティや顧客満足度が全体的に向上することが少なくありません。



コントロールパネルの [コンピュータの簡単操作センター] では、ユーザーが必要なアクセシビリティ機能を 1 か所で選択してカスタマイズできます。

注: キーボード、マウス、色、サウンドに関するガイドラインは、それぞれ別の項目として記載しています。

## デザイン コンセプト

ユーザーがコンピューター ハードウェアとソフトウェアを操作するとき、さまざまな物理的、知覚的、認知的要素が関係してきます。プログラムの機能のアクセシビリティを向上させる方法について検討する前に、存在する障害の種類と、ユーザーがコンピューターを操作する際に使用する支援テクノロジーについて知ることは有益です。

### 障碍の種類

次の表に、共通するユーザーの障害の説明、およびコンピューターのアクセシビリティを高めるために使用される最も重要な解決方法を示します。

障碍	説明	解決方法
視覚障碍	軽度 (ユーザーの 17%) から重度 (ユーザーの 9%) の視覚障碍があるユーザーを対象としています。	カスタマイズ可能な拡大鏡、色、コントラストの設定、ブライユ点字ユーティリティ、スクリーンリーダー。
聴覚障碍	軽度 (ユーザーの 18%) から重度 (ユーザーの 2%) の聴覚障碍があるユーザーを対象としています。	情報の冗長性。サウンドは、テキストや視覚的な伝達手段の補足としてのみ

		使用されます。
四肢障碍	軽度(ユーザーの19%)から重度(ユーザーの5%)の四肢障碍があるユーザーを対象としています。四肢障碍は、キーボードやマウスを使用するときに、ある特定の動きに支障をきたすことが少なくありません。	入力手段の冗長性。マウスやキーボードに相当するものを使用してアクセスするプログラム機能。
知的障碍	記憶障碍および知覚障碍を含みます。ユーザーの16%にこの障碍が見られます。	高度なカスタマイズが可能なユーザーインターフェイス(UI)。簡単でわかりやすい段階的表示の使用。アイコンとその他の視覚的支援の使用。
発作障碍	動きや点滅に対する視覚感度を含みます。	アニメーションの使用を控えるなど、インターフェイスの刺激を和らげる控えめな表示。2~55 Hzの範囲で発生する画面のちらつきをなくします。
発語障碍または言語障碍	失読症や口頭による意思伝達障碍を含みます。	スペルチェックと文章校正ユーティリティ。音声認識および読み上げテクノロジー。

上記の障碍のあるユーザーの支援に関するその他のガイドラインについては、後の「[特定の障碍への対応](#)」を参照してください。

#### 支援技術とアクセシビリティ機能の種類

##### スクリーンリーダー

スクリーンリーダーを使用すると、視覚情報が音声に変換され、視覚障害のあるユーザーは UI を操作できます。UI テキスト、コントロール、メニュー、ツールバー、グラフィックなどの画面要素はスクリーンリーダーのコンピューター音声によって読み上げられます。スクリーンリーダー支援技術に最適化されたプログラムを作成するには、スクリーンリーダーで各 UI 要素を識別する方法について計画する必要があります。

ユーザーが操作できる各 UI 要素は、キーボードからアクセスできるだけでなく、アクセシビリティアプリケーションプログラミングインターフェイス(API)を使用して公開されている必要があります。新しいアクセシビリティフレームワークである Microsoft® UI オートメーションの使用をお勧めします。UI オートメーションは Windows Presentation Foundation (WPF) をサポートし、Microsoft Windows® のすべてのバージョンに対応しています。UI オートメーションを使用すると、プログラムでデスクトップ上のほとんどの要素にアクセスでき、スクリーンリーダーなどの支援技術製品により、UI に関する情報をユーザーに提供したり、標準の入力方式以外の方法(たとえば、マウスやキーボード操作の代わりに、またはこれらと併用して音声入力を使用する)で UI を操作できます。詳細については、「[UI オートメーションの概要](#)」を参照してください。

スクリーンリーダーは重要な支援技術ですが、支援技術はこれ以外にもあります。利用可能な技術の種類については、[支援技術製品の種類に関するページ](#)を参照してください。

##### 音声認識

音声認識は Windows® のアクセシビリティ機能で、マウスやキーボードを動かす操作の必要性を少なくて、ユーザーが音声でコンピューターを操作できるようにします。ユーザーは、声を出してドキュメントや電子メールを入力したり、音声コマンドを使用したプログラムの起動や切り替え、オペレーティングシステムの制御や Web フォームの入力も実行できます。

##### 拡大鏡

拡大鏡は画面上の項目を元の 2~16 倍に拡大して、視力が低下したユーザーを補助します。ユーザーはこの機能を設定して、拡大する対象をマウス(マウスポインターが指している場所を拡大表示)、キーボード(Tab キーで移動するときにポインターが移動する領域を表示)、またはテキスト編集(入力内容を表示)に切り替えることができます。

##### 視覚効果の設定および配色

画面上の項目を拡大することに加え、視覚障害のあるユーザーは、[ハイコントラストモード](#)などのシステム設定や、背景と前景の配色をカスタマイズできる機能によりメリットを得ることができます。

## ナレーター

ナレーターは Windows のスクリーン リーダーの簡易版で、ユーザーはエラー メッセージなどの予期しないイベントを含め、画面上のテキストや UI 要素の読み上げを聞くことができます。ユーザーはアクティブ ウィンドウを表示した状態で、ナレーター メニューを聞くことができます。



ユーザーは Microsoft ナレーターが使用される範囲をカスタマイズできます。

## スクリーン キーボード

物理的なキーボードの使用が困難で、スイッチなどの代替の入力デバイスを使用する必要があるユーザーには、スクリーン キーボードが必要です。ユーザーはマウスまたは別のポインティング デバイスを使用して、少数のキー グループ、または単一のキー(スクリーン キーボードのセットアップ方法に応じて変わります)を選択できます。

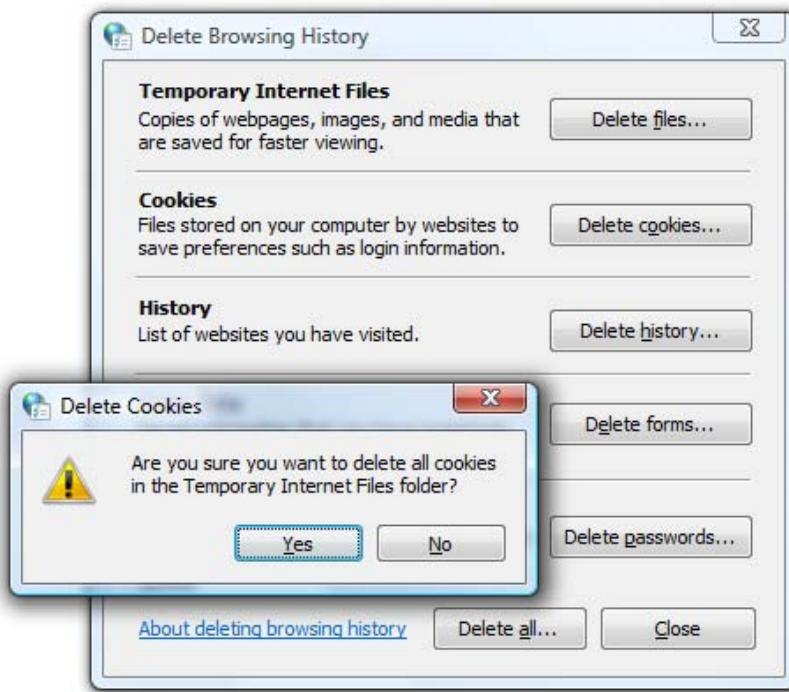
## マウス キー機能

マウス キー機能を有効にすると、キーボードの方が使いやすいユーザーはテンキー上の方向キーを使用してマウス ポインターを動かすことができます。

アクセシビリティ機能の一覧については、[Windows Vista のアクセシビリティに関するページ](#)を参照してください。

## キーボードベースのナビゲーション

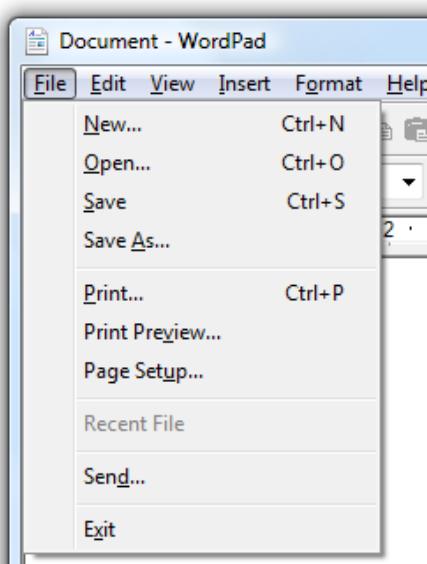
Tab キー、方向キー、Space キー、Enter キーはキーボードベースのナビゲーションにとって重要です。Tab キーを押すと、[入力フォーカス](#)が異なるコントロール グループ間で順番に移動し、方向キーを押すと、コントロール内またはグループ内のコントロール間で移動します。Space キーを押すと、入力フォーカスがあるコントロールをクリックするのと同じ動作になり、Enter キーを押すと入力フォーカスに関係なく、既定のコマンド ボタンまたはコマンド リンクをクリックするのと同じ動作になります。



この例では、ユーザーは入力フォーカスが目的のオプションに移動するまで *Tab* キーを押し、*Space* キーを押してオプションを選択します。

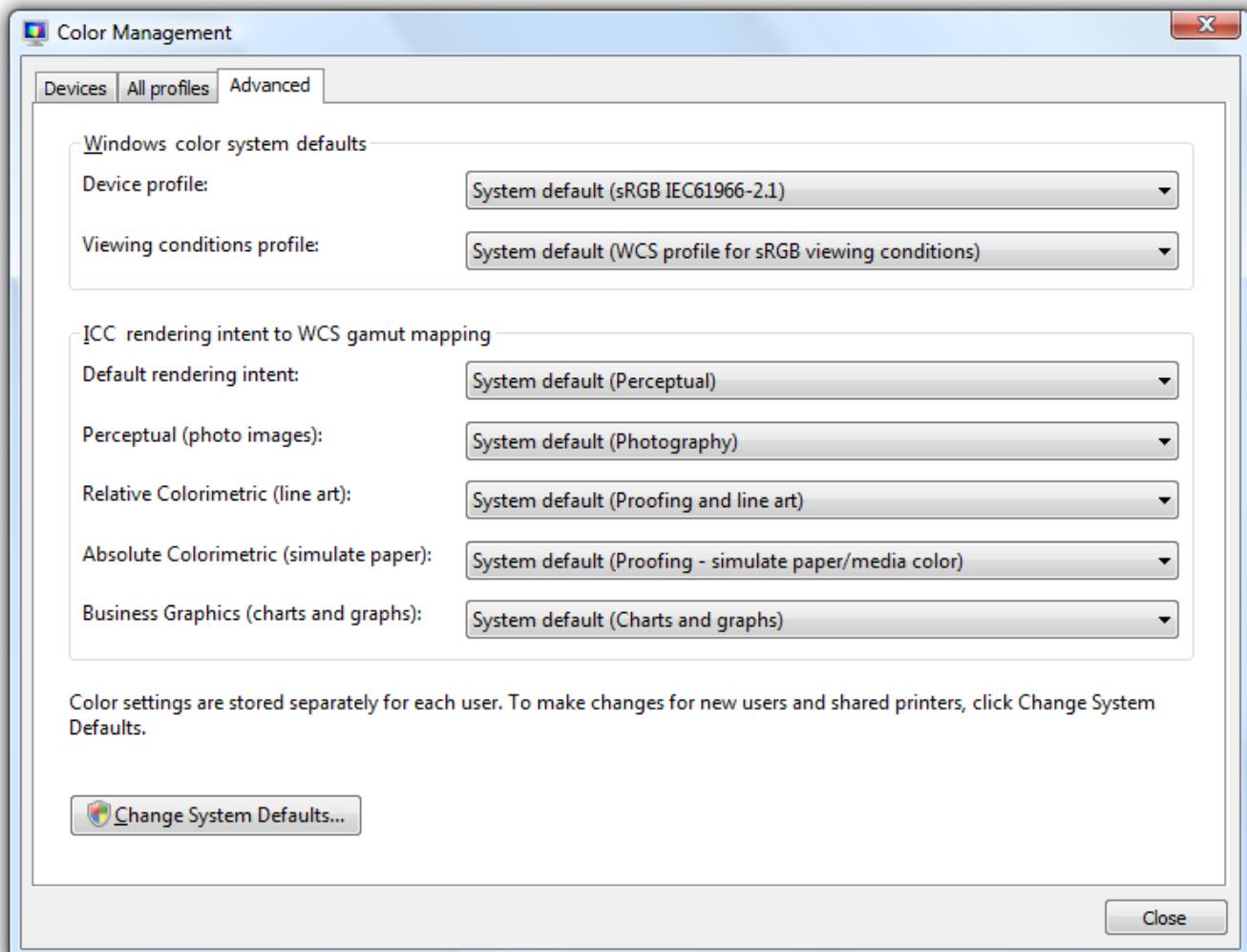
#### アクセスキー

アクセスキーを使用すると、最初にコントロールに移動しなくとも、直接オプションを選択してコマンドを開始できます。アクセスキーは、各コントロールのラベルのいずれかの文字に付いた下線で示されます。Alt キーを押しながら下線で示された文字を押すことによって、オプションやコマンドがアクティブ化されます。アクセスキーでは、大文字と小文字は区別されません。



この例では、*Alt* キーを押しながら *O* キーを押すと、[開く] コマンドがアクティブ化されます。

通常、コントロールに対して論理アクセスキーを選択することは難しくはありませんが、ウィンドウ上のコントロールの数が多いほど、アクセスキーの選択肢が少なくなる可能性が高くなります。この場合、個々のコントロールではなく、コントロールグループごとにアクセスキーを割り当てます。



この例では、アクセスキーは個々のコントロールではなく、コントロールグループごとに割り当てられています。

アクセスキーはショートカットキーと混同されることがよくありますが、ショートカットキーの割り当てはアクセスキーと異なり、別の目的があります。たとえば、ショートカットキーは *Ctrl* キーとファンクションキーの組み合わせを使用しますが、その主な目的はアクセシビリティではなく、詳しい知識のあるユーザー向けのショートカットとして使用することです。

詳細については、「[キーボード](#)」を参照してください。

### アクセシビリティに対応したデザイン: 3 つの基本的な方法

アクセシビリティに対応したプログラムは、アクセシビリティとユーザビリティの目的が重なるため、何らかの形ですべてのユーザーに役立ちます。たとえば、詳しい知識のあるユーザーの効率を可能な限り向上させるためにデザインされた機能によって、四肢障碍のためにキーボードの方が使いやすいユーザーもメリットが得られます。

3 つの基本的な方法とは、UIにおいてある程度の柔軟性を持たせること、デザインを判断するうえでユーザーのニーズと嗜好を尊重して大きな影響力を持たせること、UIへのプログラムによるアクセスを提供することであり、アクセシビリティに対応したデザインに役立ちます。

#### 柔軟性のある UI の提供

アクセシビリティに対応したデザインとは、少なくとも部分的にユーザーに選択肢を与えることです。ストレスを引き起こす、目まいのするような選択肢を並べるのではなく、ユーザーのニーズを的確に捉えた選択肢の数に限定します。マウスを使った操作が苦手な場合は、キーボードだけを使用して同じ操作を実行できます。物理的なキーボードが苦手な場合は、画面上で使用できる仮想的なキーボードがあります。

たとえば、以下の方法で柔軟性を提供します。

- たとえば、グラフィックの代わりに代替テキスト、音声の代わりにキャプションを使用するなど、テキスト以外の要素に相当する要素をユーザーが選択できるようにします。



Click to sign in to your Passport account.

グラフィックの表示を選択しないユーザーには、代わりに代替テキストを表示し、コントロールの内容と操作方法について説明する必要があります。

- たとえば、アイコンを形で区別できるようにしたり、サウンドを使用して、色以外の別の方法を提供します。



この例では、標準のアイコンをデザインによって簡単に見分けることができます。

- すべての対話型コントロールにタブストップを設定するなど、キーボードでのアクセスを提供し、マウスまたはキーボードのどちらを使用してもプログラムで同じ操作ができるようになります。
- プログラムでユーザーが適切な色のコントラストのオプションを選択できるようにします。Windows では、ハイコントラストのオプションを提供していますが、これは重度の視覚障害に対応した解決方法としてデザインされたものです。他のコントラストのオプションは、視力の低下や色盲など軽度の視覚障害のあるユーザーを対象としています。プログラムの色の使用方法が、アクセシビリティを妨げず、一次的伝達手段になっていないことを確認するには、[富士通 ColorDoctor](#) や [Vischeck](#)などのユーティリティを使用することをお勧めします。
- たとえば、フォントサイズの選択にスライダー コントロールやドロップダウン ボックスを使用して、プログラムの UI のテキストのサイズを調整する手段をユーザーに持たせるようにします。可能であれば、高 dpi モードをサポートします。
- マルチモーダルなプログラムにします。つまり、プログラムの通常モードが一部のユーザーのアクセシビリティを妨げている場合は、別の手段でこの問題を回避できるようにします。たとえば、アニメーションが表示されている場合、ユーザーの選択により、アニメーション以外の提示モードを少なくとも 1 つ用意して、情報を表示できるようにする必要があります。

マルチモーダルインターフェイスと柔軟性のある操作性により、情報に冗長性のあるアーキテクチャをユーザーに提供します。冗長性は否定的な意味合いを持つことがあります。ユーザーインターフェイスのテキストを例に挙げると、読むことに関するエクスペリエンスを合理化するために冗長性を排除することを推奨しています。これに対して、アクセシビリティのコンテキストにおいて、冗長性は肯定的で、フェールセーフなしくみやエクスペリエンスといった意味合いを持ちます。

#### ユーザーへの配慮

配慮は、一般的に、アクセシビリティに対応したプログラムをデザインするうえで重要な指針となる原則です。たとえそれが知的訓練だとしても、障害のあるユーザーとして初めてプログラムを使用することがどのようなものかを想像してみます。時間をかけて UI 画面をハイコントラストおよびさまざまな

解像度でテストし、エクスペリエンスが視覚障碍のあるユーザーに対して適切なものであることを確認します。コントロールパネルの項目の[コンピュータの簡単操作センター]にある[ショートカットキーとアクセスキーに下線を表示します]チェックボックスを選択し、アクセスキーが常に表示されるようにして、キーボードのアクセシビリティをテストします。最初に、他人への共感力に優れた開発者や設計者を招いて、さらに厳しいテストを実施することもできます。

以下の方法で、配慮を示します。

- 特定のプログラムに適用される固定した設定ではなく、システム全体に適用される設定(システムカラーなど)を使用します。ユーザーがプログラムを操作するために指定して選択したパラメーターだけでなく、オペレーティングシステムに組み込まれ、使用するプログラムに関係なくユーザーが必要とするアクセシビリティ機能についても配慮します。詳細については、[MSDN アクセシビリティ デベロッパー センター](#)を参照してください。
- カスタムコントロールよりも、Windows アクセシビリティ API が既に実装されているコモンコントロールを優先して使用します。
- すべてのキーボードショートカットなど、アクセシビリティのオプションと機能をすべて説明します。障害のあるユーザーはアクセシビリティ機能を見つけることに対して意欲的で、総合的な情報はヘルプに集約されていると想定していることが多くあります。
- アクセシビリティに対応した形式で、アクセシビリティに対応したドキュメントを作成します。つまり、ドキュメント自体もプライマリ UI と同じアクセシビリティの規則に従う必要があります。これには、フォントサイズを拡大できる機能、グラフィックの代替テキストの使用、冗長な情報のアーキテクチャ(たとえば、色分けはテキストの補足としてのみ使用するなど)が含まれます。

ソフトウェア製品において、ユーザーに対する配慮は、ユーザビリティと市場調査、効果的なサポートサービスとドキュメント、そしてもちろんデザイン上の判断に現れます。たとえば、詳しい知識のあるユーザー向けのデザインという観点から、最新機能を装備するのは自分の目的のためか、それとも詳しい知識のあるユーザーがその機能を求めていることを知っているからかを考え直してみます。後者の場合は、配慮という価値観に基づいて十分な情報を得たうえで、デザイン上の意思決定が行われています。

#### プログラムによるアクセスの提供

UIへのプログラムによるアクセスを提供することは、スクリーンリーダー、代替の入力デバイス、音声認識プログラムなどの支援技術において、ユーザーのために画面を正しく解釈するうえで不可欠です。プログラムの各 UI 画面の"マップ"を作成して、支援技術のユーザーが利用できるようにします。

これを以下の方法で適切に行います。

- Active Accessibility COM インターフェイス、IAccessible などを使用して、プログラムによるすべての UI 要素およびテキストへのアクセスを有効にします。
- IAccessible Name プロパティなどを使用して、UI オブジェクト、フレーム、ページに名前(またはタイトル)と説明を配置します。
- フォーカスの移動を含むすべての UI 操作のフォーカスイベントなど、すべての UI 操作によってプログラムによるイベントがトリガーされるようにします。

#### 4つの重要な点

- すべてのユーザーがプログラムの潜在的な能力を最大限に活用できるようにします。
- 独創的な問題解決の機会や、全体的なユーザー満足度向上の手段を得る機会として、アクセシビリティを捉えます。
- システム設定に配慮します。
- できる限りコモンコントロールを使用します。

## ガイドライン

### 全般

- アクセシビリティ機能として識別されるオペレーティングシステムや他の製品のアクティブ化された機能を中断したり、無効にしないでください。アクティブ化された機能は、対象となるオペレーティングシステムまたは製品のドキュメントを参照して見つけることができます。
- プログラムのウィンドウを画面の最前面に表示してユーザーに操作を強制しないようにします。ユーザーがタスクを実行するために、継続的な機能またはウィンドウの表示が必要な場合は、そのウィンドウは常に表示されている状態にする必要があります。ユーザーがそのウィンドウを表示するように選択している場合は、他のウィンドウに対する相対的位置に関係なく、そのウィンドウを最前面に表示する必要があります。たとえば、ユーザーが移動可能なスクリーンキーボードを他のすべてのウィンドウの最前面に常に表示されるように配置している場合は、プログラムを [ゾオーダー](#) の最上位に固定して配置し、スクリーンキーボードを隠さないようにします。
- できる限り、システムカラー、システムフォント、コモンコントロールを使用します。こうすることで、アクセシビリティに関して発生する問題の件数を大幅に減らすことができます。

## 特定の障碍への対応

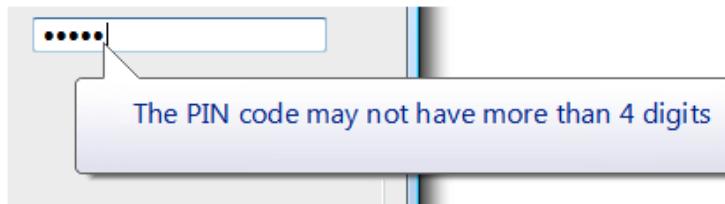
### 視覚障碍

- 意味を伝えるときに、色だけに頼らないようにします。色は、テキスト、デザイン、場所、またはサウンドの意味を補強する手段としてのみ使用します。



この例では、一次的伝達手段となるのは簡潔なツールヒント テキストです。色の使用は意味を伝えるうえで役立ちますが、二次的手段として使用します。

- グラフィックを説明する代替テキストの情報ヒントを使用します。
- グラフィック内にテキストは使用しません。視覚障碍のあるユーザーは、Web ブラウザーなどでグラフィックを無効にしていたり、単にグラフィックのテキストを見ないようにしている場合があります。
- ダイアログ ボックスやウィンドウに適切な名前を付けます。これは、画面を見ているのではなく、スクリーン リーダーなどを使用して聞いているユーザーに適切なコンテキスト情報を提供するためです。
- 視覚的な表示設定に配慮します。これは常に、テーマと GetSystemMetrics API からフォントの書体、サイズ、色、Windows 画面の要素サイズ、システム構成設定を取得して行います。
- パルーン テキストは常に簡潔にします。こうすることで読みやすくなり、スクリーン リーダーでの問題も最小限となります。



必要に応じて、パルーンでは本文テキストを追加して使用できますが、この例では、タイトルのテキストだけでも、簡潔でアクセシビリティに優れた方法で同じ目的を達成できる場合があることを示しています。

### 聴覚障碍

- 意味を伝えるときに、サウンドだけに頼らないようにします。サウンドは、テキスト、デザイン、場所、または色の意味を補強する手段としてのみ使用します。
- ユーザーが音声出力の音量を制御できるようにします。制御には Windows 音量ミキサーを使用します。詳細については、「[サウンド](#)」を参照してください。
- プログラムのサウンドは、500 ~ 3000 Hz の範囲で出力されるようにターゲットを設定します。または、ユーザーがその範囲内にサウンドを簡単に調節できるようにします。サウンドがこの範囲内で出力されると、聴覚障碍のある方が聞き取れる可能性が最も高くなります。
- 視覚的なコンテンツの妨げとならないようにキャプションを配置します。



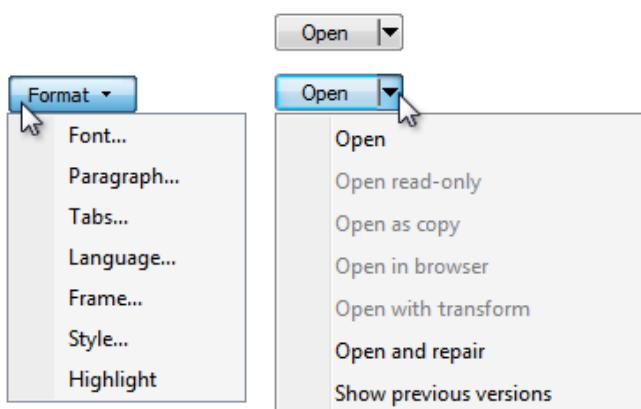
Microsoft Windows Media® Player では、キャプションは別のウィンドウに表示され、プログラムのエクスペリエンスに不可欠な部分である視覚的要素を妨げないようにしています。

#### 四肢障碍

- 絶対時間を使用する代わりに、`GetDoubleClickTime()` に対する相対的なタイムアウト値を設定します。この方法を使用すると、ユーザーの操作速度に合わせてタイムアウトを調節できます。
- すべてのメニュー項目にアクセスキーを割り当てます。これにより、キーボードで操作することを望むユーザーは、マウスを使用して操作するユーザーと同じようにプログラムを操作できます。
- 操作を実行する方法を、ダブルクリックとドラッグのみに限定しないようにします。ダブルクリックとドラッグは、一部のユーザーにとって困難を伴う操作になります。
- プログラムからメニューバーを削除しないようにします。キーボードを使用するユーザーにとって、ツールバーよりもメニューバーの方がアクセスしやすくなります。既定でメニューバーを表示する必要がない場合は、代わりにメニューバーを非表示にします。
- [ヘルプ] ボタンやリンクにタブストップを配置して、キーボードからヘルプにアクセスできるようにします。
- プログラムで割り当てられているアクセスキーを認識しやすくするために、アクセスキーを常に表示するように設定できます。コントロールパネルで、[コンピュータの簡単操作センター] にアクセスし、[キーボードを使いやすくします] をクリックして、[ショートカットキーとアクセスキーに下線を表示します] チェックボックスをオンにします。

#### 知的障碍

- 段階的表示を使用して、簡単でわかりやすい UI にします。



これらの例では、コマンドボタンから利用できるオプションは既定で非表示となっており、ユーザーは段階的表示コントロールを利用して、オプションの表示を選択できます。

- アイコン、ツールバー、その他の視覚的支援を使用してテキストを読む認知的負荷を軽減します。
- 可能であれば、テキストボックスや編集可能なドロップダウンリストで、オートコンプリート機能を装備します。これにより、ユーザーはコマンド名やファイル名、限られたオプションセットの同様の選択肢をすべて入力せずに済みます。これにより、すべてのユーザーの認知的負荷を軽減し、スペリングや入力が難しい、速度が遅い、または困難なユーザーの入力量を少なくします。
- 難しい概念は、チュートリアルやアニメーションを追加してヘルプに記載します。アニメーションは、発作障害のあるユーザーにとって問題になる場合があるので、必要な場合のみ使用するようにします。

## 発作障害

- 光ったり点滅するテキストやオブジェクト、その他の 2 ~ 55 Hz の周波数範囲で光ったり点滅する要素は使用しません。
- アニメーションの使用を制限します。ユーザーによっては、特に視野周辺部の画面の動きに対してかなり敏感になります。アニメーションを使用して何かに注目させる場合、注意を引く対象がユーザーの操作を中断するに値するものである必要があります。

## 発語障害または言語障害

- 明確で簡潔な、わかりやすいテキストを構成して記述します。ユーザビリティ テストの結果、重要な情報は語句の最後に展開すると理解の助けになることがわかりました。その他のガイドラインについては、「スタイルとトーン」を参照してください。

間違った例:

3 は次の桁ですか。

[OK] をクリックして開始します。

正しい例:

次の桁は 3 ですか。

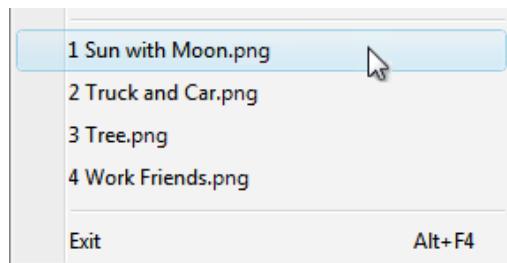
開始するには、[OK] をクリックします。

## アクセス キー

- できるだけ幅の広い文字を使用します。たとえば、w、m、大文字などを使用します。
- できるだけ特徴のある子音または母音を使用します。たとえば、"Exit"(終了)の "x" などを使用します。
- 下線が見えにくくなる文字を使用しないようにします。以下に、問題点が大きいものから順に示します。
  - 1ピクセルしか幅のない文字 (i、l など)。
  - ディセンダーのある文字 (g、j、p、q、y など)。
  - ディセンダーのある文字の隣にある文字。

## メニュー アクセス キー

- すべてのメニュー項目にアクセス キーを割り当てます。例外はありません。
- 動的なメニュー項目(最近使用されたファイルなど)については、数値的にアクセス キーを割り当てます。



この例では、Windows のペイント プログラムで、数字のアクセス キーを最近使用されたファイルに割り当てています。

- メニュー階層内で一意となるアクセス キーを割り当てます。別のメニュー階層ではアクセス キーを再び使用することができます。
- 次のようにして、アクセス キーを見つけやすくなります。
  - 最も使用頻度が高いメニュー項目については、ラベルの 1 番目または 2 番目(できれば 1 番目)の単語の先頭文字を選択します。
  - 使用頻度の低いメニュー項目については、ラベル内の特徴的な子音または母音を選択します。

## ダイアログ ボックス アクセス キー

- 可能な限り、すべての対話型コントロールまたはそのラベルに一意のアクセス キーを割り当てます。読み取り専用のテキスト ボックスは対話型コントロールである(ユーザーがスクロールし、テキストをコピーできる)ため、アクセス キーを割り当

てるメリットがあります。以下には、アクセスキーを割り当てないでください。

- [OK]、[キャンセル] および [閉じる] の各ボタン。Enter キーおよび Esc キーが、それぞれのボタンのアクセスキーに使用されています。ただし、"OK" または "キャンセル" を意味する異なるラベルのコントロールには、アクセスキーを常に割り当てます。



この例では、肯定的なコミットボタンにアクセスキーを割り当てています。

- グループラベル。通常、グループ内の各コントロールにはアクセスキーが割り当てられているため、グループラベルに割り当てる必要はありません。ただし、アクセスキーが不足している場合は、グループラベルにアクセスキーを割り当て、各コントロールに割り当てないようにします。
- 汎用的なヘルプボタン。ヘルプには F1 キーを押してアクセスします。
- リンクラベル。リンクが多すぎて一意のアクセスキーを割り当てることができなかったり、リンクの下線によってアクセスキーを示す下線がわからなくなることがあります。代わりに、Tab キーを使用してリンクにアクセスするようにします。
- タブ名。タブ間を順番に移動するには、Ctrl + Tab キーおよび Ctrl + Shift + Tab キーを使用します。
- 「...」というラベルが付けられている参照ボタン。これらにはアクセスキーを一意に割り当てることができません。
- ラベルのないコントロール。スピンコントロール、グラフィックコマンドボタン、ラベルのない段階的表示コントロールなどがあります。
- ラベルのない静的テキストまたは対話型ではないコントロールのラベル。進行状況バーなどがあります。
- 標準的なアクセスキー割り当てになるように、最初にコミットボタンにアクセスキーを割り当てます。標準的なアクセスキー割り当てが存在しない場合は、最初の文字を使用します。たとえば、[はい] コミットボタンおよび [いいえ] コミットボタンのアクセスキーは、ダイアログボックスの他のコントロールに関係なく、常に "Y" および "N" にする必要があります。
- "Don't (~しない)" が使用された否定的なコミットボタン ([キャンセル] を除く) については、"Don't" の "n" にアクセスキーを割り当てます。"Don't (~しない)" が使用されていない場合は、標準のアクセスキー割り当てを使用するか、最初の文字を割り当てます。こうすることで、すべての否定的なコミットボタンのアクセスキー割り当てに一貫性が保たれます。
- アクセスキーを簡単に見つけることができるよう、ラベルの前の方の文字 (最初の文字が理想的) をアクセスキーに割り当てます。ラベルの後半にキーワードが含まれている場合も同様です。

その他のガイドラインと例については、「[キーボード](#)」を参照してください。

## テキスト

- 外部コントロールラベルの末尾にコロンを使用します。一部の支援テクノロジーでは、コロンでコントロールラベルを識別します。
- ラベル付けする要素に対して、一貫して相対的な位置にラベルを配置します。こうすることで、支援テクノロジーによってラベルは対応するコントロールに正しく関連付けられ、画面拡大機能を使用するユーザーはラベルやコントロールを見つけるにはどこを見れば良いかが簡単にわかります。



この例では、各ドロップダウンリストのラベルの配置には一貫性があり、コロンが使用されています。

- 代替テキストは半角換算で最大 150 文字に制限します。クリック、右クリックなど、コントロールをアクティブ化する操作を説明してから、コントロールの機能について説明します。

許容される例:

ボタン。

青い丘。

より良い例:

クリックしてアカウントにサインインします。

遠くの丘の写真は、距離が遠くなるにつれて色が薄くなります。

- テキストを使用して、下線やボックス、その他のグラフィック記号を描画しないようにします。このような方法で文字を使用すると、スクリーンリーダーを使用するユーザーが混乱する可能性があります。たとえば、"X"という文字を使用して、テキストの周辺をボックスで囲むと、スクリーンリーダーソフトウェアでは、最初の1行目は"XXXXXX"と読み上げられ、続いて"x"、テキストの内容、"x"というふうに読み上げられます。

## ドキュメント

- すべてのキーボードショートカットなど、アクセシビリティのオプションと機能をすべて説明します。
- アクセシビリティに対応した形式で、アクセシブルなドキュメントを作成します。つまり、ドキュメント自体もプライマリ UI と同じアクセシビリティの規則に従う必要があります。
- "アクセスキー"と表現し、"ショートカットキー"（意味と用途が異なります）、「ニーモニック」、「アクセラレータ」とは表現しません。
- 一般的に、「障害者」ではなく、「特定の障害のあるユーザー」と表現します。ラベルではなく人を第一に考えます。

使用する表現	使用しない表現
四肢に障害がある、動作に障害がある	不具の、足の不自由な
障害のない	健常者、身体健全な、健康な
片手の (One-handed)、片手で入力するユーザー	片腕の (Single-handed)
障害のあるユーザー	身体障害者、身障者、ハンディキャップのある、障害者
知的障害、発達障害	物覚えが悪い、知恵後れ、精神的にハンディキャップがある

## ウィンドウ

ここでは、Windows® ベースのアプリケーションで各ウィンドウを使用するためのガイドラインについて説明します。

- ウィンドウの管理
- ダイアログ ボックス
- コモン ダイアログ ボックス
- ウィザード
- プロパティ ウィンドウ

# ウィンドウの管理

デザインコンセプト

ガイドライン

全般

タイトルバーとコントロール

ウィンドウ サイズ

ウィンドウの位置

ウィンドウの順序 (Zオーダー)

ウィンドウのアクティビティ

入力フォーカス

値の保持

ここでは、ウィンドウの初期表示時の既定の配置、他のウィンドウとの相対的な重なり順序 ([Zオーダー](#))、初期サイズ、表示が入力フォーカスに及ぼす影響について説明します。

以下のガイドラインで使用する用語の定義は次のとおりです。

- ・ "最上位" のウィンドウとは、オーナー ウィンドウを持たないウィンドウで、タスク バーに表示されます。たとえば、アプリケーション ウィンドウがその例です。Windows Vista® 以降では、オーナー ウィンドウを持たないダイアログ ボックスと、プロパティ シートも最上位のウィンドウと見なされます。
- ・ "所有される" ウィンドウ (子) には "オーナー" ウィンドウ (親) があり、タスク バーには表示されません。たとえば、モーダル ダイアログ ボックス、モードレス ダイアログ ボックスがその例です。
- ・ "ユーザー起動" ウィンドウは、ユーザー操作の直接の結果として表示されるウィンドウです。これ以外に、プログラムによって起動される "プログラム起動" ウィンドウと、Microsoft® Windows® によって起動される "システム起動" ウィンドウがあります。たとえば、オプション ダイアログ ボックスはユーザー起動 ウィンドウですが、会議の通知はプログラム起動 ウィンドウです。
- ・ "コンテキスト ウィンドウ" は、起動元オブジェクトとの関連性が高いユーザー起動 ウィンドウです。たとえば、コンテキスト メニュー や通知領域 アイコンをクリックして表示される ウィンドウはコンテキスト ウィンドウですが、メニュー バーの操作によって表示される ウィンドウはコンテキスト ウィンドウではありません。
- ・ "アクティブな" モニターとは、アクティブ プログラムが実行されている モニターです。
- ・ "既定の" モニターとは、スタート メニュー、タスク バー、通知領域が表示されている モニターです。

## デザイン コンセプト

ウィンドウの管理は、最も基本的なユーザー操作の 1 つです。Windows Vista より前は、ほとんどの場合、ウィンドウは所定の小さな既定 サイズで、画面の中央に配置されていました。この方法は、以前のように低解像度のモニターを 1 台 使用する場合には十分に機能しますが、最近のビデオ ハードウェアでは不十分です。

Windows は、最新のビデオ ハードウェアをサポートできるように設計されています。最新のビデオ ハードウェアは、ほとんどの場合、サポートされる最小画面解像度を大幅に上回る高解像度で実行され、モニターが複数ある場合もあります。そのために以下を実現しました。

- ・ 高度なハードウェアのメリットを十分に活用できるようにする。
- ・ ユーザーがマウスをあまり動かす必要がないようにする。
- ・ ウィンドウを予測しやすい場所に配置し、見つけやすくする。

## サポートされる最小画面解像度

Windows でサポートされる最小の有効画面解像度は、 $800 \times 600$  ピクセルです。これは、固定 サイズ ウィンドウは、最小解像度で (タスク バーの領域を確保しながら) 完全に表示される必要があり、サイズ変更可能な ウィンドウでは、最小解像度で機能する限り、 $1024 \times 768$  ピクセルの有効解像度に最適化できるという意味です。

現在、Windows PC で最も一般的な物理画面解像度は、 $1024 \times 768$  ピクセル以上ですが、 $800 \times 600$  ピクセルを対象にすることで Windows で以下が可能となります。

- ・ 小さなノートブック PC を含む、あらゆる最新のハードウェアで機能させる。

- 高dpi(ドット/インチ)設定をサポートする。
- アクセシビリティのために、大きめのフォントをサポートする。
- 世界中で使用可能なハードウェアをサポートする。

サポートする最小解像度を選択するにあたって、適切なバランスを確保する必要があります。高解像度を対象にすると、ほとんどの最新ハードウェアにとってほぼ最適といえるエクスペリエンスを実現できますが、一方、低解像度を対象にすると、設計者は利用可能な画面領域を最大限に活用することができません。

対象ユーザーが Windows の最小解像度を大幅に上回る解像度を使用していると想定できる場合は、大幅にサイズ変更可能なウィンドウを使用し、画面領域を最大限に活用してプログラムを構築することができます。

## ガイドライン

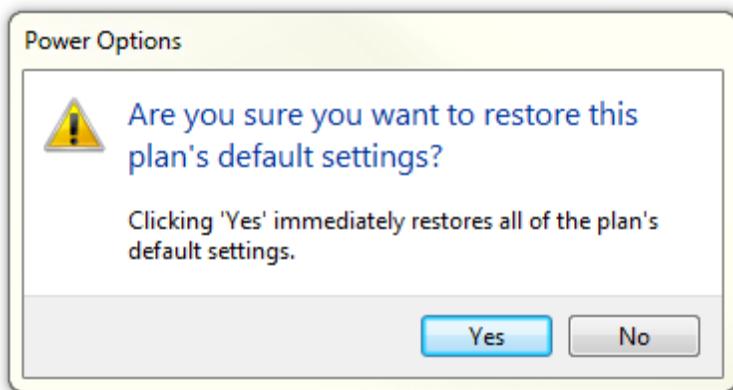
### 全般

- Windows の最小有効解像度 (800 × 600 ピクセル) をサポートします。セーフモードで動作させる必要がある重要なユーザーインターフェイス(UI)では、640 × 480 ピクセルの有効解像度をサポートします。タスクバーと共に表示するウィンドウについては、縦方向に 48 相対ピクセル 短くして、タスクバー用の領域を確保します。
- サイズ変更可能なウィンドウのレイアウトを 1024 × 768 ピクセルの有効解像度に最適化します。より低い解像度の場合にも機能するように、これらのウィンドウのサイズが自動的に変更されるようにします。
- 96 dpi (100%) の 800 × 600 ピクセル、120 dpi (125%) の 1024 × 768 ピクセル、および 144 dpi (150%) の 1200 × 900 ピクセルで、必ずウィンドウをテストします。コントロール、テキスト、ウィンドウなどが切り詰められていないか、アイコンやビットマップが引き伸ばされたりしていないかなど、レイアウト上の問題をチェックします。
- タッチ PC やモバイル PC 向けシナリオのプログラムでは、120 dpi に最適化します。現在、タッチ PC やモバイル PC では高 dpi 画面が普及しています。
- サイズ変更可能ウィンドウにサイズ変更グリフを表示する必要はありません。サイズ変更グリフは右下に表示されていましたが、不要になった理由は次のとおりです。
  - 右下だけでなく、ウィンドウのあらゆるサイズと端を変更できます。
  - グリフを表示するにはステータスバーが必要ですが、サイズ変更可能ウィンドウの多くにはステータスバーがありません。
  - ウィンドウサイズを変更可能であることを提示するうえで、サイズ変更可能ウィンドウの境界線とサイズ変更ポインターは、サイズ変更グリフより効果的です。

### タイトルバー コントロール

タイトルバー コントロールは、次のように使用します。

- [閉じる]。標準のウィンドウ枠を使用するすべてのメインウィンドウとサブウィンドウには、タイトルバー上に [閉じる] ボタンを用意する必要があります。[閉じる] をクリックすると、ウィンドウをキャンセルしたり、閉じることができます。



この例では、ダイアログ ボックスのタイトルバーに [閉じる] ボタンがありません。

- ・[最小化]。すべてのメイン ウィンドウと長時間動作するモードレスのサブ ウィンドウ(進行状況ダイアログなど)には、[最小化] ボタンを用意する必要があります。[最小化] をクリックすると、ウィンドウがタスクバー ボタンに最小化されます。このため、最小化可能なウィンドウにはタイトルバー アイコンが必要になります。
- ・[最大化]/[元に戻す]。すべてのサイズ変更可能なウィンドウには、[最大化]/[元に戻す] ボタンを用意する必要があります。[最大化] をクリックすると、ウィンドウが最大サイズ(ほとんどのウィンドウは全画面)で表示され、[元に戻す] をクリックすると、ウィンドウが以前表示されたサイズで表示されます。しかし、中には全画面表示するメリットがないウィンドウもあるため、そのようなウィンドウの場合は、使いやすい一番大きなサイズで表示されます。

## ウィンドウ サイズ

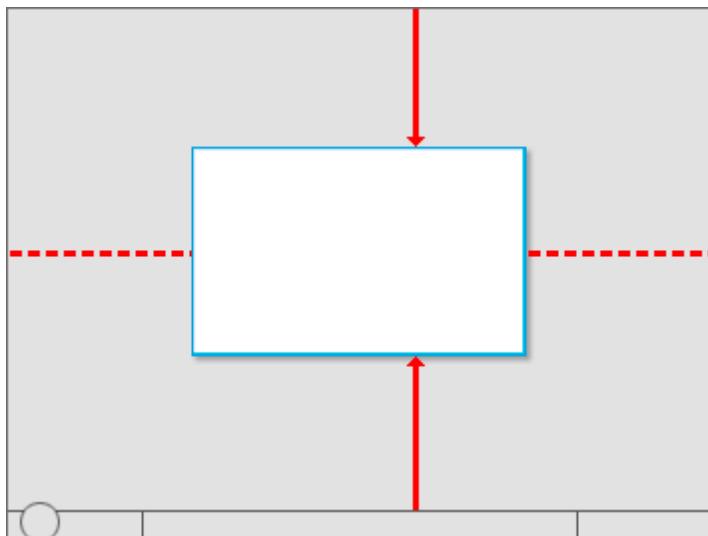
- ・コンテンツに適した既定のウィンドウ サイズを選択します。画面を効率的に使用できる場合は、ウィンドウの初期サイズを大きくします。
- ・サイズ変更可能なウィンドウを使用するとスクロールバーの使用やデータの切り詰めを避けることができる場合は、常にサイズ変更可能なウィンドウを使用します。コンテンツや一覧が動的に変化するウィンドウでは、サイズ変更ウィンドウを有効活用できます。
- ・テキスト ドキュメントでは、1行の文字数は半角換算で 65 文字以内になるようにします。これより長いとテキストが読みにくくなります(句読点や空白文字も含めてカウントします)。
- ・固定サイズのウィンドウの場合:
  - ・[作業領域](#)内に全体が表示され、収まるサイズにする必要があります。
- ・サイズ変更可能なウィンドウの場合:
  - ・高解像度に合わせて最適化されますが、表示の際には実際の画面解像度に合わせて、必要に応じてサイズが小さくなります。
  - ・ウィンドウ サイズが大きいほど、表示される情報が多くなる必要があります。少なくともウィンドウ 領域の一部または1つのコントロールに、サイズ変更可能なコンテンツを含めるようにします。
  - ・既定の元のサイズは、最大化時のサイズまたはこれに近いサイズにしないでください。代わりに、全画面表示にしなくとも最も使いやすい一般的なサイズを既定のサイズに指定します。ユーザーが全画面表示にする場合、サイズ変更ではなくウィンドウの最大化を実行することに注意してください。
  - ・コンテンツを使用できなくなるサイズがある場合は、最小 ウィンドウ サイズを設定する必要があります。サイズ変更可能なコントロールに対しては、機能を維持できる最小サイズを設定します(リスト ビューが役目を果たすことができる最小の列幅など)。
  - ・より小さいサイズでもコンテンツを使用できる場合は、表示を変更する必要があります。



この例では、ウィンドウが小さくなりすぎて標準形式で表示できない場合に、Windows Media® Player の表示形式が変更されています。

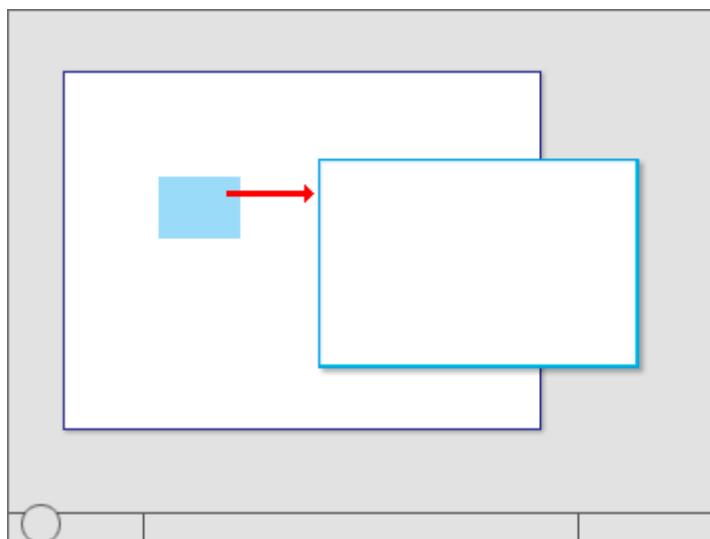
## ウィンドウの位置

- ・以下のガイドラインでは、"中央配置" は、垂直方向についてモニターの中央やや上方向寄りに配置することを表し、完全に中央に配置することは表しません。モニターまたはオーナー ウィンドウの上端とウィンドウの上端の間を 45%、モニターまたはオーナー ウィンドウの下端とウィンドウの下端の間を 55% にします。この配置は、視線が自然に画面の上方向に偏るために行います。

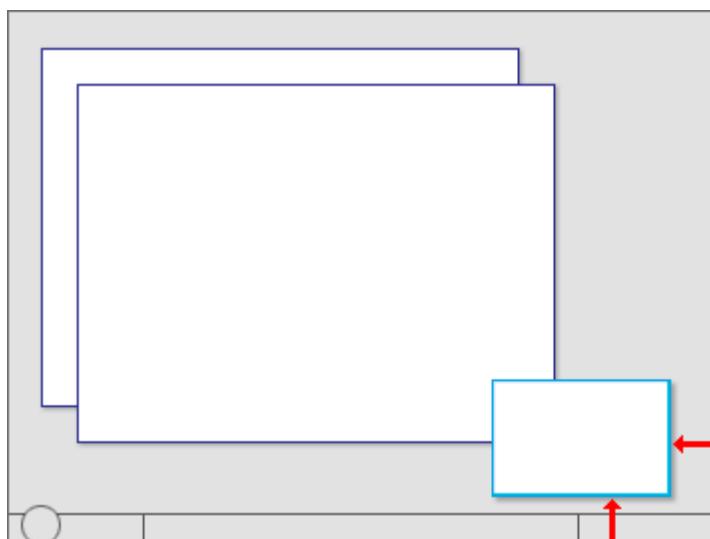


"中央配置" は、垂直方向についてモニターの中央やや上方向寄りに配置することを表す。

- コンテキスト ウィンドウの場合は、必ず起動元オブジェクトの近くに表示します。表示するウィンドウで起動元のオブジェクトが隠れることがないように、適切な位置に配置します。
  - マウスを使用して表示する場合、可能であれば、右下にオフセットして表示します。

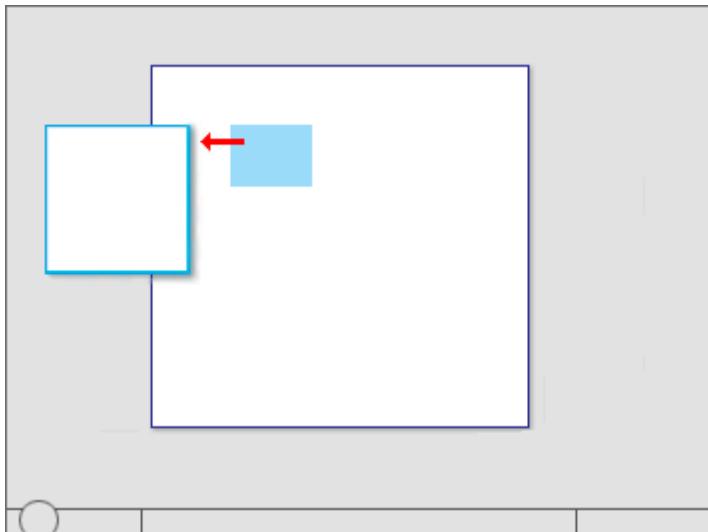


コンテキスト ウィンドウは、起動元オブジェクトの近くに表示します。



通知領域アイコンから起動されるウィンドウは、通知領域の近くに表示されます。

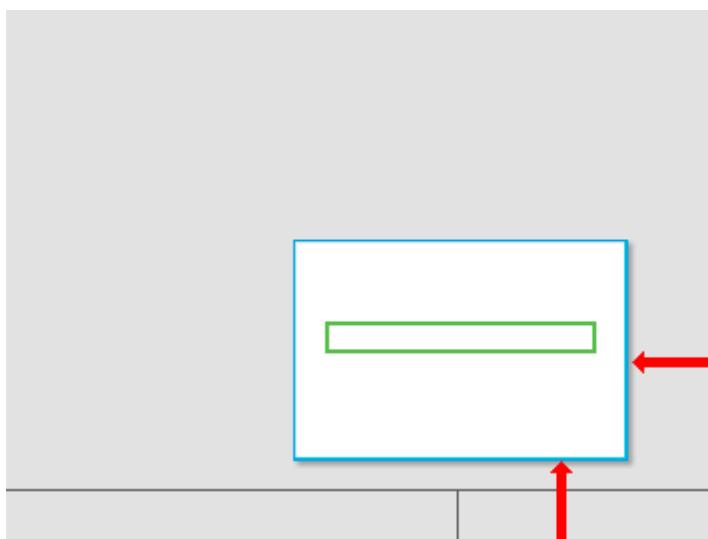
- ペンを使用して表示する場合、可能であれば、ユーザーの手で隠れることがないように配置します。右利きのユーザーであれば左側に、左利きのユーザーであれば右側に表示します。



ペンを使用する場合は、コンテキスト ウィンドウがユーザーの手で隠れることがないように配置します。

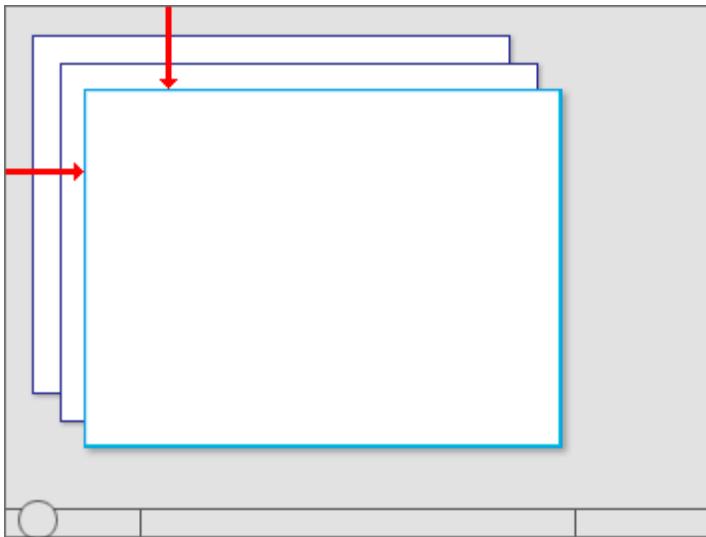
開発者向け情報: マウス イベントとペンイベントは、[GetMessageExtraInfo](#) API を使用して判別できます。また、ユーザーの利き手は、[SystemParametersInfo](#) API と SPI\_GETMENUDROPALIGNMENT を使用して判断できます。

- 進行状況ダイアログ ボックスは、アクティブ モニターの右下の適切な位置に配置します。



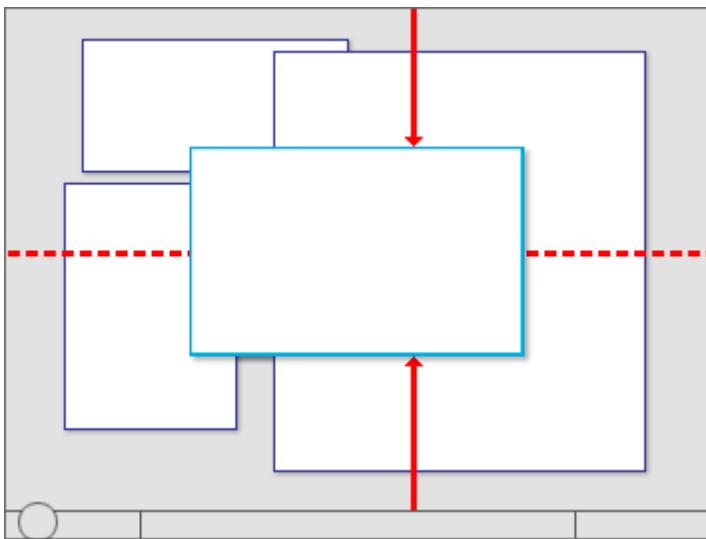
進行状況ダイアログ ボックスは右下に配置します。

- 現在のコンテキストやユーザーの操作に関係がないウィンドウは、現在のポインターの位置から離れた場所に配置します。これにより、偶発的な対話操作を防止できます。
- 最上位のアプリケーションまたはドキュメントのウィンドウは、モニターの左上から離れていく位置に、始点を重ねて表示します。アクティブ プログラムで作成された場合はアクティブ モニターを使用し、それ以外の場合は既定のモニターを使用します。



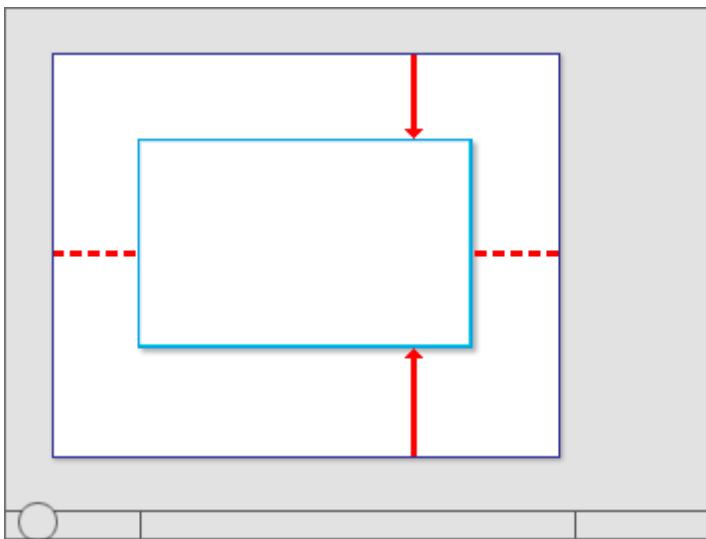
最上位のアプリケーションまたはドキュメントのウィンドウは、モニターの左上から離れていく位置に重ねて表示します。

- 最上位のユーティリティのウィンドウの場合は、必ずモニターの“中央配置”にします。アクティブ プログラムで作成された場合はアクティブ モニターを使用し、それ以外の場合は既定のモニターを使用します。



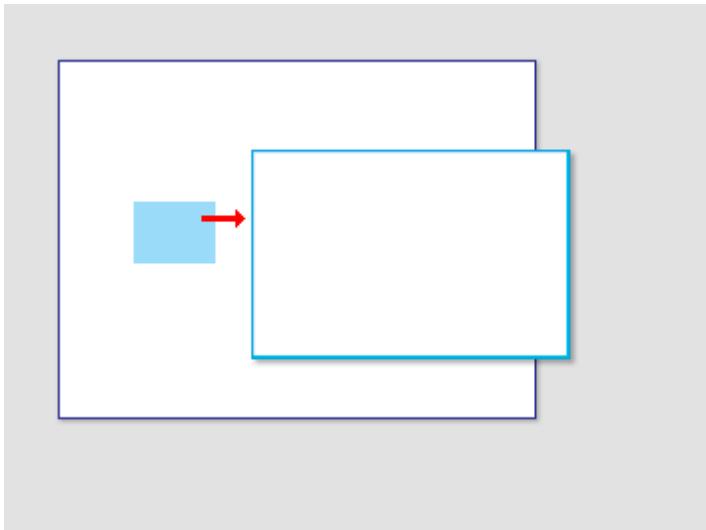
最上位のユーティリティのウィンドウは中央配置にします。

- 所有されるウィンドウの初期表示は、オーナー ウィンドウ前面の“中央配置”にします。その後の表示については、最後に表示された位置(オーナー ウィンドウに対する相対的な位置)に表示する方が使いやすいと思われる場合は、その表示方法も検討します。



所有されるウィンドウの初期表示は、オーナー ウィンドウ前面の中央配置にします。

- モードレス ダイアログ ボックスの初期表示は、見つけやすいように、オーナー ウィンドウの前面に配置します。ただし、ユーザーがオーナー ウィンドウをアクティブ化した場合に、モードレス ダイアログ ボックスを非表示にすることもあります。



モードレス ダイアログ ボックスの初期表示は、見つけやすいように、オーナー ウィンドウの前面に配置します。

- 表示先のモニターでウィンドウ全体を表示できるように、必要に応じて初期表示位置を調整します。サイズ変更可能ウィンドウのサイズが表示先モニターより大きい場合は、表示できるようにサイズを小さくします。

#### ウィンドウの順序 (Z オーダー)

- 所有されるウィンドウは、必ずオーナー ウィンドウの前面に配置します。所有されるウィンドウをオーナー ウィンドウの背面に配置しないでください。ほとんどの場合、ユーザーはこのウィンドウを見ることができません。
- z オーダーに関するユーザーの選択を尊重します。ユーザーがウィンドウを選択した場合は、プログラムのインスタンスに関連付けられているウィンドウ (選択されたウィンドウとそのオーナーまたは所有されるウィンドウ) のみを z オーダーの最上位に移動します。同一プログラムの別のインスタンスなど、他のウィンドウの順序は変更しません。

#### ウィンドウのアクティブ化

- ウィンドウの状態に関するユーザーの選択を尊重します。表示されているウィンドウに注目してもらう必要がある場合は、注意を引くためにタスクバー ボタンを 3 回点滅させてから強調表示のままにしますが、これ以外の動作は行いません。ウィンドウを元のサイズに戻したり、アクティブ化したりしないでください。また、サウンド効果も使用しないでください。その代わり、ユーザーの準備が整ったときにウィンドウをアクティブ化してもらうようにします。
  - 例外: ウィンドウがタスクバーに表示されていない場合は、代わりに、すべてのウィンドウの最前面に表示し、タイトルバーを点滅させます。
- メイン ウィンドウを元に戻した場合は、すべてのサブ ウィンドウも元に戻す必要があります。これは、サブ ウィンドウ固有のタスクバー ボタンがある場合にも該当します。元に戻す場合は、サブ ウィンドウをメイン ウィンドウの前面に表示します。

#### 入力フォーカス

- ユーザー起動操作で表示されたウィンドウに入力フォーカスを設定する必要がありますが、これはウィンドウが直ちに表示される場合に限ります (5 秒以内)。表示されたウィンドウは、入力フォーカスを一度取得できます。
  - ウィンドウの表示が遅い (5 秒を超える) と、多くの場合、ユーザーは表示を待っている間に別の作業を実行します。このような場合にフォーカスが取得されるとわずらわしく感じます。複数回繰り返される場合はなおさらです。
- 直ちに表示されないウィンドウまたはシステム起動操作によって表示されるウィンドウには、入力フォーカスを設定しません。その代わり、フォーカスを設定せずに前面に表示し、ユーザーの準備が整ったときにア

クティブ化してもらうようにします。

- 例外: 資格情報マネージャー。

## 値の保持

- ウィンドウを再表示する場合は、最後のアクセス時と同じ状態で表示することを検討します。ウィンドウを閉じるときに、使用モニター、ウィンドウ サイズ、位置、状態(最大化されているか元のサイズか)を保存してください。再表示するときには、適切なモニターを使用し、保存されたウィンドウ サイズ、位置、状態(最大化されているか元のサイズか)を復元します。また、これらの属性を、ユーザーごとに、プログラムインスタンス レベルで保持することも検討します。次の場合は例外です。
  - ユーザーが完全に最初からやり直す可能性が高いウィンドウについては、これらの属性を保存または保持しません。
  - Windows Tablet コンピューターおよびタッチ テクノロジー コンピューターで使用される可能性が高いプログラムについては、横長と縦長の両方のウィンドウの状態を保存します。詳細については、「[Designing for Varying Display Sizes \(さまざまな表示サイズ向けに設計\)](#)」を参照してください。
- 現在のモニター構成では、最後に表示された状態でウィンドウを表示できない場合は、次の処理を行います。
  - 最後に使用したモニターへのウィンドウの表示を試行します。
  - ウィンドウ サイズがモニターより大きい場合は、必要に応じてウィンドウのサイズを変更します。
  - ウィンドウがモニター内に収まるように、必要に応じて位置を左上に移動します。
  - 以上の処理で問題が解決しない場合は、ウィンドウの配置をガイドラインの既定の位置に戻します。可能な場合は、以前のサイズに戻すようにしてください。

## ダイアログ ボックス

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

全般

モーダル ダイアログ ボックス

モードレス ダイアログ ボックス

複数のダイアログ ボックス

複数ページのダイアログ ボックス

提示方法

タイトルバー

対話操作

進行状況ダイアログ ボックス

アイコンとグラフィック

コミット ボタン

コマンド リンク

今後、この<アイテム>を表示しない

[後で確認する]

詳細表示/簡易表示

脚注

コントロールの無効化または削除とエラー メッセージの表示

必須の入力

エラー処理

ヘルプ

既定値

推奨されるサイズと間隔

テキスト

ドキュメント

"ダイアログ ボックス" は、ユーザーがコマンドを実行したり、ユーザーに質問したり、情報や進行状況フィードバックを提供するためのサブ ウィンドウです。



典型的なダイアログ ボックス。

ダイアログ ボックスは、タイトルバー(ダイアログ ボックスを呼び出したコマンド、機能、またはプログラムを表す)、オプションのメイン指示テキスト(ダイアログ ボックスを使用する目的を説明する)、コンテンツ エリアのさまざまなコントロール(オプションを表示する)、およびコミット ボタン(ユーザーがタスクをコミットする方法を示す)で構成されています。

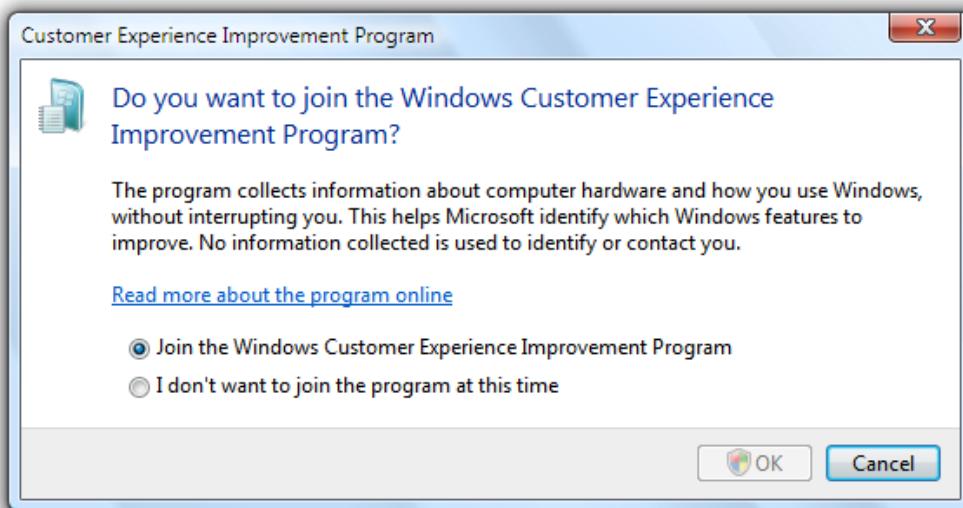
ダイアログ ボックスには、基本的に 2 種類あります。

- モーダル ダイアログ ボックス: 完了し、閉じてからでなければ、ユーザーはオーナー ウィンドウでの作業を続行できません。このダイアログ ボックスは、作業を続行する前に完了させる必要がある、重要なまたは頻度が低い、1 回限りのタスクに最も適しています。
- モードレス ダイアログ ボックス: ユーザーは、必要に応じて、ダイアログ ボックスとオーナー ウィンドウを切り替えることができます。このダイアログ ボックスは、頻度が高く、繰り返し発生する、継続的なタスクに最も適しています。

"タスク ダイアログ ボックス" は、タスク ダイアログ ボックス アプリケーション プログラミング インターフェイス (API) を使用して実装されたダイアログ ボックスです。次の要素をさまざまに組み合わせて作成できます。

- タイトルバー: ダイアログ ボックスを呼び出したアプリケーションまたはシステム機能を示します。
- メイン指示テキスト: オプションのアイコンを表示し、ダイアログ ボックスを使用する目的をユーザーに示します。
- コンテンツ エリア: 説明的な情報とコントロールを表示します。
- コマンド エリア: [キャンセル] ボタンなどのコミット ボタン、オプションの [その他のオプション] コントロール、[今後、この<アイテム>を表示しない] コントロールを表示します。

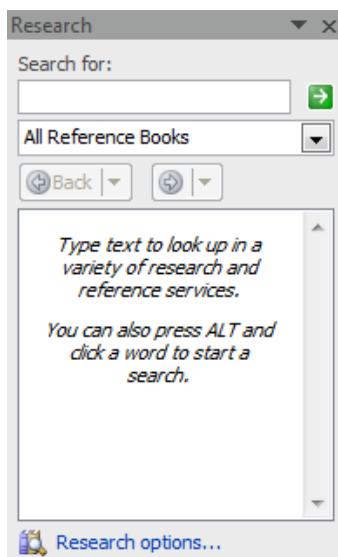
- 脚注エリア: オプションの追加説明およびヘルプを表示します。通常、使用経験の少ないユーザーを対象にしています。



典型的なタスク ダイアログ ボックス。

タスク ダイアログ ボックスは作成しやすく、外観に一貫性を持たせることができるために、適切な場合は常にタスク ダイアログ ボックスを採用することをお勧めします。タスク ダイアログ ボックスを使用するには、Windows Vista® 以降が必要であるため、以前のバージョンの Microsoft® Windows® には適しません。

"作業ウィンドウ" はダイアログ ボックスに似ていますが、別ウィンドウではなく、ウィンドウ内に表示されます。そのため、ダイアログ ボックスよりも直接的で、コンテキストに即している印象を与えます。技術的には異なりますが、作業ウィンドウはダイアログ ボックスに非常に似ているため、このトピックでは両方のガイドラインを紹介します。



典型的な作業ウィンドウ。

**プロパティ ウィンドウ**は特殊なダイアログ ボックスで、オブジェクト、オブジェクトのコレクション、またはプログラムのプロパティを表示および変更するために使用します。また、通常、プロパティ ウィンドウでは複数のタスクをサポートしますが、ダイアログ ボックスでは1つのタスクまたはタスクの1つのステップをサポートします。使用方法が特殊なため、プロパティ ウィンドウについては、別のガイドラインで説明します。

ダイアログ ボックスには、[タブ](#)を追加できます。追加した場合は、"タブ付きダイアログ ボックス"と呼ばれます。プロパティ ウィンドウかどうかは、タブの使用ではなく、プロパティが表示されているかどうかによって決まります。

**注:** レイアウト、ウィンドウの管理、コモン ダイアログ ボックス、プロパティ ウィンドウ、ウィザード、確認、エラー メッセージ、警告 メッセージに関するガイドラインは、それぞれ別の項目として記載しています。

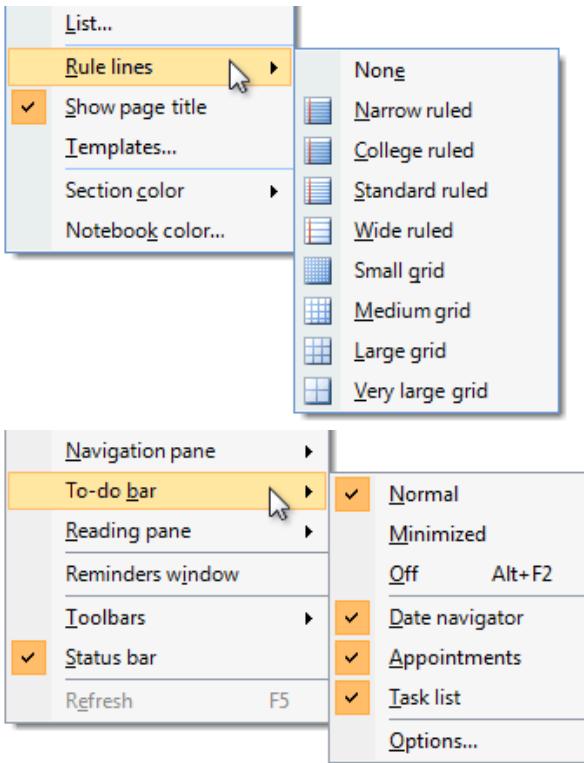
## 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- ユーザーに情報を提供する、質問をする、またはユーザーがオプションを選択してコマンドやタスクを実行できるようにする目的かどうか。該当しない場合は、別のユーザーインターフェイス(UI)を使用します。
- オブジェクト、オブジェクトのコレクション、またはプログラムのプロパティを表示および変更することが目的かどうか。該当する場合は、代わりに[プロパティ ウィンドウ](#)または[ツールバー](#)を使用します。
- コマンドまたはツールのコレクションを提示することが目的かどうか。該当する場合は、ツールバーまたは[パレット ウィンドウ](#)を使用します。
- ユーザーが操作を継続することの確認が目的かどうか。継続しない明確な理由があるかどうか、およびユーザーが継続しない可能性が十分にあるかどうか。該当する場合は、[確認](#)を使用します。
- エラーメッセージまたは警告メッセージを表示することが目的かどうか。該当する場合は、[エラー メッセージ](#)または[警告メッセージ](#)を使用します。
- 目的が、次のいずれかであるかどうか。
  - ファイルを開く
  - ファイルを保存する
  - フォルダーを開く
  - テキストを検索または置換する
  - ドキュメントを印刷する
  - 印刷するページの属性を選択する
  - フォントを選択する
  - 色を選択する
  - ファイル、フォルダー、コンピューター、またはプリンターを参照する
  - Microsoft Active Directory® でユーザー、コンピューター、またはグループを検索する
  - ユーザー名とパスワードの入力を求める

該当する場合は、代わりに適切な[コモン ダイアログ ボックス](#)を使用します。コモン ダイアログ ボックスの多くには、拡張性があります。

- 複数のウィンドウを必要とする複数のステップから成るタスクを実行することが目的かどうか。該当する場合は、代わりに[タスク フロー](#)または[ウィザード](#)を使用します。
- ユーザーがすばやく対応する必要がなく、無視することもできる、現在のユーザー操作に関連しないシステム イベントまたはプログラム イベントをユーザーに通知することが目的かどうか。該当する場合は、代わりに[通知](#)を使用します。
- プログラムの状態を表示することが目的かどうか。該当する場合は、代わりに[ステータスバー](#)を使用します。
- インプレース UI を使用する方が適しているかどうか。ダイアログ ボックスは注意を促すので、ユーザーの作業を中断されることになります。現在のコンテキストの外部でユーザーが操作を実行する必要がある場合など、作業を中断することが妥当な場合があります。また、インプレース UI(作業ウィンドウなど)に直接表示するか、必要に応じて[段階的表示](#)を使用して、コンテキスト内で UI を表示する方が適切な場合もあります。
- それほど重大ではないユーザー入力の問題または特殊な状態を表示することが目的かどうか。該当する場合は、代わりに[パルーン](#)を使用します。
- タスク フローの場合は、別のページを使用する方が適しているかどうか。通常は、タスクが 1 つのウィンドウ内でページからページへ進行するようにします。ダイアログ ボックスは、インプレース コマンドの確認、インプレース コマンドの入力の取得、主要なタスク フローの外部で個別に実行することが適している独立したサブタスクの実行に使用します。
- オプションを選択する場合は、ユーザーがオプションを変更する可能性があるかどうか。該当しない場合は、次のような代替手段を検討します。
  - ユーザーに確認せずに既定のオプションを使用し、後からユーザーが変更できるようにする。
  - オプションを提示するバージョン(たとえば、メニューの [印刷...])とオプションを提示しないバージョン(たとえば、ツールバーの [印刷])を用意する。通常、ツールバーのコマンドは即座に処理し、ダイアログ ボックスを表示しないようになります。
- オプションを選択する場合は、オプションを提示するシンプルで直接的な方法があるかどうか。該当する場合は、次のような代替手段を検討します。
  - さまざまなコマンドを選択する場合は、[分割ボタン](#)を使用する。
  - コマンド、チェック ボックス、ラジオ ボタン、および簡単な一覧を備えたサブメニューを使用する。



これらの例では、ダイアログ ボックスの代わりにサブメニューを使用して、シンプルな選択肢を提示しています。

## デザイン コンセプト

ダイアログ ボックスを適切に使用すると、簡単にプログラムに機能を追加し、柔軟性を与えることができます。適切に使用しないと、ユーザーの手をわざらわせ、作業の流れを止め、プログラムを使用するのが面倒だという印象を与えることになります。モーダル ダイアログ ボックスは、ユーザーの注意を促します。ダイアログ ボックスは、他の代わりになる UI より簡単に実装できる場合が多いため、使いすぎ傾向があります。

ダイアログ ボックスは、使用状況に合わせた[設計特性](#)にするのが最も効果的です。ダイアログ ボックスの設計の大部分は、目的(オプションを提示する、質問する、情報またはフィードバックを提供する)、種類(モーダルまたはモードレス)、およびユーザー操作(必須の応答、任意の応答、または確認)によって決まります。使用状況の大部分は、コンテキスト(開始するのがユーザーか、プログラムか)、ユーザー操作の可能性、および表示頻度によって決まります。

効果的なダイアログ ボックスを設計するには、次の要素を効果的に使用します。

- [ダイアログ ボックスのテキスト](#)
- [メイン指示テキスト](#)
- [\[今後、この<アイテム>を表示しない\] オプション](#)

### 最も重要な点

ダイアログ ボックスの設計(目的、種類、およびユーザー操作によって決まる)が使用状況(コンテキスト、ユーザー操作の可能性、および表示頻度によって決まる)に合っていることを確認します。

詳細と例については、「[ダイアログ ボックスのデザイン コンセプト](#)」を参照してください。

## 使用パターン

ダイアログ ボックスには、いくつかの使用パターンがあります。

- **質問ダイアログ ボックス(ボタンを使用):** ユーザーに1つ質問をするか、コマンドの確認を求め、簡単な応答を水平に配置されたコマンド ボタンで提示します。
- **質問ダイアログ ボックス(コマンド リンクを使用):** ユーザーに1つ質問をするか、実行するタスクの選択を求め、詳細な応答を垂直に配置されたコマンド リンクで提示します。
- **選択ダイアログ ボックス:** 一連の選択肢をユーザーに提示します。通常、コマンドをより正確に指定するために使用します。質問ダイアログ ボックスとは異なり、選択ダイアログ ボックスでは複数の質問をすることができます。
- **進行状況ダイアログ ボックス:** 長い時間がかかる処理(5秒を超える)の進行状況のフィードバックと処理の取り消しや停止のためのコマンドをユーザーに提示します。
- **情報ダイアログ ボックス:** ユーザーが要求した情報を表示します。

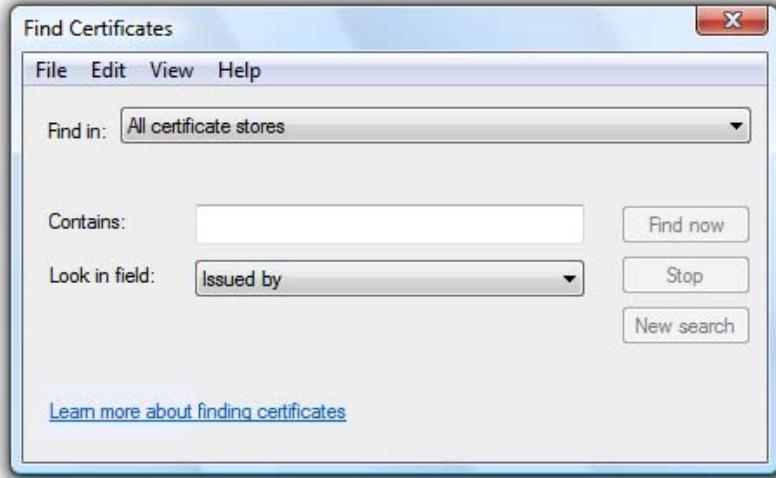
詳細と例については、「[ダイアログ ボックスの使用パターン](#)」を参照してください。

## ガイドライン

### 全般

- スクロール可能なダイアログ ボックスは使用しません。通常の使用で、全体を確認するためにスクロールバーを使用する必要があるダイアログ ボックスを使用しないでください。代わりに、ダイアログ ボックスを設計し直します。[段階的表示](#)または[タブ](#)の使用を検討します。
- メニュー バーやステータス バーを使用しません。代わりに、ダイアログ ボックス自体から直接、または関連するコントロールのコンテキストメニューを使用して、コマンドおよび状態にアクセスできるようにします。
  - 例外: ダイアログ ボックスを使用してメイン ウィンドウ(ユーティリティなど)を実装する場合は、メニュー バーを使用しても問題ありません。

間違った例:



この例では、[証明書の検索]が、メニュー バーを使用したモードレス ダイアログ ボックスになっています。

- ダイアログ ボックスにすぐに注意を払う必要がなく、プログラムがアクティブではない場合は、注意を引くためにタスク バー ボタンを 3 回点滅させてから強調表示にします。これ以外の動作は行いません。ウィンドウを元のサイズに戻したり、アクティビティ化したり、音を鳴らしたりしないでください。代わりに、ユーザーが選択したウィンドウの状態を尊重し、準備が整つたときにユーザーがウィンドウをアクティビティ化できるようにします。

その他のガイドラインと例については、「[タスク バー](#)」を参照してください。

### モーダル ダイアログ ボックス

- 作業を続行する前に完了させる必要がある、重要なまたは頻度が低い、1 回限りのタスクに使用します。
- 明示的にコミットされるまで変更が反映されないように、[遅延型のコミット モデル](#)を使用します。
- 外観に一貫性を持たせるために、適切な場合は常にタスク ダイアログ ボックスを使用して実装します。タスク ダイアログ ボックスを使用するには、Windows Vista® 以降が必要なため、以前のバージョンの Windows には適しません。

### モードレス ダイアログ ボックス

- 頻度が高く、繰り返し発生する、継続的なタスクに使用します。
- 即座に変更が反映されるように、[即時型のコミット モデル](#)を使用します。
- モードレス ダイアログ ボックスの場合、ウィンドウを閉じるための明示的な [閉じる] コマンド ボタンをダイアログ ボックスに使用します。いずれの場合も、ウィンドウを閉じるためにタイトルバーの [閉じる] ボタンを使用します。
- ドッキング可能なモードレス ダイアログ ボックスの作成を検討します。モードレス ダイアログ ボックスをドッキングできるようにすると、より柔軟に配置できるようになります。



Microsoft Office に使用されているモードレス ダイアログ ボックスには、ドッキング可能なものがあります。

### 複数のダイアログ ボックス

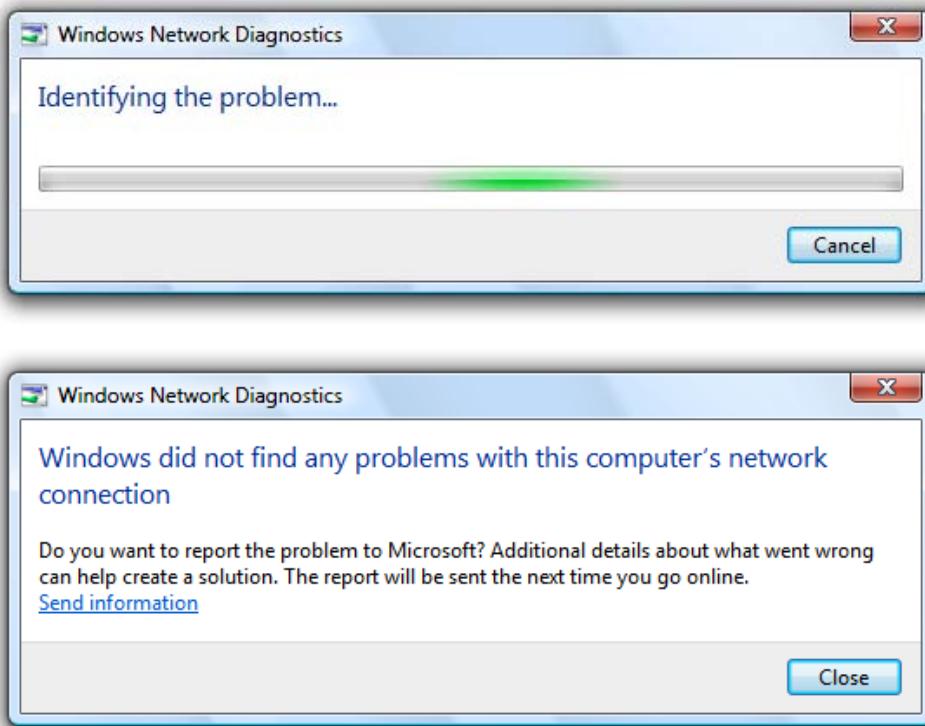
- 親選択ダイアログ ボックスから、一度に複数の子選択ダイアログ ボックスを表示しないようにします。複数のダイアログ ボックスを表示すると、コミット ボタンの意味が理解しづらくなります。必要に応じて、他の種類のダイアログ ボックス(質問ダイアログ ボックスなど)を表示してください。

- 関連する一連のダイアログ ボックスの場合、可能であれば、複数ページのダイアログ ボックスを使用します。明確な関連性がない場合は、個別のダイアログ ボックスを使用します。

#### 複数ページのダイアログ ボックス

- 次のような "関連する" 一連のページがある場合は、個別のダイアログ ボックスの代わりに、複数ページのダイアログ ボックスを使用します。
  - 単一の入力ページ (オプション)
  - 進行状況ページ
  - 単一の結果ページ

タスクが別の場所から開始される場合があるため、入力ページはオプションです。こうすることで、操作性が安定し、簡単で、軽快になります。



この例では、進行状況ページと結果ページで構成された [Windows ネットワーク診断] ダイアログ ボックスを示しています。

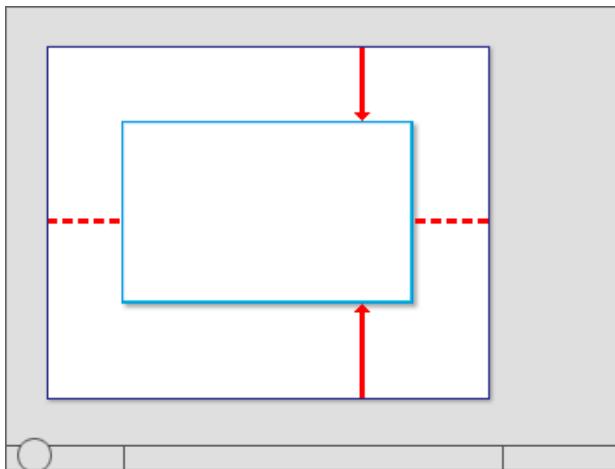
- 入力ページが標準のダイアログ ボックスである場合は、複数ページのダイアログ ボックスを使用しません。この場合は、標準のダイアログ ボックスを一貫して使用することが重要になります。
- [戻る] ボタンと [次へ] ボタンを使用せず、4 ページ以上にならないようにします。複数ページのダイアログ ボックスは、フィードバックを提供する 1 つのステップから成るタスクに使用します。複数のステップから成るタスクに使用する ウィザード ではありません。ウィザードは、複数ページのダイアログ ボックスに比べ、重く、直接的でないという印象を与えます。
- 入力ページでは、タスクを開始するための具体的なコマンド ボタンまたはコマンド リンクを使用します。
- 入力ページと進行状況ページには [キャンセル] ボタンを使用し、結果ページには [閉じる] ボタンを使用します。

開発者向け情報: 複数ページのタスク ダイアログ ボックスの作成には、[TDM\\_NAVIGATE\\_PAGE](#) メッセージを使用します。

#### 提示方法

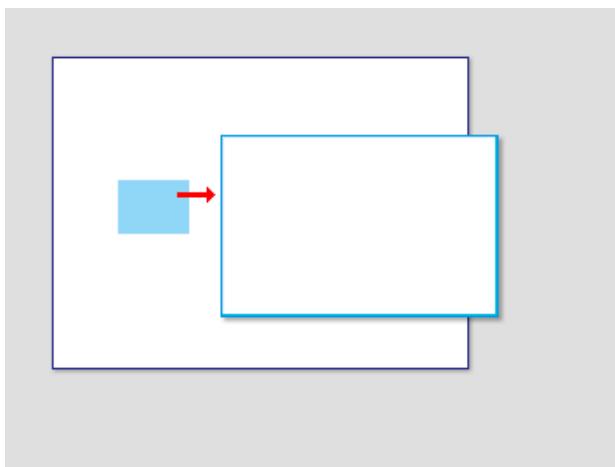
簡単に検索してアクセスできるダイアログ ボックスを作成するには、呼び出し元にダイアログ ボックスを明確に関連付け、複数のモニターでうまく動作するようにします。

- ダイアログ ボックスの初期表示は、オーナー ウィンドウ前面の "中央配置" にします。その後の表示については、最後に表示された位置 (オーナー ウィンドウに対する相対的な位置) に表示する方が使用しやすいと思われる場合は、その表示方法も検討します。



ダイアログ ボックスの初期表示は、オーナー ウィンドウ前面の中央配置にします。

- コンテキストが重要なダイアログ ボックスの場合は、起動元オブジェクトの近くに表示します。ただし、表示するダイアログ ボックスで起動元のオブジェクトが隠れることがないように、適切な位置に配置します(できれば右下にオフセットします)。



オブジェクトのプロパティはオブジェクトの近くに表示します。

- モードレス ダイアログ ボックスの初期表示は、見つけやすいように、オーナー ウィンドウの前面に配置します。ユーザーがオーナー ウィンドウをアクティブ化した場合に、モードレス ダイアログ ボックスを非表示にすることもあります。
- 表示するモニターでダイアログ ボックス全体を表示できるように、必要に応じて初期表示位置を調整します。サイズ変更可能 ウィンドウのサイズが表示先モニターより大きい場合は、表示できるようにサイズを小さくします。
- ダイアログ ボックスを再表示する場合は、最後のアクセス時と同じ状態で表示させることを検討します。ダイアログ ボックスを閉じるときに、使用モニター、ウィンドウ サイズ、位置、状態(最大化されているか元のサイズか)を保存してください。再表示するときには、適切なモニターを使用し、保存されたダイアログ ボックス サイズ、位置、状態(最大化されているか元のサイズか)を復元します。また、これらの属性を、ユーザーごとに、プログラム インスタンス レベルで保持することも検討します。
- ウィンドウ サイズが変更可能で、コンテンツを使用できなくなるサイズがある場合は、最小 ウィンドウ サイズを設定する必要があります。サイズが小さくてもコンテンツを使用できるように提示方法を変更することを検討します。



この例では、Windows Media Player® が小さくされすぎて標準の形態を維持できなくなると、形態が変わることを示しています。

- [常に手前に表示] 属性は使用しません。
  - 例外: ダイアログ ボックスで実質的にモーダルな処理を実行しているが、オーナー ウィンドウにアクセスするために一時中断する必要がある場合にのみ使用します。たとえば、ドキュメントのスペル チェックを実行しているときに、ユーザーは [スペル チェック] ダイアログ ボックスを離れて、ドキュメントにアクセスし、間違いを訂正することができます。

詳細と例については、「[ウィンドウの管理](#)」を参照してください。

## タイトルバー

- ダイアログ ボックスには、タイトルバー アイコンを配置しません。タイトルバー アイコンは、[メイン ウィンドウとサブ ウィンドウ](#)を視覚的に区別する場合に使用します。
  - 例外: ダイアログ ボックスを使用してメイン ウィンドウ (ユーティリティなど) を実装し、タスク バーに表示される場合は、タイトルバー アイコンを配置します。この場合、タスク バー上の表示を最適化するために、特徴的な情報をタイトルの最初の方に簡潔にまとめるようにします。
- ダイアログ ボックスには [閉じる] ボタンを常に配置します。モードレス ダイアログ ボックスには、[最小化] ボタンも配置します。サイズ変更可能なダイアログ ボックスには、[最大化] ボタンを配置します。
- [閉じる] ボタンは無効にしません。[閉じる] ボタンがあれば、ユーザーは自分の裁量で不必要的ウィンドウを閉じることができます。
  - 例外: 進行状況 ダイアログ ボックスで、有効な状態になるために、またはデータの消失を防止するためにタスクを完了するまで実行する必要がある場合は、[閉じる] ボタンを無効にします。
- このダイアログ ボックスでは、タイトルバーの [閉じる] ボタンをクリックした場合に、[キャンセル] ボタンまたは [閉じる] ボタンと同じ結果になるようにする必要があります。[OK] ボタンと同じ結果にならないようにします。
- タイトルバーのキャプションおよびアイコンが既にウィンドウの上部付近に明確に表示されている場合は、冗長にならないように、タイトルバーのキャプションおよびアイコンを非表示にします。ただし、Windows が内部で使用するタイトルを適切に設定する必要があります。

## 対話操作

- ユーザーが開始したダイアログ ボックスは、表示されたときに入力フォーカスを常に取得します。プログラムによって開始されたダイアログ ボックスは入力フォーカスを取得しません。これは、ユーザーが別のウィンドウを操作している可能性があるためです。誤ってフォーカスがダイアログ ボックスに移ると、意図しない結果が生じる場合があります。
  - ユーザーが最初に操作する可能性の高いコントロールに最初の入力フォーカスを割り当てます。通常は ("必ず" ではありません)、最初の対話型コントロールに割り当てます。最初の入力フォーカスをヘルプ リンクに割り当てないでください。
  - キーボード ナビゲーションの場合、タブ オーダーは論理的な順序 (一般的に、左から右、上から下) に設定します。通常、タブ オーダーは読む順序に従いますが、次の例外を設けることを検討します。
    - 最もよく使用されるコントロールはタブ オーダーの先頭の方に配置する。
    - ヘルプ リンクはダイアログ ボックスの下部に配置し、タブ オーダーをコミット ボタンより後に設定する。
- ユーザーは目的があってダイアログ ボックスを表示しているということを前提にして、タブ オーダーを設定します。たとえば、選択 ダイアログ ボックスを表示するのは選択するためであり、確認し、[キャンセル] をクリックするためではありません。
- Esc キーを押すと、アクティブなダイアログ ボックスは常に閉じます。これは、[キャンセル] または [閉じる] が配置されているダイアログ ボックスでも、結果を元に戻すことができないため [キャンセル] の名前が [閉じる] に変更されている場合でも同じです。

## アクセス キー

- 可能な限り、すべての対話型コントロールまたはそのラベルに一意のアクセス キーを割り当てます。[読み取り専用のテキスト ボックス](#)は対話型コントロールである (ユーザーがスクロールし、テキストをコピーできる) ため、アクセス キーを割り当てるメリットがあります。以下には、アクセス キーを割り当てないでください。
  - [OK]、[キャンセル] および [閉じる] の各ボタン。Enter キーおよび Esc キーが、それぞれのボタンのアクセス キーに使用されています。ただし、"OK" または "キャンセル" を意味する異なるラベルのコントロールには、アクセス キーを常に割り当てます。



この例では、肯定的なコミット ボタンにアクセス キーを割り当てています。

- グループ ラベル。通常、グループ内の各コントロールにはアクセス キーが割り当てられているため、グループ ラベル

に割り当てる必要はありません。ただし、アクセスキーが不足している場合は、グループラベルにアクセスキーを割り当て、各コントロールに割り当てないようにします。

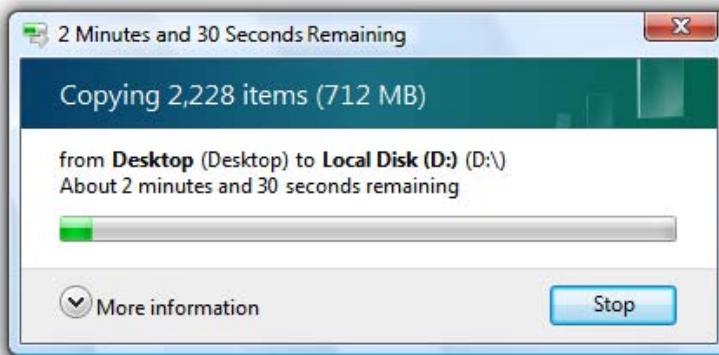
- 汎用的なヘルプボタン。ヘルプには F1 キーを押してアクセスします。
- リンクラベル。リンクが多すぎて一意のアクセスキーを割り当てられない場合がよくあります。また、リンクを表すために下線をよく使用するため、アクセスキーを示す下線がわからなくなります。代わりに、リンクのアクセスには Tab キーを使用します。
- タブ名。タブ間を順番に移動するには、Ctrl + Tab キーおよび Ctrl + Shift + Tab キーを使用します。
- 「...」というラベルが付けられている参照ボタン。これらの参照ボタンにはアクセスキーを一意に割り当てることができません。
- ラベルのないコントロール。スピンコントロール、グラフィックコマンドボタン、ラベルのない段階的表示コントロールなどがあります。
- ラベルのない静的テキストまたは対話型ではないコントロールのラベル。進行状況バーなどがあります。
- 可能な限り、標準的なアクセスキーの割り当てに従って、よく使用されるコマンドにアクセスキーを割り当てます。アクセスキーを一貫して常に割り当てるとはできません。しかし、特に、頻繁に使用されるダイアログボックスには一貫性を持たせることをお勧めします。
- 標準的なアクセスキー割り当てになるように、最初にコミットボタンにアクセスキーを割り当てます。標準的なアクセスキー割り当てが存在しない場合は、最初の文字を使用します。たとえば、[はい]コミットボタンおよび[いいえ]コミットボタンのアクセスキーは、ダイアログボックスの他のコントロールに関係なく、常に "Y" および "N" にする必要があります。
- アクセスキーを簡単に見つけることができるよう、ラベルの前の方の文字(最初の文字が理想的)をアクセスキーに割り当てます。ラベルの後半にキーワードが含まれている場合も同様です。
- できるだけ幅の広い文字を使用します。たとえば、w、m、大文字などを使用します。
- できるだけ特徴のある子音または母音を使用します。たとえば、Exit(終了)の "x" などを使用します。
- 下線が見えにくくなる文字を使用しないようにします。以下に、問題点が大きいものから順に示します。
  - 1ピクセルしか幅のない文字(i、lなど)。
  - ディセンダーのある文字(g、j、p、q、yなど)。
  - ディセンダーのある文字の隣にある文字。

他のガイドラインと例については、「[キーボード](#)」を参照してください。

## 進行状況ダイアログボックス

時間がかかるタスクの場合は、タスクが完了するまでの間にユーザーが別の作業を行うことを想定します。タスクを無人で実行できるように設計します。

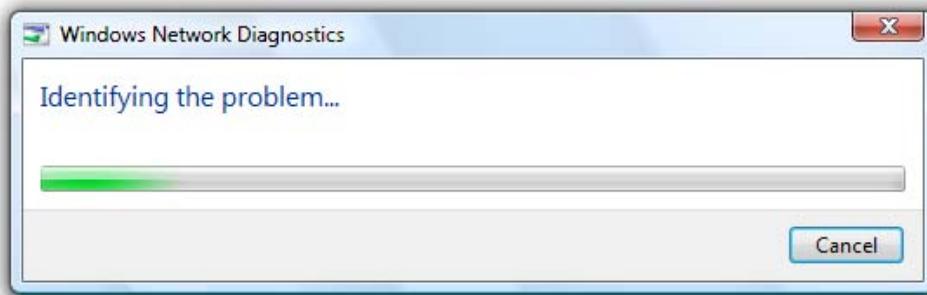
- 処理が完了するのに 5 秒よりも長くかかる場合は、進行状況フィードバックダイアログボックスを表示します。その際に、処理の取り消しや停止のためのコマンドも一緒に表示します。
  - 例外: ウィザードとタスクフローでは、タスクが(別のページに移動することなく)同じページで実行される場合のみ、進行状況の表示にモダルダイアログボックスを使用して、待っている間にユーザーが別の作業をできないようにします。それ以外の場合は、進行状況ページまたはインプレースの進行状況を使用します。
- 処理に時間がかかるタスク(30 秒を超える)をバックグラウンドで実行できる場合は、ユーザーがその間にプログラムを継続して使用できるように、モードレスな進行状況ダイアログボックスを使用します。
- モードレス進行状況ダイアログボックスの場合は、次のように設計します。
  - タイトルバーに [最小化] ボタンを配置します。
  - タスクバーに表示されるようにします。
- モードレス進行状況ダイアログボックスは、オーナーウィンドウを閉じても、継続して実行できるように実装します。



この例では、オーナーウィンドウを閉じても、ファイルのコピーは継続されます。

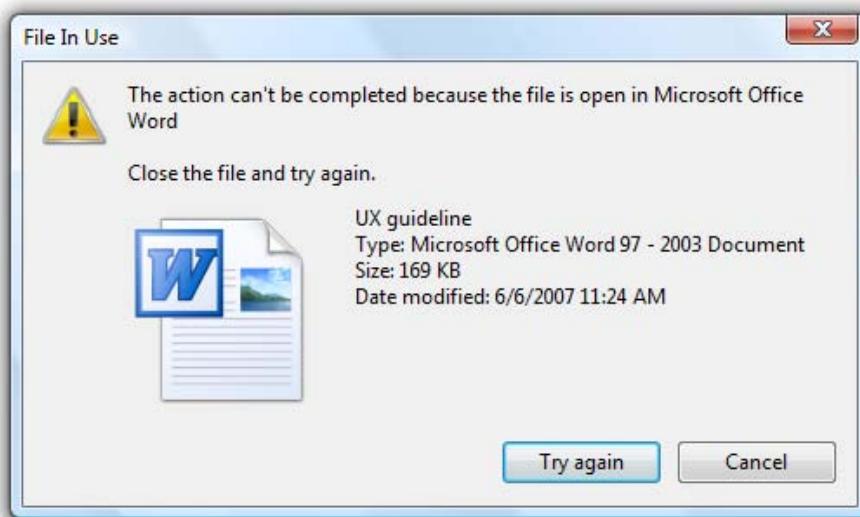
- 完了まで数秒以上かかる処理、または完了しない可能性のある処理の場合は、処理を停止するためのコマンドボタンを提供します。キャンセルすると何の副次的な影響もなく環境が以前の状態に戻る場合は、ボタンに "キャンセル" というラベルを付け

ます。これ以外の場合は、部分的に処理が完了した状態になることを示すために、ボタンに "停止" というラベルを付けます。処理のある時点で環境を元の状態に戻すことができなくなる場合は、処理の途中でボタンのラベルを "キャンセル" から "停止" に変更します。



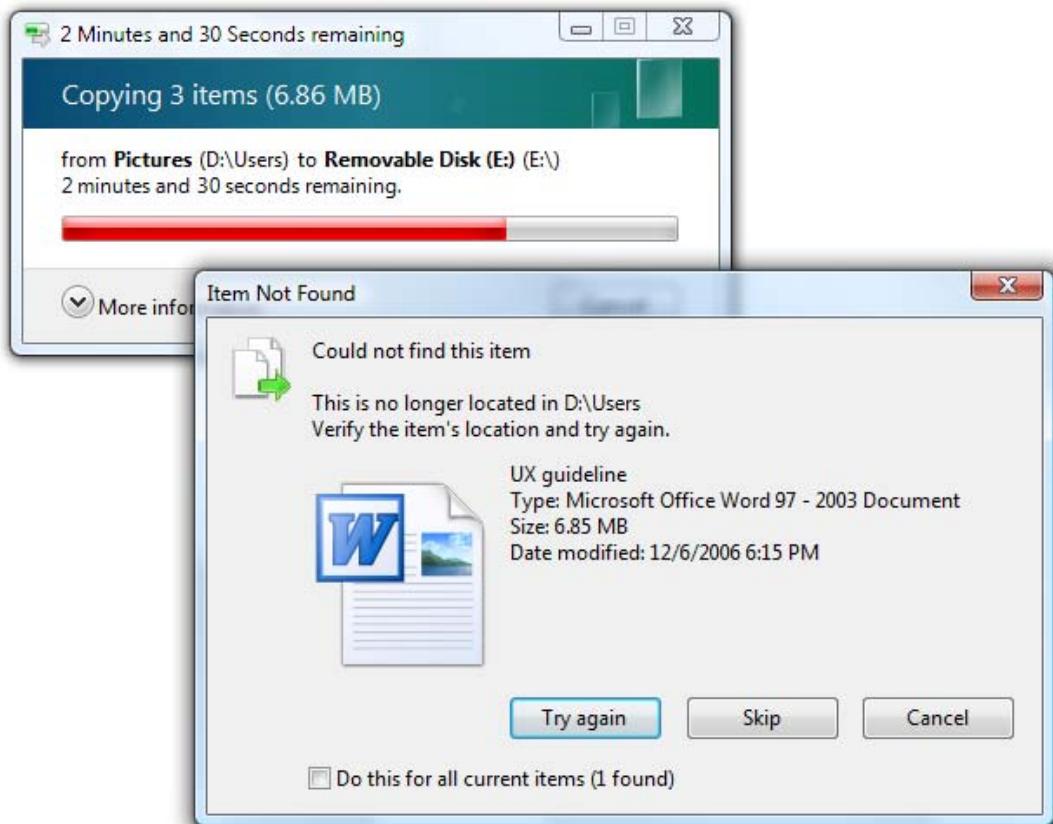
この例では、問題の診断を停止しても副次的な影響はありません。

- 処理が完了するまでに数分以上かかり、ユーザーの作業効率が低下する場合は、処理を一時停止するためのコマンド ボタンを用意します。こうすることで、ユーザーはタスクの完了と作業の完了のどちらかを選択する必要がなくなります。
- タスクを開始する前に、できる限り多くの情報を収集します。
- 回復可能な問題を検出した場合は、タスクの最後に、検出したすべての問題をユーザーに処理させるようにします。この方法が現実的でない場合は、問題が発生したときに処理させるようにします。
- 回復可能なエラーが発生したという理由で、タスクを破棄しないでください。



この例では、Windows エクスプローラーで回復可能なエラーが発生した後、ユーザーはタスクを続行できます。

- 進行状況バーを赤に変えることで、問題が発生したこと通知します。



この例では、ファイルのコピー中にリムーバブルディスクが取り外されました。

- 結果がユーザーに明らかにわかる場合は、正常に完了したときに自動的に進行状況ダイアログボックスを閉じます。それ以外の場合は、次のように、フィードバックを使用して問題の報告のみを行います。
  - 簡単なフィードバックを表示するには、進行状況ダイアログボックスにフィードバックを表示し、[キャンセル]ボタンを[閉じる]ボタンに変更します。
  - 詳細なフィードバックを表示するには、進行状況ダイアログボックスを閉じて、[情報ダイアログボックス](#)を表示します。

完了したことをフィードバックするための通知は使用しません。進行状況ダイアログボックスまたは[操作の成功通知](#)のどちらかを使用します。両方を使用しないでください。

#### 残り時間

- 以下の時刻形式を使用します。以下のうち最大時間単位がゼロでない形式から始めて、最大時間単位がゼロになった時点で次の形式に移行します。

進行状況バーの場合:

関連情報をコロン形式で表示する場合:

残り時間: h 時間 m 分

残り時間: m 分 s 秒

残り時間: s 秒

画面領域を優先する場合:

残り h 時間 m 分

残り m 分 s 秒

残り s 秒

その他の場合:

残り h 時間 m 分です

残り m 分 s 秒です

残り s 秒です

タイトルバーの場合:

hh:mm 残り

mm:ss 残り

0:ss 残り

このコンパクトな形式で重要な情報を最初に示すことで、タスクバーで切り捨てられないようにします

す。

- 見積もりは正確に行いますが、正確に見せかけた情報は提示しません。最大単位が時間の場合は、有用な場合に分を提示しますが、秒は提示しません。

間違った例:

hh 時間 mm 分 ss 秒

- 見積もりを最新の状態に保ちます。残り時間の見積もりは、少なくとも 5 秒ごとに更新します。
- 残り時間に情報を集中します。これこそユーザーが最も気にする情報です。合計経過時間は、タスクが繰り返されそうなときなどの、経過時間が役立つシナリオのみで提示します。残り時間の見積もりが進行状況バーと共に表示される場合は、完了率のパーセントは表示しません。この情報は進行状況バー自体で表現されます。
- 文法的に正しくします。数値が 1 の場合は単数形を使用します(英語の場合)。

間違った例:

1 minutes, 1 seconds

- センテンススタイルの大文字使用します。

詳細と例については、「[進行状況バー](#)」を参照してください。

## アイコンとグラフィック

### グラフィック

- 見た目が良くても、場所を取るだけで意味のない大きなグラフィックは使用しないようにします。代わりに、外観はシンプルにします。

間違った例:



この例では、大きなグラフィックに意味はありません。

### タイトルバー アイコン

- ダイアログボックスには、タイトルバー アイコンを配置しません。
  - 例外: ダイアログボックスを使用してメイン ウィンドウ(ユーティリティなど)を実装し、タスクバーに表示される場合は、タイトルバー アイコンを配置します。

### 本文アイコン

- 以下の設計パターンに基づいて本文アイコンを選択します。

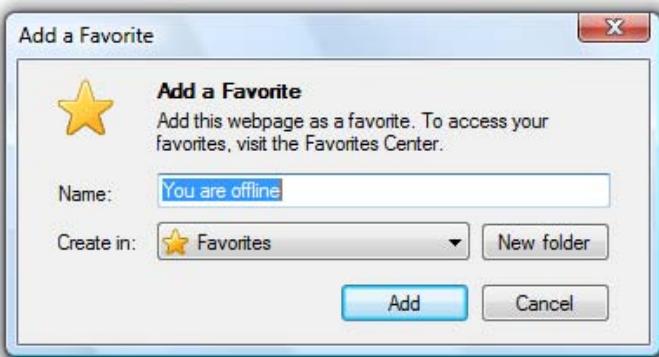
パターン	本文アイコン
質問ダイアログ ボックス	プログラム、機能、オブジェクト、警告アイコン(データが消失する可能性やシステムアクセスが失われる可能性がある場合)、セキュリティ警告、またはなし。
選択ダイアログ ボックス	なし。
進行状況ダイアログ ボックス	なし(ただし、アニメーションを表示できます)。

間違った例:



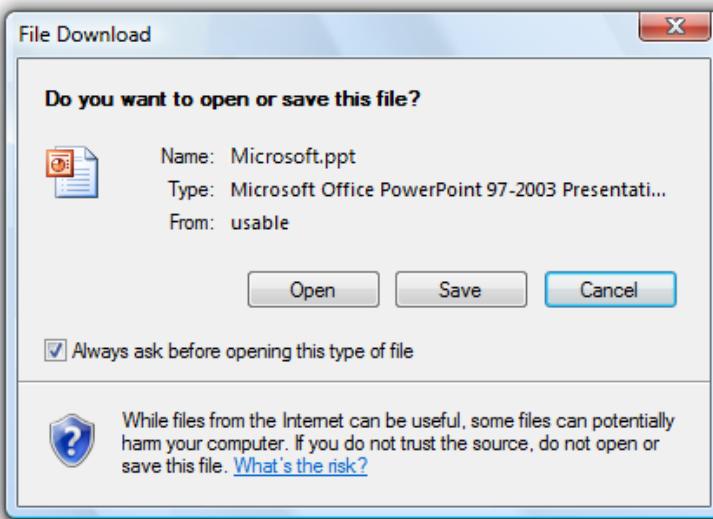
この例では、データが消失する可能性やシステムアクセスが失われる可能性がない質問に対して、警告アイコンが使用されているため、不適切です。

- ユーザーがプログラムの機能を視覚的に識別できるように、アイコンの使用を検討します。この手法は、アイコンが簡単に識別でき、プログラムのさまざまな場所で使用されている場合に、最も効果的です。



この例では、黄色い星のアイコンを使用してお気に入りを示しています。このアイコンは簡単に識別でき、お気に入りを示すためにWindowsで一貫して使用されています。

- ユーザーが問題のオブジェクトを識別できるアイコンを使用します。



この例では、ユーザーは開くまたは保存するファイルの種類をオブジェクトのアイコンから識別できます。

- ヒントがなくても機能を理解できるアイコンの使用を検討します。



この例では、これらのアイコンによってユーザーが機能の効果をイメージしやすくなっています。

- バージョン情報ダイアログ ボックスには、アプリケーションをブランド化するアイコンを使用します。



この例では、アプリケーションを識別およびブランド化するために、バージョン情報ダイアログ ボックスにビットマップを使用しています。

#### 脚注アイコン

- 脚注がある場合は、脚注の内容を要約した脚注アイコンの使用を検討します。



この例では、脚注アイコンによって、セキュリティに影響を与える問題が発生したことがわかります。

- 本文アイコンと同じような脚注アイコンは使用しません。
- 標準のエラー アイコンおよび情報 アイコンは使用しません。エラー状況は本文アイコンで表現する必要があります。脚注は常に情報を表示するために使用します。そのため、情報 アイコンを使用すると、冗長になります。ただし、危険な結果が生じることをユーザーに警告する場合は、標準の警告 アイコンおよび黄色のセキュリティ シールドを使用します。

詳細と例については、「[アイコン](#)」を参照してください。

#### コミット ボタン

注:

- コマンド リンクを使用する質問ダイアログ ボックスではボタンの代わりにコマンド リンクを使用するため、これらのガイド ラインは適用されません。
- "実行する" と "実行しない" はそれぞれ、メイン指示テキストに対する肯定的な応答と否定的な応答です。

#### 全般

- 以下に示すように、設計パターンに基づいてコミットボタンを選択します。

パターン	コミットボタン
質問ダイアログボックス(ボタンを使用する)	簡潔なコマンドセット: [はい]/[いいえ]、[はい]/[いいえ]/[キャンセル]、"実行する"/[キャンセル]、"実行する"/"実行しない"、"実行する"/"実行しない"/[キャンセル]のいずれか。
質問ダイアログボックス(リンクを使用する)	[キャンセル]。
選択ダイアログボックス	<ul style="list-style-type: none"> <li>モーダルダイアログボックス: [OK]/[キャンセル]または"実行する"/[キャンセル]</li> <li>モードレスダイアログボックス: ダイアログボックスとタイトルバーの[閉じる]ボタン</li> <li>作業ウィンドウ: タイトルバーの[閉じる]ボタン</li> </ul>
進行状況ダイアログボックス	環境を(副次的な影響を残さず)元の状態に戻す場合は、[キャンセル]を使用。それ以外の場合は、[停止]を使用。
情報ダイアログボックス	[閉じる]を使用。

- [適用]を除くすべてのコミットボタンで、ダイアログボックスウィンドウを閉じることができます。
- コミットボタンに対しては確認を行いません。不必要に確認すると、非常にわざわざしくなります。次の場合は例外です。
  - その操作によって大きな問題が発生する可能性がある。
  - その操作が明らかに他の操作と矛盾している。
  - 操作を間違った場合に、データ、時間、または労力に関する重大な損失をユーザーに与える可能性がある。
 その他のガイドラインと例については、「[確認](#)」を参照してください。
- コミットボタンは無効にしません。次の場合は例外です。
  - 変更するためにユーザーが昇格する必要がある場合は、ユーザーが変更するまで、肯定的なコミットボタンを無効にします。こうすると、ユーザーが実行できる操作は[キャンセル]のクリックのみになるため、ウィンドウを閉じるために昇格を実行することがなくなります。
  - その他の例外については、「[コントロールの無効化または削除とエラーメッセージの表示](#)」を参照してください。
- コミットボタンは、ダイアログボックスの下部に1列に並べ、右揃えにします。ただし、脚注エリアよりも上に配置します。コミットボタンが1つ([OK]など)の場合でも、このようにします。

間違った例:



この例では、[OK]ボタンが中央に配置されているため、不適切です。

- 次の順番でコミットボタンを配置します。
  - [OK]/"実行する"/[はい]
  - "実行しない"/[いいえ]
  - [キャンセル]
  - [適用](該当する場合)
  - [ヘルプ](該当する場合)
- 関連するコミットボタンを多数配置する場合は、[分割ボタン](#)を使用してまとめます。
- コミットボタン(ウィンドウが閉じるボタン)とその他のすべてのコマンドボタン([詳細設定]など)を明確に区別します。

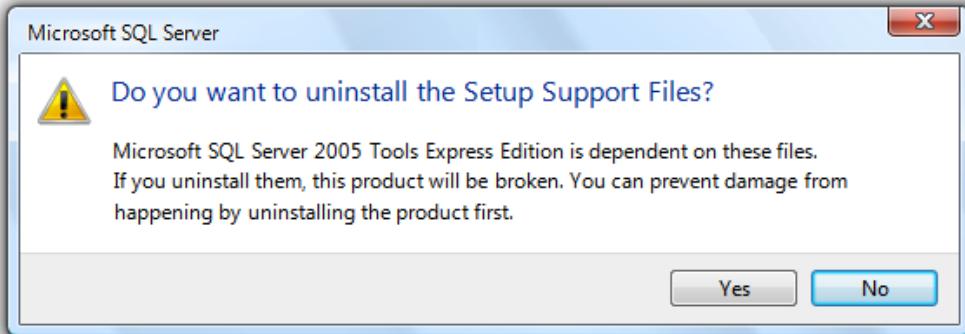
#### メイン指示テキストへの応答

- 肯定的なコミットボタンには、[OK]、[はい]/[いいえ]などの汎用的なラベルではなく、メイン指示テキストに対する具体的な応答を示したラベルを使用します。ユーザーがボタンのテキストを読むだけでオプションを理解できる必要があります。次の

場合は例外です。

- 情報ダイアログ ボックスなど、設定項目がないダイアログ ボックスには、[閉じる] を使用します。設定項目があるダイアログ ボックスには、[閉じる] を使用しないでください。
- [保存] や [選択] など、"具体的な応答" が汎用的な場合は、[OK] を使用します。具体的な設定/設定群を変更する場合は、[OK] を使用します。
- メイン指示テキストがない従来のダイアログ ボックスの場合は、[OK] などの汎用的なラベルを使用します。多くの場合、このようなダイアログ ボックスの目的は具体的なタスクを実行することではないため、具体的な応答は使用しません。
- 一部のタスクでは、情報に基づいた判断を行うために、ユーザーによく考えさせ、注意して読ませる必要があります。一般的に、[確認](#)を行うケースがこれに該当します。そのような場合は、汎用的なコミット ボタン ラベルを意図的に使用して、ユーザーにメイン指示テキストを読むように促し、軽率な判断をさせないようにします。

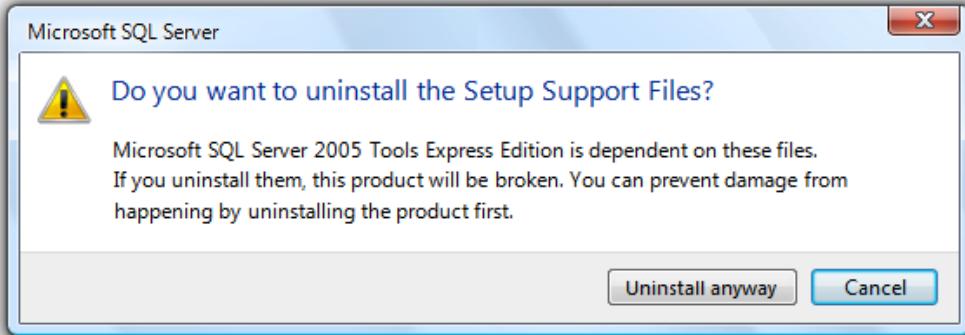
正しい例:



この例では、[はい] と [いいえ] のコミット ボタンによって、ユーザーに少なくともメイン指示テキストを読ませるようにしています。

- また、"anyway (このまま)" いう語を肯定的なコミット ボタン ラベルに追加して、ダイアログ ボックスに続行を推奨しない理由が示されていることを表すこともできます。こうすることで、ユーザーは続行する前にダイアログ ボックスをよく読むようになります。

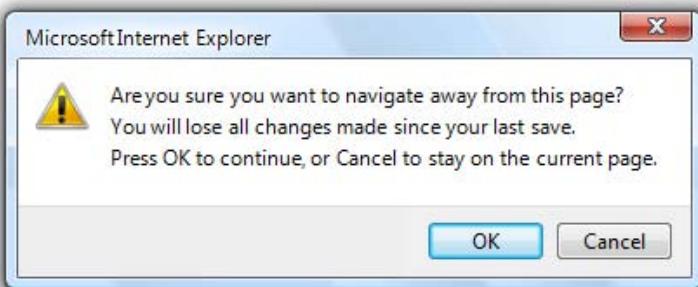
正しい例:



この例では、ユーザーが注意して続行する必要があることを示すために、"anyway (このまま)" という語がコミット ボタン ラベルに追加されています。

- 否定的なコミット ボタンには、メイン指示テキストに対する具体的な応答の代わりに、[キャンセル] または [閉じる] を使用します。ユーザーがダイアログ ボックスを見て、タスクを実行する必要がないことに気付くことはかなり頻繁にあります。[キャンセル] または [閉じる] の代わりに具体的な応答を使用すると、キャンセルする方法を判断するために、すべてのコミット ボタンをよく読む必要があります。[キャンセル] または [閉じる] というラベルを一貫して使用していると、簡単に見つけることができます。次の場合は例外です。
  - [はい]/[キャンセル] は使用しません。[はい]/[いいえ] をペアとして常に使用します。
  - [キャンセル] ではあいまいな場合は、具体的な応答を使用します。詳細については、「間接ダイアログ ボックスのコミット ボタン」を参照してください。
- コンテンツ エリアのテキストを使用して、汎用的なラベルに具体的な意味を持たせないようにします。代わりに、具体的なコミット ボタン ラベルを使用するか、ラベルが長くなる場合は[リンクを使用する質問ダイアログ ボックス](#)を使用します。

間違った例:

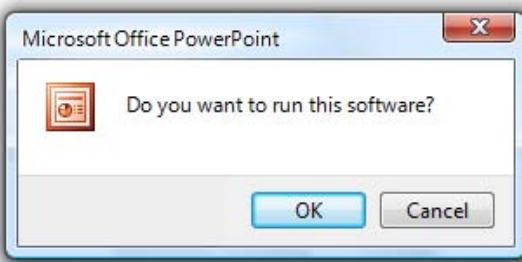


この例では、[OK] に "続行"、[キャンセル] に "このページを表示したままにする" という操作をそれぞれ割り当てています。

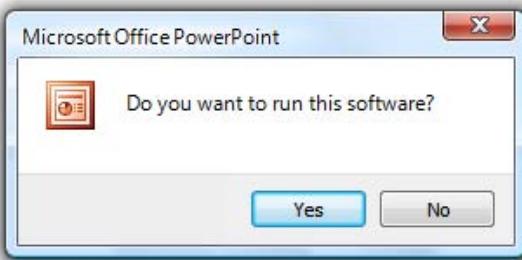
#### [はい]/[いいえ] ボタン

- できる限り、[はい]/[いいえ] ボタンよりも具体的な応答を使用します。[はい]/[いいえ] を使用しても問題はありませんが、具体的な応答を使用すると、即座に理解し、効率的に判断できます。ただし、確認の場合は、通常、[はい]/[いいえ] ボタンを配置して、応答する前に検討するようにユーザーに促します。
- "はい" か "いいえ" で答える質問に応答するだけの場合は、[はい]/[いいえ] ボタンを使用します。その場合のメイン指示テキストは、当然、"はい" か "いいえ" で答えられる質問にする必要があります。"はい" か "いいえ" で答える質問には、[OK] および [キャンセル] を使用しません。

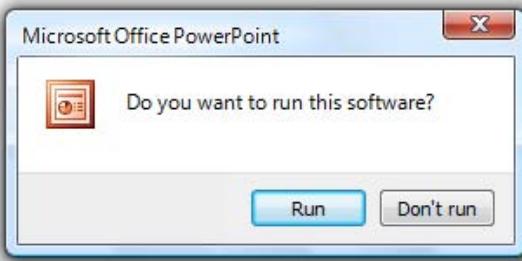
間違った例:



正しい例:



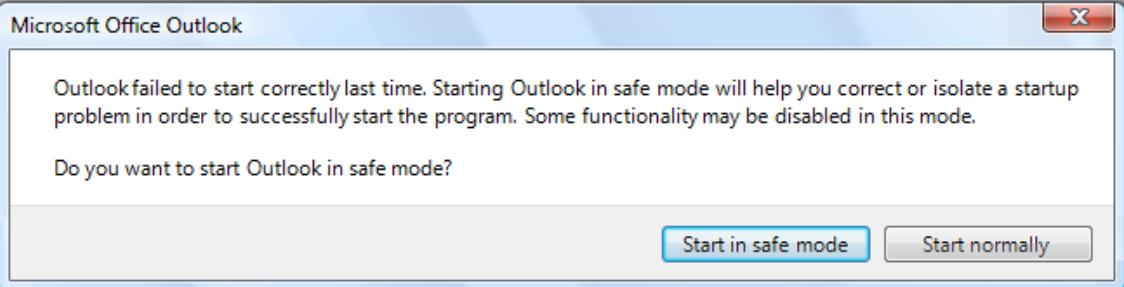
より良い例:



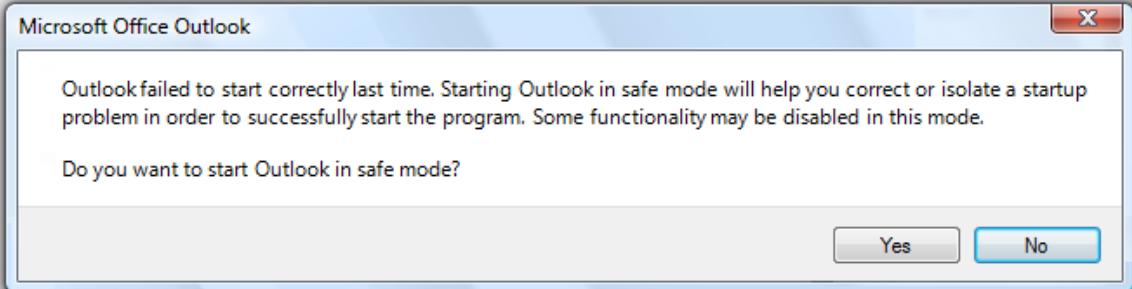
これらの例は、"はい/いいえ" で答える質問なので、[はい]/[いいえ] で十分ですが、具体的な応答の方が適しています。

- コミットボタンに具体的な語句を使用すると長くなる場合や不自然になる場合は、メイン指示テキストを "はい" か "いいえ" で答えられる記述にします。また、メイン指示テキストに対する応答が長い場合(5語あるいは15文字以上)は、コマンドリンクを使用することもできます。

間違った例:



正しい例:



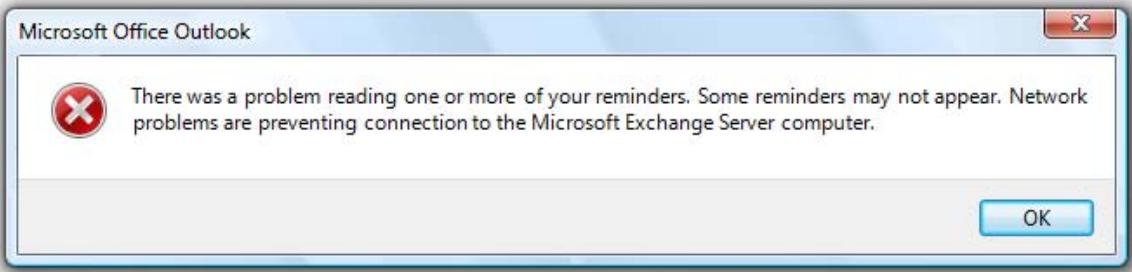
間違った例の具体的な応答は長すぎるため、正しい例では [はい]/[いいえ] を使用しています。

- "いいえ" という応答の意味が不明確な場合は、[はい]/[いいえ] ボタンを使用しないようにします。該当する場合は、代わりに具体的な応答を使用します。

#### [OK] ボタン

- モーダルダイアログボックスの場合、[OK] をクリックすることは、値を適用すること、タスクを実行すること、およびウィンドウを閉じることを意味します。
- 質問への応答には、[OK] ボタンを使用しません。
- [OK] ボタンでは、Enter キーが既定のアクセスキーになっているため、アクセスキーを割り当てません。これらのボタンにアクセスキーを割り当てないことで、他のアクセスキーの割り当てが容易になります。
- [OK] ボタンにラベルを正しく付けます。[OK] ボタンには、"Ok" や "Okay" ではなく、"OK" というラベルを付けます。
- エラーや警告には [OK] ボタンを使用しません。問題に対して、"OK" と応答することはできません。代わりに、[閉じる] を使用します。

間違った例:



この例では、[OK] の代わりに [閉じる] を使用する必要があります。

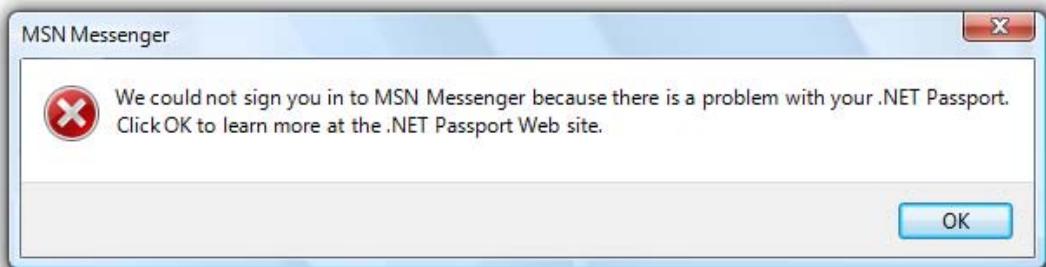
- モードレスダイアログボックスには、[OK] ボタンを使用しません。代わりに、タスクに固有のコミットボタン(たとえば、[検索])を使用します。ただし、[閉じる] ボタンのみが必要なモードレスダイアログボックスもあります。

#### [キャンセル] ボタン

- [キャンセル] をクリックすることは、すべての変更を破棄すること、タスクをキャンセルすること、ウィンドウを閉じること、および環境を(副次的な影響を残さず)元の状態に戻すことを意味します。入れ子になっている選択ダイアログボックスの場合、親選択ダイアログボックスで[キャンセル]をクリックすると、子選択ダイアログボックスで行った変更もすべて破棄されます。
- ユーザーが明示的に変更を破棄できるように[キャンセル] ボタンを用意します。ダイアログボックスには、明確な出口が必要です。ユーザーがタイトルバーの[閉じる] ボタンを見つけることを期待しないでください。
  - 例外: 設定項目のないダイアログボックスには、[キャンセル] ボタンを使用しません。この場合、[OK] ボタンおよび[閉じる] ボタンのみが表示されます。

じる] ボタンをクリックすると、[キャンセル] と同じ結果になります。

間違った例:



この例では、[閉じる] ボタンがタイトルバーにしかないため、ユーザーには選択肢がないように見えます。

- 質問への応答には [キャンセル] ボタンを使用しません。

間違った例:



この例では、"はい" か "いいえ" で答える質問に対する応答に、[OK] および [キャンセル] が使用されているため、不適切です。

- [キャンセル] ボタンでは、Esc キーが既定のアクセスキーになっているため、アクセスキーを割り当てません。これらのボタンにアクセスキーを割り当てないことで、他のアクセスキーの割り当てが容易になります。
- モードレスダイアログボックスには、[キャンセル] ボタンを使用しません。代わりに、[閉じる] を使用します。
- [キャンセル] ボタンは無効にしません。常にユーザーがダイアログボックスをキャンセルできるようにする必要があります。
  - 例外: 進行状況ダイアログボックスで処理をキャンセルできないときがある場合は、[キャンセル] ボタンを無効にします。ただし、そうした処理をいつでもキャンセルできるように設計する方がより優れたソリューションです。

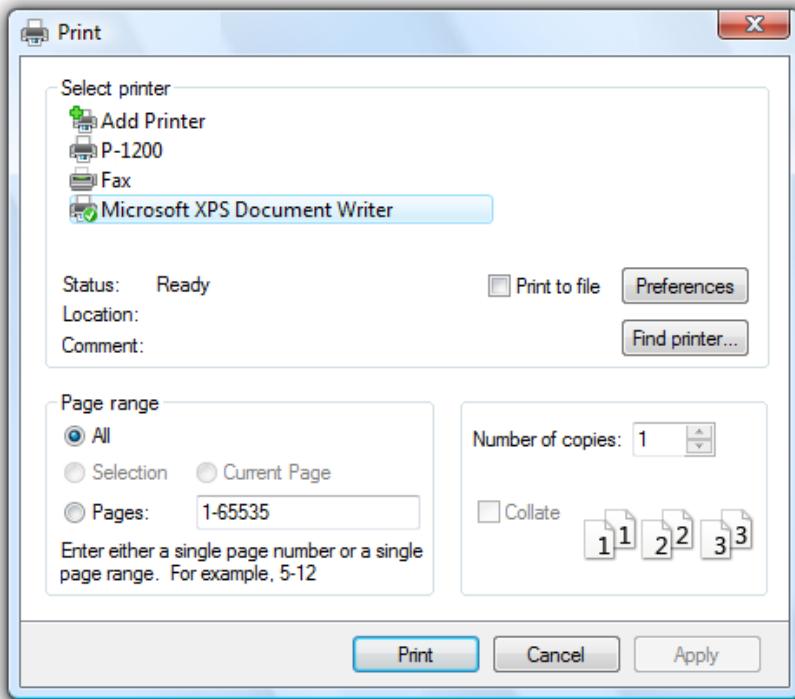
#### [閉じる] ボタン

- モードレスダイアログボックスおよびキャンセルできないモーダルダイアログボックスには、[閉じる] ボタンを使用します。
- [閉じる] をクリックすることは、既存の副次的な影響を残したまま、ダイアログボックスを閉じることを意味します。[完了] は必須の構成ではないため、使用しません。入れ子になっている選択ダイアログボックスの場合、親選択ダイアログボックスで [閉じる] をクリックすると、子選択ダイアログボックスで行った変更がすべて保持されます。
- ダイアログボックス内に明示的な [閉じる] ボタンを配置します。ダイアログボックスには、明確な出口が必要です。ユーザーがタイトルバーの [閉じる] ボタンを見つけることを期待しないでください。
- タイトルバーの [閉じる] ボタンには [キャンセル]/[閉じる] ボタンと同じ効果があることを確認してください。
- [閉じる] ボタンでは、Esc キーが既定のアクセスキーになっているため、アクセスキーを割り当てません。これらのボタンにアクセスキーを割り当てないことで、他のアクセスキーの割り当てが容易になります。

#### [適用] ボタン

- プロパティシートやコントロールパネル以外のダイアログボックスには、[適用] ボタンを使用しません。[適用] ボタンをクリックすると、保留中の変更が適用されますが、ウィンドウは開いたままになります。こうすることで、ユーザーがウィンドウを閉じる前に変更内容を評価できるようになります。ただし、[適用] ボタンが必要なのは、プロパティシートとコントロールパネルだけです。

間違った例:



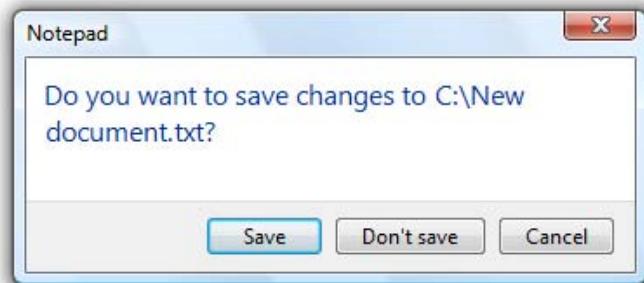
この例では、不必要的 [適用] ボタンが選択ダイアログ ボックスに使用されています。

#### 間接ダイアログ ボックスのコミット ボタン

注: 間接ダイアログ ボックスは、タスクの間接的な結果、またはシステムやバックグラウンド プロセスで問題が発生した結果として、コンテキストとは直接関係なく表示されます。間接ダイアログ ボックスの場合、キャンセルの意味がダイアログ ボックスのキャンセルとも、タスク全体のキャンセルとも考えられるため、[キャンセル] ボタンの役割が不明確です。

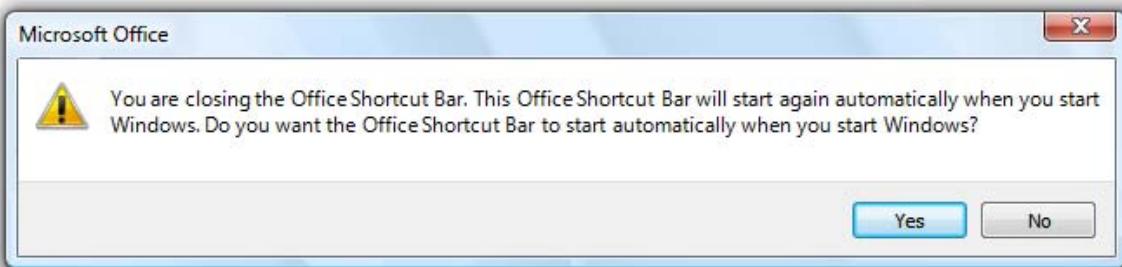
- ユーザーがダイアログ ボックスとタスク全体の両方をキャンセルする必要がある場合は、両方を実行するためのコミット ボタンを用意します。ダイアログ ボックスをキャンセルするボタンには、メイン指示テキストに対する否定的な応答を使用したラベルを付けます。タスク全体をキャンセルするボタンには、"キャンセル" というラベルを付けます。"キャンセル" を使用すると、そのダイアログ ボックスを多くのコンテキストで使用できます。

正しい例:



この例は、メモ帳でファイルを保存していないときに [新規] または [メモ帳の終了] をクリックすると表示されるダイアログ ボックスを示しています。[いいえ] をクリックすると、保存せずにダイアログ ボックスが閉じます。[キャンセル] をクリックすると、[新規] コマンドまたは [メモ帳の終了] コマンドがキャンセルされます。

間違った例:



この例では、このダイアログ ボックスが表示されることになったタスク (Office ショートカット バーを閉じる) をキャンセルする方法がありません。このダイアログ ボックスには、[キャンセル] ボタンが必要です。

- ユーザーがタスクではなくダイアログ ボックスをキャンセルする必要がある場合は、メイン指示テキストに対する具体的で否定的な応答をラベルに付けたボタンを使用します。[キャンセル] ボタンは使用しません。

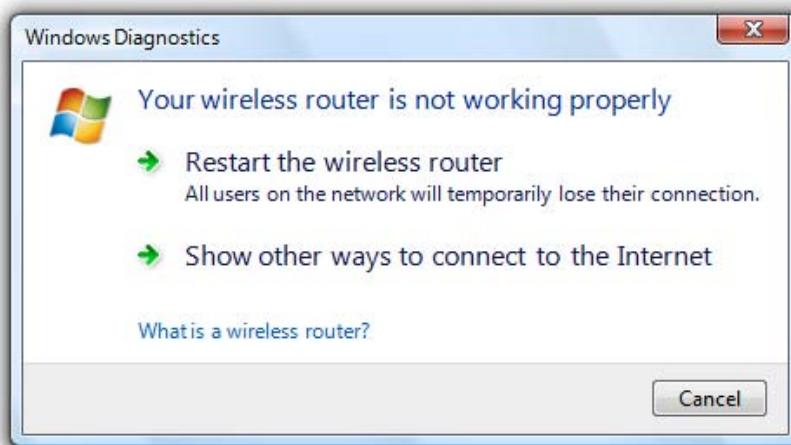


この例は、ActiveX コントロールをインストールする Web ページに移動したことで間接的に表示されるダイアログ ボックスを示しています。ここで [キャンセル] を使用するとあいまいになるため、代わりに [実行しない] を使用しています。

詳細と例については、「[コマンド ボタン](#)」を参照してください。

#### コマンド リンク

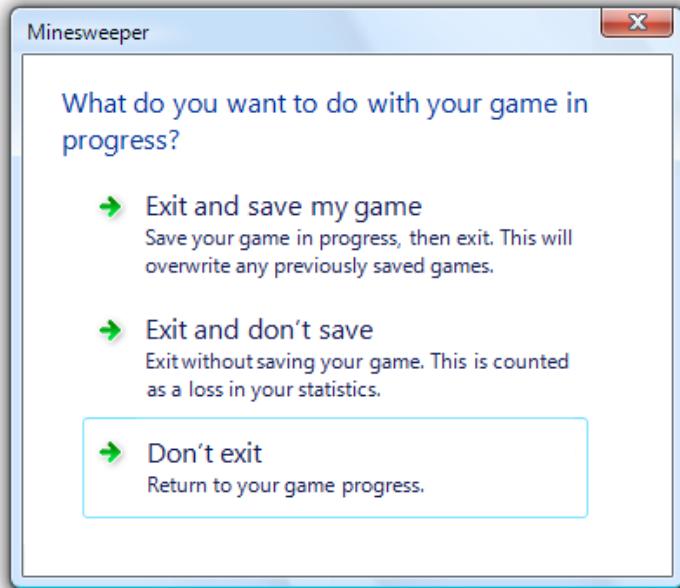
- 長いコマンドを提示する場合は、コマンド ボタンやラジオ ボタンと [OK] ボタンの組み合わせの代わりに、コマンド リンクを使用します。こうすると、ユーザーは1回のクリックで応答できます。ただし、この方法は質問が1つの場合にのみ使用できます。
- 最も一般的に使われるコマンド リンクを先頭に置きます。使用される頻度順に主に従って並べますが、論理的にも自然になるようにします。
  - 例外: "すべて実行する" 旨のコマンド リンクは先頭に置きます。
- コマンド リンクにさらに詳しい説明が必要な場合は、補足説明を記述します。補足説明には、そのコマンドをどのような場合に選択する必要があるか、選択すると何が起こるかを記載します。
- コマンド リンクの表現の言い換えになるような補足説明は使用しません。補足説明は、コマンド リンクだけではわかりにくい場合に限って使います。あるコマンド リンクに補足説明を付けたからといって、他のすべてのコマンド リンクに対しても補足説明が必要になるわけではありません。



この例では、1つのオプションの影響が補足説明で示されています。

- 語句は動詞から始め、末尾に句読点は付けません（英語の場合）。
- 強く推奨されるコマンドに対しては、ラベルに "(推奨)" を追加することを検討します。補足説明に対してではなく、リンクラベルに追加します。
- 詳しい知識のあるユーザー向けのコマンドに対しては、ラベルに "(詳細設定)" を追加することを検討します。補足説明に対してではなく、リンクラベルに追加します。
- 明示的な [キャンセル] ボタンを常に用意します。この用途でコマンドリンクを使用しないでください。

間違った例：



この例のダイアログ ボックスでは、[キャンセル] ボタンの代わりにコマンド リンクを使用しています。

詳細と例については、「[コマンドリンク](#)」を参照してください。

今後、この <アイテム> を表示しない

- [今後、この<アイテム>を表示しない] というオプションを使って、同じダイアログ ボックスが繰り返し表示されないようにユーザーが設定できるようにすることを検討します。ただし、これは他に良い手段がない場合に限ります。ユーザーが本当に必要としている場合は、ダイアログ ボックスを常に表示することをお勧めします。必要としていない場合は、単純に削除します。
- <アイテム>を具体的なアイテムに置き換えます。たとえば、[今後、この通知を表示しない] とします。ダイアログ ボックスを指す場合は、一般的に、[今後、このメッセージを表示しない] を使用します。
- ユーザー入力を今後の既定値として使用する場合は、その旨を明確に示します。オプションの下に、"選択内容は、今後既定値として使用されます" という文を追加します。
- このオプションは既定でオンにしません。ダイアログ ボックスを本当に1回だけ表示する必要がある場合は、確認せずに1回だけ表示します。このオプションをユーザーの手をわざらわせることの言い訳に使用しないでください。既定の動作がわざらしいものでないことを確認してください。

間違った例：



この例は、1回だけ表示する必要があるメッセージを示しています。確認する必要はありません。

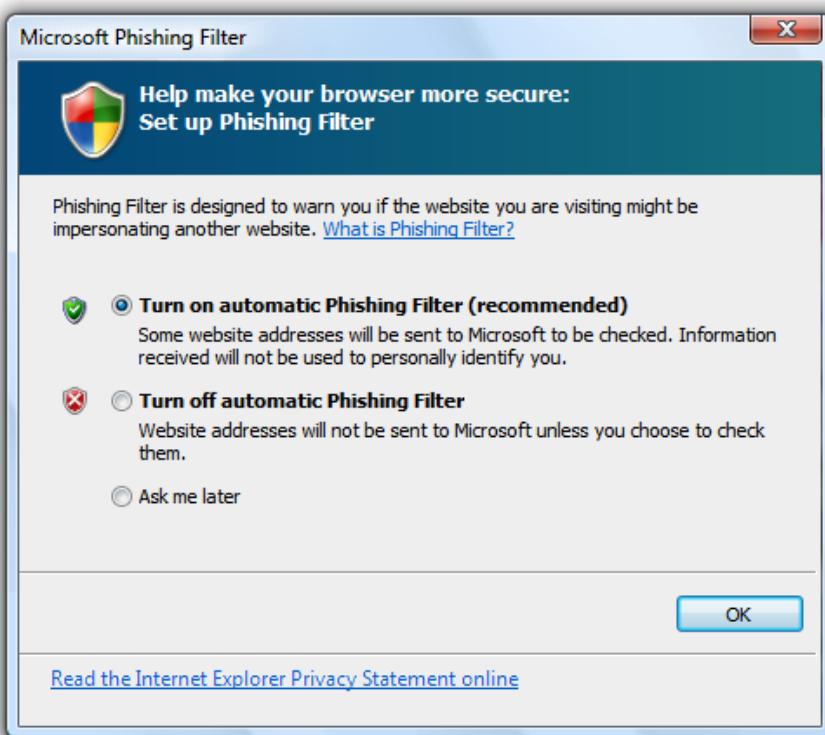
- 設定は、ユーザーごとに維持されるようにします。
- ユーザーがこのオプションをオンにして [キャンセル] をクリックした場合、このオプションは "有効" になります。この設定はメタオプションであるため、副次的な影響を残さないという [キャンセル] の標準の動作には従いません。ユーザーが今後そのダイアログ ボックスを表示しないようにする場合、そのダイアログ ボックス自体もキャンセルする可能性が高いことに注意してください。
- ユーザーがこれらのダイアログ ボックスを元に戻す必要がある場合は、プログラムの [オプション] ダイアログ ボックスに "メッセージの復元" コマンドを追加します。

詳細については、デザインコンセプトの「[今後、この<アイテム>を表示しない] オプションの使用」を参照してください。

#### [後で確認する]

- 次の場合にのみ、ダイアログ ボックスを閉じるために、このオプションを用意します。
  - ダイアログ ボックスが間接ダイアログ ボックスである。ユーザーは別のタスクに専念している可能性があります。
  - ユーザーは応答する必要があるが、即座に行う必要はない。そのため、ユーザーは作業を続行できます。
  - 十分な検討や労力が必要な質問である。この場合、時間をかけければ、ユーザーはより適切な判断を下すことができます。
  - ダイアログ ボックスまたはオプションが後で自動的に表示される。実際にユーザーは後で確認を求められます。

間違った例:



この例では、質問が単純なため、[後で確認する] オプションを追加することで、かえって複雑になっています。

- この場合は、ユーザーがその場で応答することを期待しつつも、[キャンセル] または [閉じる] のどちらかで普通にダイアログ ボックスを閉じることができます。このオプションを適切に使用できる場面はほとんどありません。

#### 詳細表示/簡易表示

- 対象ユーザーが通常は必要としない高度なオプションおよびコマンド、使用頻度の低いオプションおよびコマンド、または詳細情報を表示/非表示にするには、詳細表示/簡易表示を切り替える段階的表示ボタンを使用します。標準的な用途では、このようにしてダイアログ ボックスを簡素化します。一般的に使用されるオプション、コマンド、または情報を非表示にしないでください。ユーザーが見つけられない可能性があります。

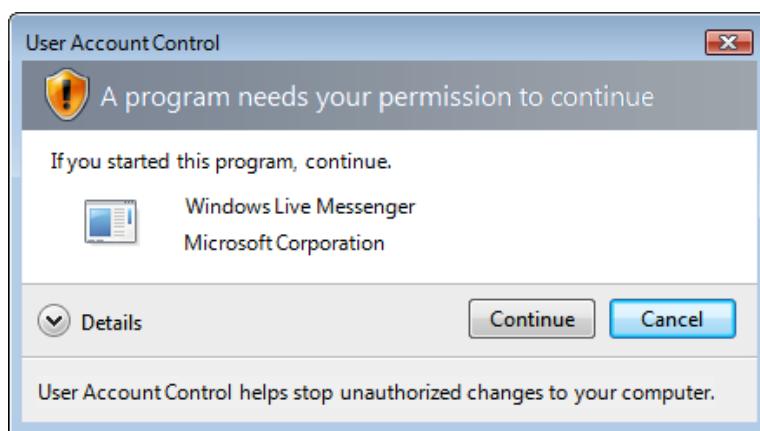


この例では、既定で、使用頻度の低いオプションを非表示にしています。

- 詳細を表示する必要が本当にある場合を除いて、詳細/簡易コントロールは使用しません。同じ情報を別の言葉で言い換えただけのものにならないように注意してください。
- ヘルプを表示する目的で詳細/簡易コントロールを使用しないでください。そのような場合は、ヘルプリンクまたは脚注を使用してください。
- タスク ダイアログ ボックスでは、詳細/簡易コントロールと [今後、この<アイテム>を表示しない] を共に使用しません。共に使用すると、外観が不自然になります。
- ラベルのガイドラインについては、「[段階的表示](#)」を参照してください。

#### 脚注

- 脚注は、ダイアログ ボックスの目的に不可欠ではないものの、ユーザーが判断を下す際に役立つ情報を表示するために使用します。ダイアログ ボックスに応答するときに、多くのユーザーが脚注を読まなくても、情報に基づいた判断を行うことができるようになります。



この例では、脚注の情報は補足説明であり、不可欠ではありません。

#### コントロールの無効化または削除とエラー メッセージの表示

- コントロールが現在のコンテキストで適用されない場合は、次の方法を検討します。
  - ユーザーが有効にできないコントロール、またはユーザーが使用することが想定されず、状態が頻繁に変化しないコントロールは削除します。不要なコントロールを削除することで、簡潔なダイアログ ボックスになり、見落としがなくなります。コントロールの表示/非表示を頻繁に切り替えると、わざわざしくなります。
  - ユーザーが使用することが想定され、状態が頻繁に変化するコントロールや、無効になっている理由をユーザーが簡単に推測できるコントロールは無効にします。簡単に推測できる例としては、何らかの入力を求める空のテキスト ボックスが 1 つあるときにコミット ボタンが無効になっている場合が挙げられます。テキスト ボックスや編集可能なドロップダウン リストに関する重大ではないユーザー入力の問題を表示する場合は、[バルーン](#)を使用します。ただし、問題がバルーンで説明できない場合や、複数のコントロールに関連する場合は、簡単に推測できません。
- コントロールを有効にしたままで、使用方法を誤ったときにエラー メッセージを表示します。この場合、コントロールが無効になっていると、無効になっている理由をユーザーが理解することが難しくなります。問題を特定するために、ユーザーは試行錯誤と論理的な推測を行うことになります。問題を明確に説明する、有用なエラー メッセージを表示するようにします。
- ヒント: コントロールを無効にするか、またはエラー メッセージを表示するかで迷った場合は、まず実際にエラー メッセージを作成してみてください。対象ユーザーが簡単に思い付かないような役立つ情報がエラー メッセージに含まれている場合は、コントロールを有効にしたままで、エラーを表示するようにします。その他の場合は、コントロールを無効にします。
- コントロールを無効にした場合は、関連するすべてのコントロールも無効にします。たとえば、ラベル、補足説明、コマンド ボタンがあります。ただし、[グループ ボックス](#)がある場合、グループ ボックスは無効にしません。

## 必須の入力

- ユーザーがコントロールに情報を入力する必要があることを示す場合は、次の方法を検討します。
  - 何も表示しないが、必須の入力を見落としている場合に、エラーメッセージを表示する。この方法は、ほとんどの入力が任意の場合やコントロールを見落とす可能性が低い場合に使用すると、煩雑さがなくなり効果的です。また、エラーメッセージの数を少なくすることができます。
  - ラベルの先頭にアスタリスク (\*) を使用して、必須の入力であることを示す。次のどちらかの方法で、アスタリスクについて説明します。
    - コンテンツエリアの下部の脚注に、"必須の入力" と表示する。
    - アスタリスクのツールヒントで "必須の入力" と表示する。

この方法は、必須のコントロールが多くない場合は効果的ですが、ほとんどのコントロールが必須の場合は効果的ではありません。

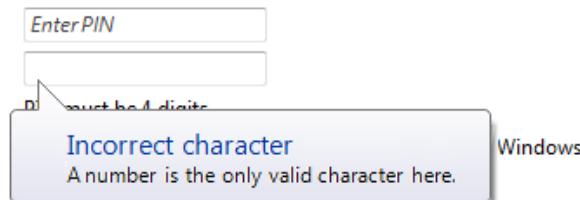
Sales price: \$0.00  
\* Income account:  
\* Item tax: taxable

この例では、必須の入力であることを示すために、アスタリスク (\*) を使用しています。

- すべてのコントロールが必須の入力である場合、コンテンツエリア上部の適切な場所に、"すべて入力する必要があります" と表示する。この方法は、こうした特定の場合に使用すると、煩雑さを減らすことができます。
- 任意の入力には、ラベルの後ろに"(任意)" を追加する。この方法は、ほとんどの入力が必須の場合は効果的ですが、それ以外の場合は効果的ではありません。
- 一貫性を保つために、できる限りプログラム全体で同じ方法を使用して必須の入力であることを示します。具体的には、必須の入力と任意の入力のどちらかを必要に応じて示し、同じプログラム内で両方を使用しないようにします。

## エラー処理

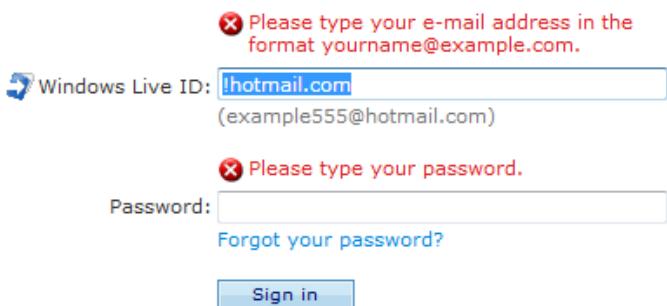
- 有効なユーザー入力しか受け付けないコントロールを使用することで、エラーを防止します。また、妥当な既定値を表示することで、エラーの数を減らすことができます。
- できる限り早い段階でユーザー入力を検証し、可能であれば入力した時点でエラーを表示します。
- ユーザー入力の問題には、モードレスのエラー処理(インプレースエラーまたはバルーン)を使用します。
  - テキストボックスにフォーカスがあるとき、または、テキストボックスからフォーカスが移動した直後に、ユーザー入力の問題が検出された場合、問題がそれほど重大ではなく一過性のものであれば、**バルーン**を使用します。インプレースメッセージの表示に必要な空き画面領域または動的レイアウトは、バルーンには必要ありません。一度に1つのバルーンだけを表示します。問題がそれほど重大なものではないので、エラーアイコンは不要です。バルーンは、クリックされるか、問題が解決されるか、一定の時間が経過すると消えます。



この例では、コントロールにフォーカスがある状態で、入力に問題があることをバルーンを使って通知しています。

- 操作から時間が経過した後でエラーが検出される場合は、**インプレースエラー**を使用します。コミットボタンをクリックした時点で見つかるエラーなどに使用します(即座に反映される設定にはインプレースエラーを使用しないでください)。一度に複数のインプレースエラーを表示できます。標準のテキストおよび 16 × 16 ピクセルのエラーアイコンを使用します。ユーザーが操作を確定し、他のエラーがないと判断されるまで、インプレースエラーは消えません。

## Sign in



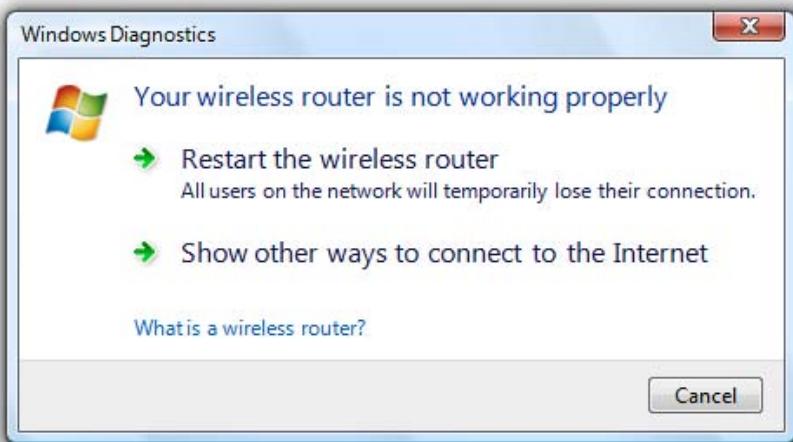
この例では、コミットボタンをクリックして初めてエラーが検出されるので、インプレースエラーが使用されています。

- 上記以外の問題には、モーダルなエラー処理(タスクダイアログボックスまたはメッセージボックス)を使用します。複数のコントロールが関連するエラー、その場で指摘する必要がないエラー、コミットボタンがクリックされた時点で検出される入力以外のエラーなどが、これに該当します。
- 入力の問題が検出され、報告された場合は、間違ったデータが入力されている最初のコントロールに入力フォーカスを設定します。必要に応じてコントロールをスクロールして見える状態にしてください。

詳細と例については、「[エラーメッセージ](#)」および「[バルーン](#)」を参照してください。

## ヘルプ

- ユーザーインスタンスを提供する場合は、次の方法を検討してください(望ましい順に列挙してあります)。
  - 対話型コントロールに内容を表すラベルを付けます。ユーザーは、他のどのテキストよりも、対話型コントロール上のラベルを読む可能性が高い傾向にあります。
  - スタティックテキストラベルを使用し、コンテキストに即した説明を追加します。
  - 対応するヘルプトピックへの具体的なヘルプリンクを提供します。
- ダイアログボックスのコンテンツエリアの下部にヘルプリンクを配置します。ダイアログボックスに脚注があり、ヘルプリンクがそれに関連している場合は、ヘルプリンクを脚注内に配置します。



この例では、ヘルプリンクはダイアログボックス全体に関連しています。

- 例外: ヘルプトピックを持つ設定グループがダイアログボックスに複数ある(通常はグループボックス内にある)場合は、ヘルプリンクをグループの下部に配置します。
- 一般的なヘルプトピックリンク、あいまいなヘルプトピックリンク、または汎用的なヘルプボタンは使用しません。ユーザーは、汎用的なヘルプを無視する傾向にあります。

詳細と例については、「[ヘルプ](#)」を参照してください。

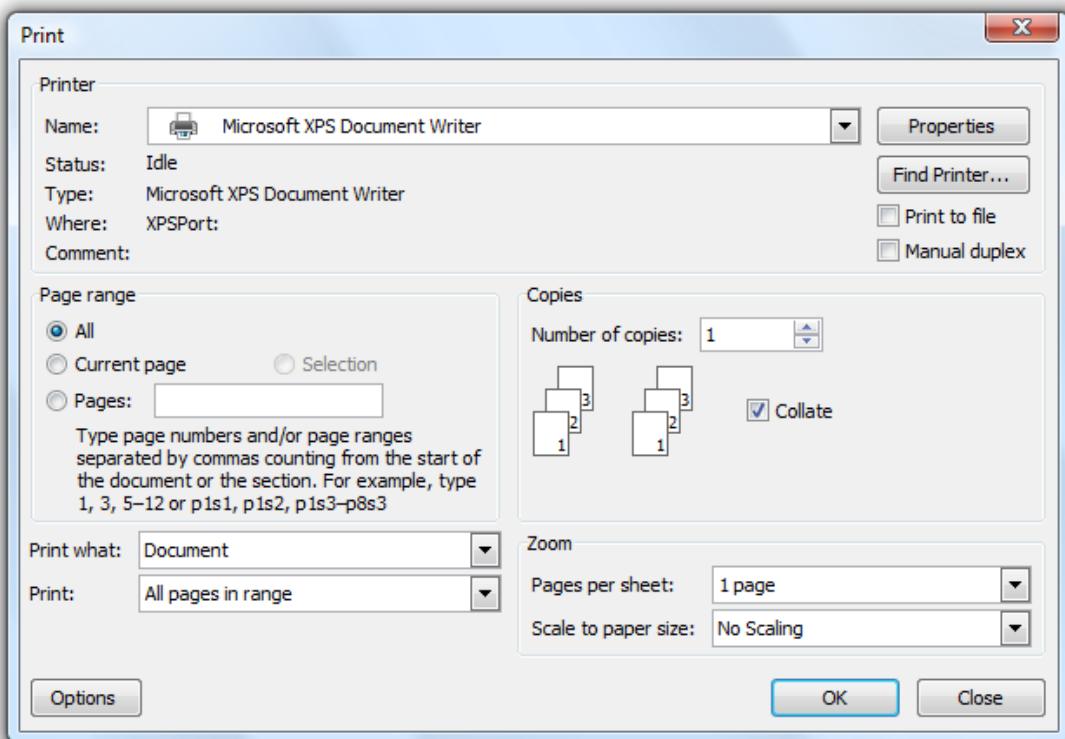
## 既定値

- ダイアログボックスには必ず1つ、既定のコミットボタンを置きます。
- 質問ダイアログボックスの場合:
  - 既定値には、最も安全(データの消失やシステムアクセスが失われることを防ぐため)で、最もセキュリティの高い応答を選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用される応答または最も便利な応答を選択します。
  - 例外: リスクのある応答は既定に設定しません。ただし、そのコマンドの処理を容易に元に戻す方法がある場合を

除きます。

- 選択ダイアログ ボックスの場合:

- 最初の既定値には、最も安全(データの消失やシステムアクセスが失われることを防ぐため)で、最もセキュリティの高い値を各コントロールに対して選択します。安全性およびセキュリティを判断要素として考える必要がない場合は、最もよく使用されるオプションや最も便利なオプションを選択します。
- 2回目以降の既定値には、そのオプションで前回選択された値が繰り返し使用されることが多く、またそうすることが安全な場合は、そのオプションを再度選択します。それ以外の場合は、最初の既定値を選択します。



この例は、ユーザーが前回と同じ選択をする可能性が高い印刷設定です。ただし、印刷部数は変更される可能性があるため、この設定は再度選択しません。

## 推奨されるサイズと間隔

- Windows の最小画面解像度 800 × 600 ピクセルをサポートします。レイアウトは、1024 × 768 ピクセルの画面解像度を使用したサイズ変更可能なウィンドウに合わせて最適化します。
- サイズ変更可能なウィンドウを使用するとスクロールバーの使用やデータの切り詰めを避けることができる場合は、常にサイズ変更可能なウィンドウを使用します。コンテンツや一覧が動的に変化するウィンドウでは、サイズ変更ウィンドウを有効活用できます。
- 固定サイズのウィンドウの場合、全体が表示され、[作業領域](#)に収まるサイズにする必要があります。
- サイズ変更可能なウィンドウは高解像度に最適化できますが、表示する際の実際の画面解像度に応じてサイズを縮小します。
- コンテンツに適した既定のウィンドウ サイズを選択します。画面を効率的に使用できる場合は、ウィンドウの初期サイズを大きくします。

## テキスト

### 全般

- 冗長なテキストは削除します。タイトル、メイン指示テキスト、補足指示テキスト、コンテンツエリア、コマンドリンク、コミット ボタンなどに冗長なテキストがないか確認してください。通常、メイン指示テキストと対話型コントロールのテキストはそのまま残し、他の部分にある冗長なテキストを削除します。
- 肯定文を使用します。肯定文の方が、ユーザーは理解しやすくなります。

#### 正しい例:

ファイルとプリンターの共有を有効にしますか?

#### 間違った例:

ファイルとプリンターの共有を無効にしますか?

ただし、表現は関連付けられているコマンドと一致させる必要があります。コマンドが否定的な表現であった場合、たとえば [無効にする] というコマンドの確認には、"無効にする" という表現を使用します。

- 必要な場合は、ダイアログ ボックスを示すために "ウィンドウ" という語を使用します。
- ユーザーに指示を伝える場合は、二人称 ("あなた/あなたの") を使用します(英語の場合)。これは、メイン指示テキストおよび

コンテンツ エリアで使用します。これ以外のほとんどの場合は、二人称を明示しません。

例:

Choose the pictures you want to print (印刷する画像を選択します)

Choose an account (アカウントを選択します)

- ユーザーがプログラムに指示を伝える場合は、一人称 ("私は/私に/私の") を使用します (英語の場合)。これは、メイン指示テキストに応答するコンテンツ エリアのコントロールで使用します。

例: Print the photos on my camera. (カメラの写真を印刷します。)

## ダイアログ ボックスのタイトル

- ダイアログ ボックスを呼び出したコマンド、機能、またはプログラムを識別できるタイトルを使用します。
  - ダイアログ ボックスをユーザーが開始した場合は、コマンド名または機能名を使用します。次の場合は例外です。
    - いくつもの異なるコマンドによってダイアログ ボックスが表示される場合は、プログラム名を明記するようにします。
    - メイン指示テキストと内容が重複している場合は、代わりにプログラム名を使用します。
  - システムまたはプログラムによって開始される (そのため、コンテキストに即していない) 場合は、プログラム名または機能名を使用してコンテキストを示します。
  - タイトルで、ダイアログ ボックスで実行する操作を説明することはしません。説明はメイン指示テキストで行います。
- コマンドに基づいた名前ではなく、正確なコマンド名を使用します。ただし、省略記号がある場合、省略記号は含めません。必要に応じて、コマンドのメニュー タイトルを利用して、適切なタイトルを作成します。例: [挿入] メニューの [オブジェクト...] コマンドの場合は、"オブジェクトの挿入" というタイトルを使用します。
- モードレス ダイアログ ボックスでタスク バー上に表示される場合は、タスク バー上のタイトルの表示を最適化します。その際に、特徴をなす情報をタイトルの最初の方に簡潔にまとめます。例: "66% 完了"、"3 件あります" などを使用します。
- タイトルに "ダイアログ ボックス" や "進行状況" という語は含めません。これらの語を明示せず、省略することで、ユーザーの視認性が向上します。
- タイトルスタイルの大文字化**を使用し、末尾に句読点は付けません。

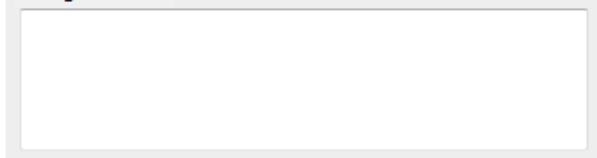
## メイン指示テキスト

- メイン指示テキストは、ダイアログ ボックスの作業内容を簡潔に説明するために使用します。指示テキストは、具体的な文、必須の指示、または質問である必要があります。優れた指示テキストとは、ダイアログ ボックスの操作方法のみに重点を置かず、ダイアログ ボックスの目的をユーザーに伝えるテキストです。
- 説明がなくてもわかる場合は、メイン指示テキストを省略します。その場合は、ダイアログ ボックスのコンテンツだけで処理内容が明確に言い表されています。たとえば、[ファイルを開く] や [ファイルの保存] というコモン ダイアログ ボックスの場合、そのコンテキストと設計によって目的が明確になっているため、メイン指示テキストは必要ありません。
- メイン指示テキストと同じ内容を繰り返すようなコントロール ラベルは省略します。この場合、メイン指示テキストにアクセスキーを割り当てます。

許容される例:

Enter your billing address

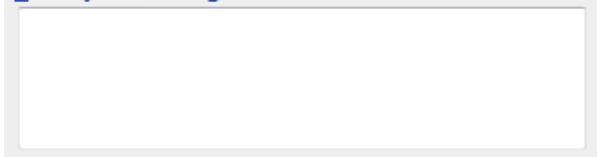
Billing address:



この例では、テキスト ボックスのラベルで、メイン指示テキストと同じ内容が繰り返されています。

より良い例:

Enter your billing address



この例では、冗長なラベルは削除され、メイン指示テキストにアクセスキーが割り当てられています。

- 簡潔な 1 文だけを使用します。メイン指示テキストには必須の情報だけを含めます。さらに詳しい情報を説明する必要がある場合は、補足指示テキストを使用してください。
- 可能な限り、具体的な動詞を使用します。ユーザーにとって、具体的な動詞 (接続する、保存する、インストールする、など) は、汎用的な動詞 (構成する、管理する、設定する、など) よりも有益です。

- ・センテンススタイルの大文字化を使用します。
- ・指示テキストが文の場合、文末の句点は含めません。指示テキストが疑問文の場合は、疑問符を付けます。
- ・進行状況ダイアログ ボックスの場合、進行中の処理を簡潔に説明する語句を使用し、末尾に省略記号を付けます。例: 画像を印刷しています... とします。
- ・ヒント: メイン指示テキストが適切かどうかを判断するには、友人に説明する場合を想像してみます。メイン指示テキストに対する応答が不自然であったり、不便であったり、使いにくかったりする場合は、メイン指示テキストを変更します。

#### 補足指示テキスト

- ・必要に応じて、補足指示テキストを使用して、ページの理解や使用に役立つ追加情報を表示します。詳細情報や用語の定義を提示できます。
- ・ダイアログ ボックスがシステムまたはプログラムによって開始される(そのため、コンテキストに即していない)場合は、補足指示テキストを使用して、ダイアログ ボックスが表示された理由を説明します。通常、このようなダイアログ ボックスのコンテキストは不明確です。
- ・メイン指示テキストの言い回しを少し変えて繰り返すことは避けます。追加する情報がない場合、補足指示テキストは省略します。
- ・文を使用し、末尾に句点を付けます。センテンススタイルの大文字化を使用します。

#### コマンド リンク

- ・コマンド リンクの処理の内容と他のコマンド リンクとの違いがはっきりわかる、簡潔なリンク テキストにします。これだけで内容が理解できる、メイン指示テキストに対応した表現である必要があります。リンクの本来の意味や、他のリンクとの違いを、ユーザーが考えこむようなものにはしないでください。
- ・コマンド リンクは常に動詞から始めます(英語の場合)。
- ・センテンススタイルの大文字化を使用します。
- ・末尾に句読点は付けません。
- ・必要に応じて、句点を付けた完全な文を使用して詳細に説明します。ただし、この説明は必要な場合にのみ追加します。1つのコマンド リンクに説明が必要な場合でも、他のすべてのコマンド リンクに説明を付ける必要はありません。

詳細と例については、[コマンド リンク](#)のガイドラインを参照してください。

#### コミット ボタン

- ・単独で意味をなし、メイン指示テキストに対応した応答になる、具体的なコミット ボタン ラベルを使用します。ラベル以外のものを読まなくともユーザーが意味を理解できることが理想です。一般にユーザーは、スタティックテキストよりも、コマンド ボタンのラベルを優先的に読む傾向があります。
- ・コミット ボタンのラベルは動詞から始めます(英語の場合)。"OK"、"はい"、"いいえ"は例外です。
- ・センテンススタイルの大文字化を使用します。
- ・末尾に句読点は付けません。
- ・一意な[アクセスキー](#)を割り当てます。
  - ・例外: [OK] ボタンと[キャンセル] ボタンでは、それぞれ Enter キーおよび Esc キーが既定のアクセスキーになっているため、アクセスキーを割り当てません。これらのボタンにアクセスキーを割り当てないことで、他のアクセスキーの割り当てが容易になります。

#### ドキュメント

ダイアログ ボックスに言及するときは、以下のこと留意します。

- ・プログラミングおよびその他の技術文書では、"ダイアログ ボックス"と記述します。それ以外のドキュメントでは、ダイアログ ボックスのタイトルを使用します。タイトルバーが非表示の場合は、メイン指示テキストを使用します。
- ・一般的に、ユーザー向けドキュメントでダイアログ ボックスに言及する必要がある場合は、"ウインドウ"を使用します。簡単な質問ダイアログ ボックスや確認は、"メッセージ"と表記することもできます。
- ・大文字と小文字の区別を含め、タイトル テキストまたはメイン指示テキストを正確に引用します。
- ・タイトルを半角の角かっこ([ ])で囲み、可能な場合は太字にします。

例: [Windows セキュリティ] の [その他のオプション] をクリックします。

# ダイアログ ボックスのデザイン コンセプト

[ダイアログ ボックス](#)

[ダイアログ ボックスの使用パターン](#)

ダイアログ ボックスを適切に使用すると、簡単にプログラムに機能を追加し、柔軟性を与えることができます。適切に使用しないと、ユーザーの手をわざらわせ、作業の流れを止め、プログラムを使用するのが面倒だという印象を与えることになります。モーダル ダイアログ ボックスは、ユーザーの注意を促します。ダイアログ ボックスは、他のユーザー インターフェイス (UI) より簡単に実装できる場合が多いため、使用しすぎる傾向があります。

## 効果的なダイアログ ボックスの設計

ダイアログ ボックスは、使用状況に合わせた設計特性にするのが最も効果的です。ダイアログ ボックスの設計の大部分は、目的 (オプションを提示する、質問する、情報またはフィードバックを提供する)、種類 (モーダルまたはモードレス)、およびユーザー操作 (必須の応答、任意の応答、または確認) によって決まります。使用状況の大部分は、コンテキスト (開始するのがユーザーか、プログラムか)、ユーザー操作の可能性、および表示頻度によって決まります。

### ダイアログ ボックスの特徴

モーダル ダイアログ ボックスには、次の特徴があります。

- ユーザーが現在作業しているのとは別のウィンドウに表示される。
- ユーザーの操作が必要である (オーナー ウィンドウでの作業を続行する前に、ダイアログ ボックスを閉じる必要があります)。
- ユーザーの作業を中断させる。
- [遅延型のコミット モデル](#)を使用する (明示的にコミットするまで変更されません)。
- タスクをコミットするためのコマンド ボタンを備えている。
- 作業を続行する前に完了させる必要がある、重要または頻度が低い、1回限りのタスクに最も適している。

モードレス ダイアログ ボックスには、次の特徴があります。

- 作業ウィンドウまたは独立したウィンドウを使用したコンテキストで表示できる。
- ユーザーの操作を必要としない (必要に応じて、ユーザーはダイアログ ボックスまたは作業ウィンドウと呼び出し元のウィンドウを切り替えることができます)。
- [即時型のコミット モデル](#)を使用する (即座に変更が反映されます)。
- ウィンドウを閉じるためのコマンド ボタンを備えている。
- 頻度が高い、繰り返し発生する、または継続的なタスクに最も適している。

### ダイアログ ボックスの対話操作

ダイアログ ボックスでのユーザー操作にも、さまざまな種類があります。

- 必須の応答。ユーザーは必須項目を入力するために応答する必要があります。たとえば、[置換] コマンドでは、ダイアログ ボックスを表示して、ユーザーが検索と置換を行うための文字列を指定できるようにする必要があります。
- 任意の応答。ユーザーは任意の項目を入力するために応答することができます。ただし、通常は既定値をそのまま使用します。たとえば、ユーザーは [印刷] ダイアログ ボックスのオプションを変更することができますが、通常は既定のオプションをそのまま使用します。
- 確認のみ。ユーザーが唯一できる操作は、確認するために、ダイアログ ボックスを読み、閉じることです。

### ダイアログ ボックスのコンテキスト

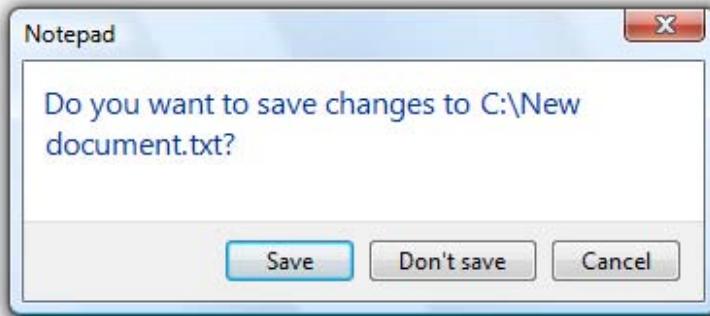
ダイアログ ボックスはさまざまなコンテキストで表示されます。

- ユーザーによる開始。ダイアログ ボックスが、ユーザーの操作によって直接的または間接的に表示されます。
- システムまたはプログラムによる開始。ダイアログ ボックスが、ユーザーの操作に関係なく表示されます。

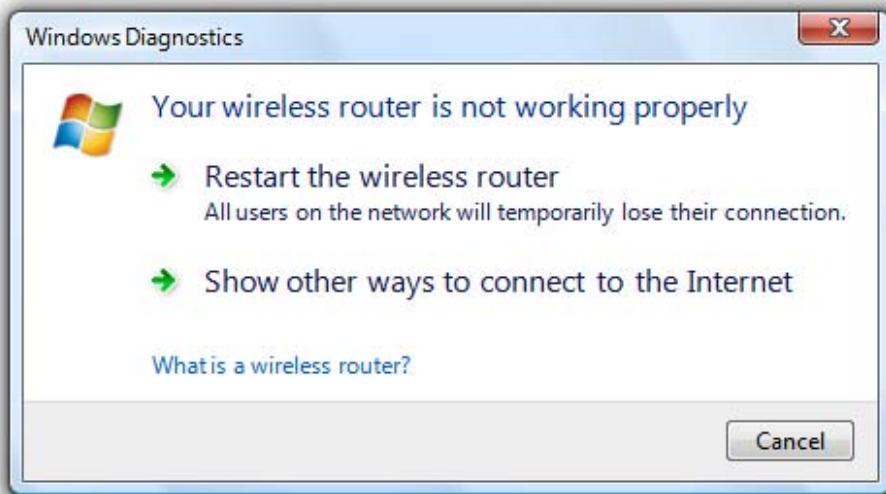
### 効果的なダイアログ ボックスの作成

効果的な例を紹介します。

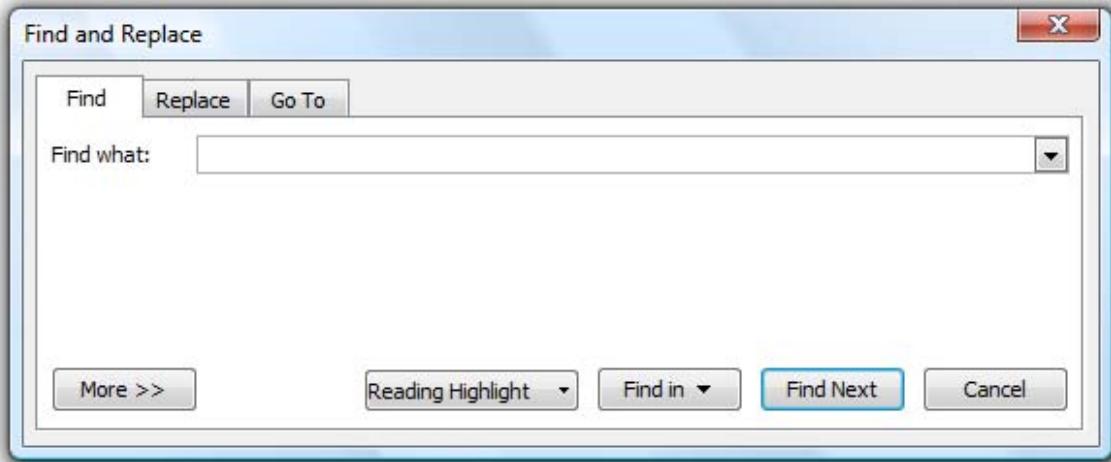
- モーダル ダイアログ ボックスは、ユーザーが開始する、応答の必要な一時的なタスクに最適です。



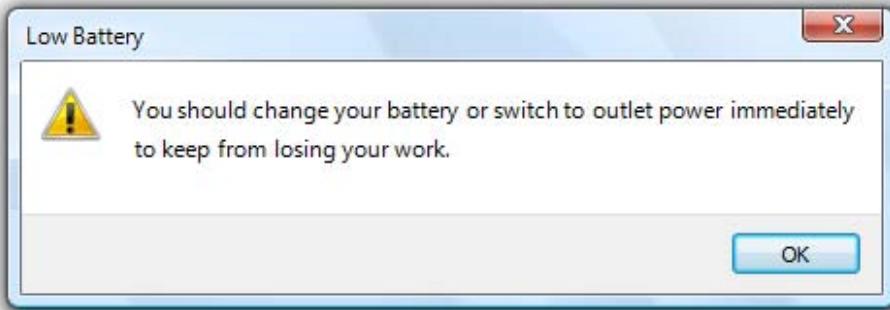
- モーダル ダイアログ ボックスは、システムまたはプログラムによって低い頻度で開始される、応答の必要な一時的なタスクに最適です。



- モードレス ダイアログ ボックスは、ユーザーが開始する、継続的なタスクに最適です。



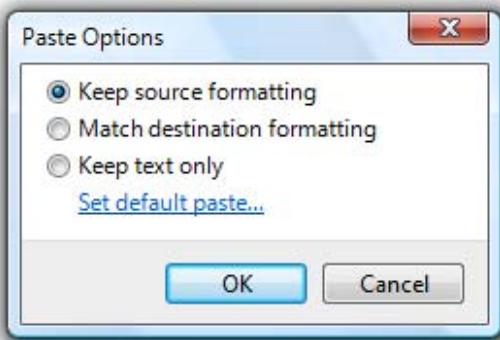
- モーダル ダイアログ ボックスは、重要なまたは頻度が低い、プログラムによって開始される、ユーザーの操作を変更させる可能性のあるメッセージに適しています。



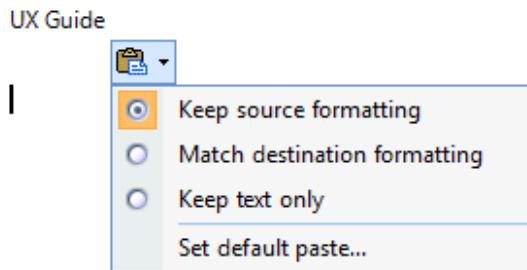
次に、効果的でない例を紹介します。

- モーダル ダイアログ ボックスは、ユーザーが変更する必要のないオプションを提示する、頻度が高いタスクには適していません。この場合は、ユーザーに確認せずに既定のオプションを使用し、後からユーザーが変更できるようにします。

間違った例:



正しい例:



正しい例では、ユーザーが Microsoft® Word でテキストを貼り付けたときに、モーダル ダイアログ ボックスではなく、モードレスなスマート タグが表示されています。このようにすると、ユーザーは応答する必要がありません。

- モーダル ダイアログ ボックスは、ユーザーの操作を変更させる可能性が低いメッセージには適していません。この場合は、通知やステータスバーを使用するか、または何もしません。

間違った例:

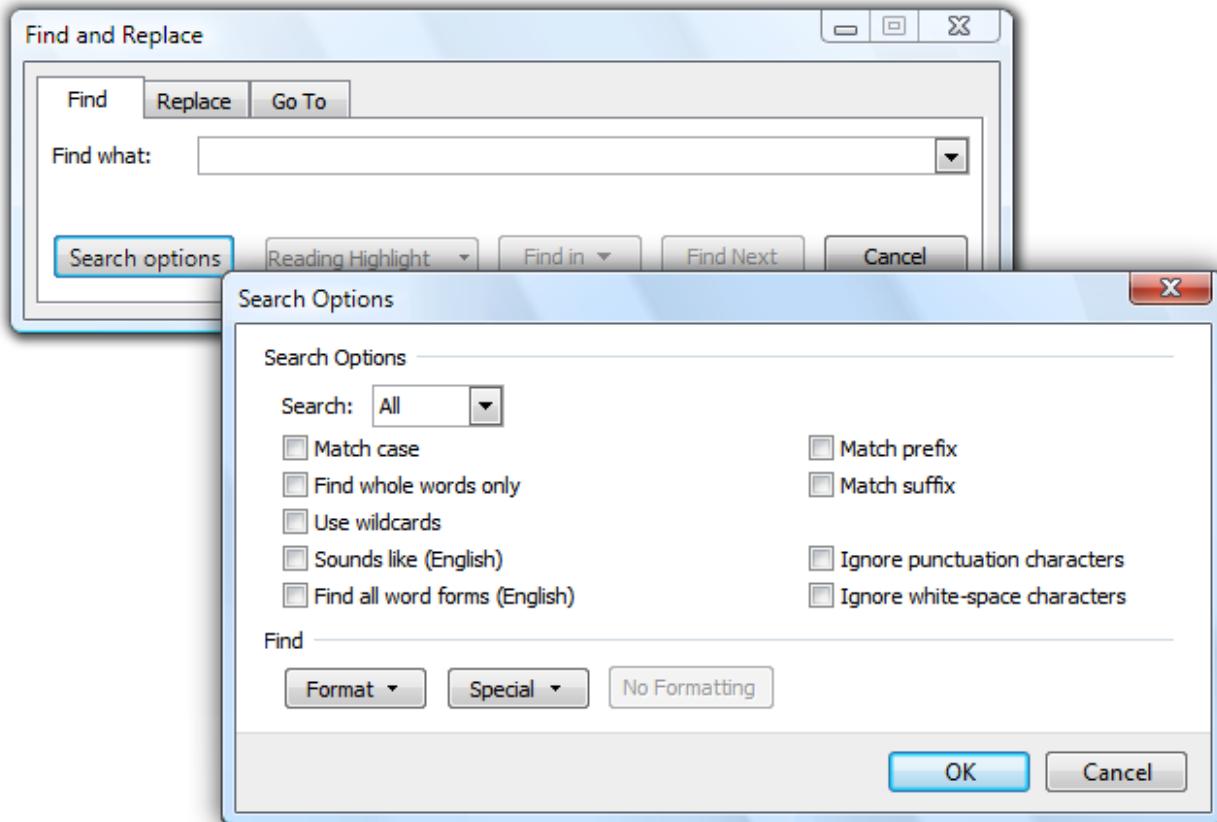


正しい例:

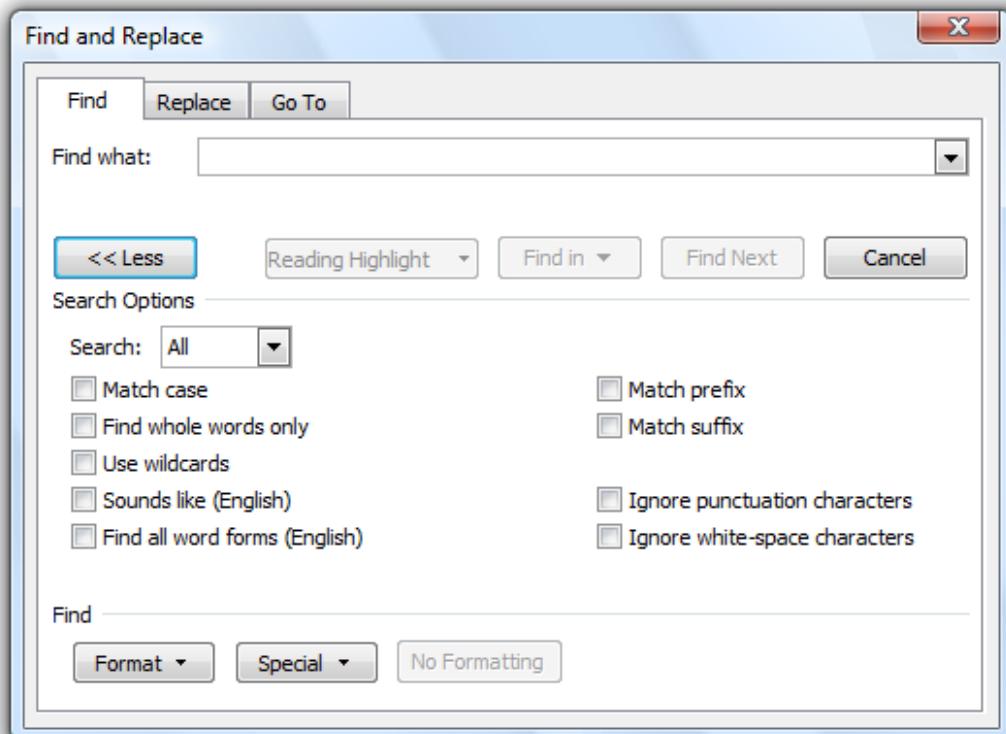
モーダル ダイアログ ボックスを使用して状態を表示するとわずらわしくなります。正しい例では、Microsoft Outlook® で、この目的のためにステータス バーが適切に使用されています。

- 多くの場合、モーダル ダイアログ ボックスは、重要なオプションの表示には適していません。この場合、インプレース UI に直接表示するか、必要に応じて段階的表示を使用して、重要なオプションを表示します。

間違った例:



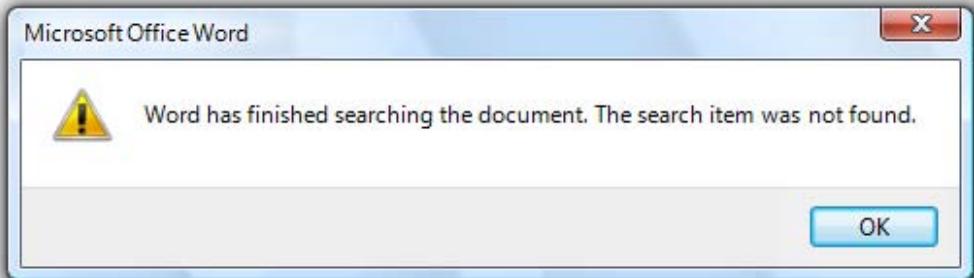
正しい例:



この例では、子ダイアログ ボックスで検索オプションを設定できますが、Word では段階的表示を正しく使用しています。

- フィードバックには確認が必要なため、モーダル ダイアログ ボックスは、あまり重要ではないフィードバックの提供には適していません。

間違った例:



この例では、Word でドキュメントの検索が終了したことを示すために、モーダル ダイアログ ボックスを使用しています。ユーザーは [OK] ボタンをクリックして、確認する必要があります。この場合は、モードレスなフィードバックの方が適しています。

#### 最も重要な点

ダイアログ ボックスの設計(目的、種類、およびユーザー操作によって決まる)が使用状況(コンテキスト、ユーザー操作の可能性、および表示頻度によって決まる)に合っていることを確認します。

#### 効果的でないダイアログ ボックスの改善

不適切なモーダル ダイアログ ボックスを解決する最善の方法は、ダイアログ ボックスを削除するか、モーダルではない方法で設計し直すことです。これをモーダル ダイアログ ボックスはすべて不適切で、削除する必要があるという意味だと判断する人がいます。この判断は正しくありません(ただし、Web ベースのアプリケーションは例外で、ポップアップ ブロック機能ができるだけ回避する必要があります)。たとえば、ユーザー操作が "必要" なタスクの場合は、モーダル ダイアログ ボックスを使用してユーザーの作業を中断させることはまったく問題がありません。モーダル ダイアログ ボックスは、多くの状況に適しています。すべてのモーダル ダイアログ ボックスを削除しようとすると、UI 全体の品質が低下する場合があります。

効果的でないダイアログ ボックスがある場合は、次の代替手段を検討します。

- ダイアログ ボックスがモーダルである必要がない場合は、モードレス ダイアログ ボックス、作業ウィンドウ、またはその他のモードレスな UI を使用します。
- 確認のみを求めるダイアログ ボックスの場合は、対処しやすいようにコマンドを追加します。たとえば、ダイアログ ボックスが問題を提示している場合は、問題の解決策も提供します。
- 「適切なユーザー インターフェイスかどうかの判断基準」を参照して、より良い解決策があるかどうかを調べます。
- ダイアログ ボックスを削除してもユーザー エクスペリエンスに悪影響が出ない場合は、削除します。

#### [今後、この <アイテム> を表示しない] オプション

オプションのモーダル ダイアログ ボックスは、特に頻繁に表示される場合に、わずらわしいと感じます。通常、こうしたダイアログ ボックスでは、ユーザーに繰り返し発生する状況やこうした状況に対処するための機能について通知します。この問題を解決する一般的な方法は、今後そのダイアログ ボックスが表示されないようにする [今後、この<アイテム> を表示しない] オプションを用意することです。



この例では、繰り返し発生する状況をユーザーに詳しく説明しているダイアログ ボックスを示しています。[今後、この<アイテム>を表示しない] をオンになると、今後そのダイアログ ボックスは表示されません。

この解決策には問題があります。ユーザーが以下を行うことを前提にしているためです。

- どのダイアログ ボックスを表示しないようにするかについて、感情（「しつこい」）ではなく、今後の必要性に基づいて理性的な判断をする。
- 今後繰り返し発生する状況を正確に認識できる。
- どれだけ前にダイアログ ボックスが表示されたかにかかわらず、前回表示されたダイアログ ボックスの情報を記憶しており、適用できる。

これらの前提条件が1つでも揃わない場合、必要なときにダイアログ ボックスが表示されないだけでなく、表示しないようにしたダイアログ ボックスを元に戻す明確な方法はありません。

ダイアログ ボックスに [今後、この<アイテム>を表示しない] オプションが必要だと判断するということは、そのダイアログ ボックスがわずらわしく、必要がない可能性があるということでもあります。このオプションを追加する前に、次の代替手段を検討します。

- メッセージを表示する必要がないように、設計を変更する。
- 設計に関して、このダイアログ ボックスを表示する必要が本当にあるのかどうか、また、ユーザーにこの情報を知らせない場合、悪影響が生じるかどうかを十分に検討する。悪影響が生じる場合は常に表示させます。それ以外の場合は、まったく表示させません。

間違った例:



この例では、Outlook のアラームを削除しても悪影響が生じないため、ユーザーはこの確認を必要としないません。したがって、この確認を表示する必要はまったくありません。

- インプレース UI、段階的表示、モードレスな UI (バルーン、ツールヒントなど)、通知などのあまりわざらわしくない方法を使用します。

既定で、適切な設定を行います。不適切な初期設定の修正をユーザーに実行させないでください。プログラムのさまざまな場所で不必要的モーダル ダイアログ ボックスを使用すると、ユーザーに詳しい情報を与えるよりも、反感を買う可能性が高くなることに注意してください。ある時点で、ユーザーはそうしたダイアログ ボックスを読まずに、閉じるようになります。

情報をしばらくの間表示する必要が本当にあり、ダイアログ ボックスが最善の方法であるという確信がある場合にのみ、[今後、この<アイテム>を表示しない] オプションを使用します。ユーザーがこれらのダイアログ ボックスを元に戻す必要がある場合は、プログラムの [オプション] ダイアログ ボックスに "メッセージの復元" コマンドを追加します。

ヒント: 作成したプログラムを実行しているときに、[今後、この<アイテム>を表示しない] の既定値を変更しないようにします。こうすることで、ユーザーと同じ方法でプログラムの操作性を評価できます。

## ダイアログ ボックスの使用パターン

ダイアログ ボックス

ダイアログ ボックスのデザインコンセプト

ダイアログ ボックスには、いくつかの使用パターンがあります。

質問ダイアログ

ボックス(ボタンを使用)

ユーザーに1つ

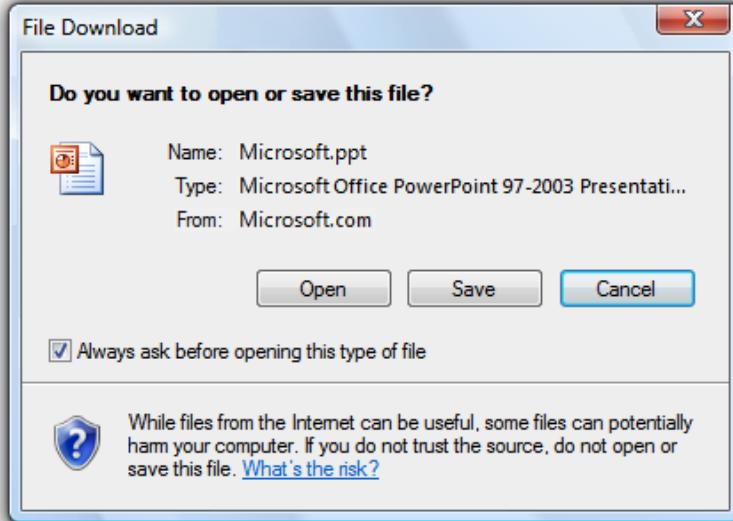
質問をし、簡単

な応答を水平に

配置されたコマ

ンド ボタンで提

示します。



この *Windows® Internet Explorer®* の例では、ユーザーにファイルを開くか、保存するかを質問しています。

種類: モーダル。

メイン指示テキスト: たずねる質問(指示テキストとして記述できます)。

アイコン: プログラム、機能、オブジェクト、警告アイコン(データが消失する可能性やシステム アクセスが失われる可能性がある場合)、セキュリティ警告、またはなし。

コミット ボタン: [はい]/[いいえ]、[はい]/[いいえ]/[キャンセル]、"実行する"/[キャンセル]、"実行する"/"実行しない"、"実行する"/"実行しない"/[キャンセル])のいずれか。"実行する" および "実行しない" には、メイン指示テキストに対する具体的な応答が入ります。

その他のコントロール: ユーザーが情報に基づいて判断する際に役立つ補足説明、詳細情報を表示するシェブロン コントロール、および今後その質問が表示されないようにできる [今後、この<アイテム>を表示しない] オプションを使用できます。

迷惑度: 既定の応答を問題なく推測できる、本当に選択肢がない、または選択肢の違いが不明確な場合は高くなります。

確認も質問として表示されますが、これについては「[確認](#)」で具体的に説明します。

質問ダイアログ

ボックス(コマンド リンクを使用)

ユーザーに1つ

質問をするか、

実行するタスク

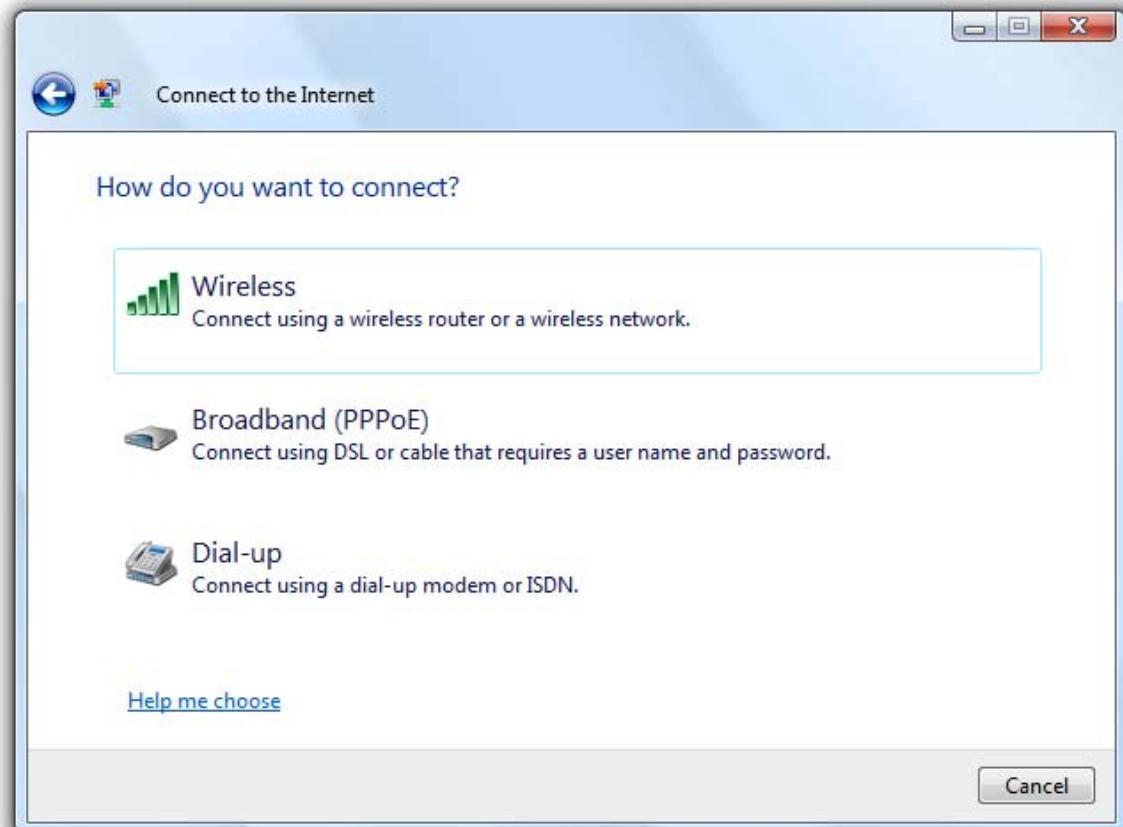
の選択を求め、

詳細な応答を垂

直に配置された

コマンド リンク

で提示します。



この例では、ユーザーは *Windows* からデバイスのインストールを求められています。コマンド ボタンの代わりにコマンド リンクを使用することで、詳細な応答を提示できます。

コマンド ボタンの場合とは異なり、このダイアログ ボックスでは、複数の応答や長いテキストを使用して説明する必要がある応答を表示できます。

種類: モーダル。

メイン指示テキスト: たずねる質問 (指示テキストとして記述できます)。

アイコン: プログラム、機能、オブジェクト、警告アイコン (データが消失する可能性やシステム アクセスが失われる可能性がある場合)、セキュリティ警告、またはなし。

コマンド リンク: メイン指示テキストに対する 2 つ以上の詳細で具体的な応答。

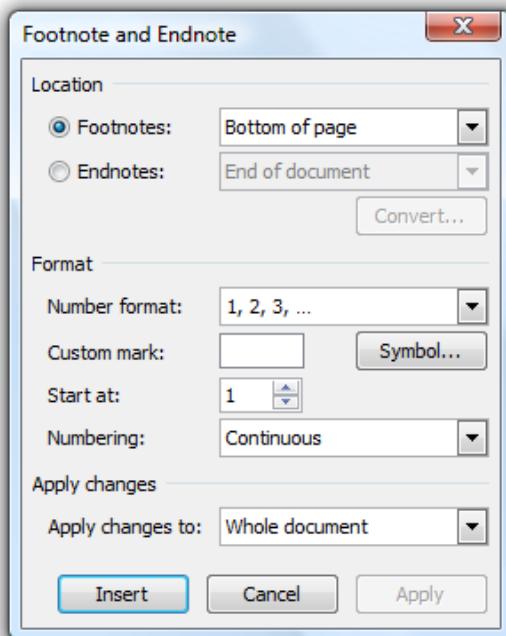
コミット ボタン: [キャンセル]。

その他のコントロール: ユーザーが情報に基づいて判断する際に役立つ補足説明、および詳細情報を表示するシェブロンを使用できます。

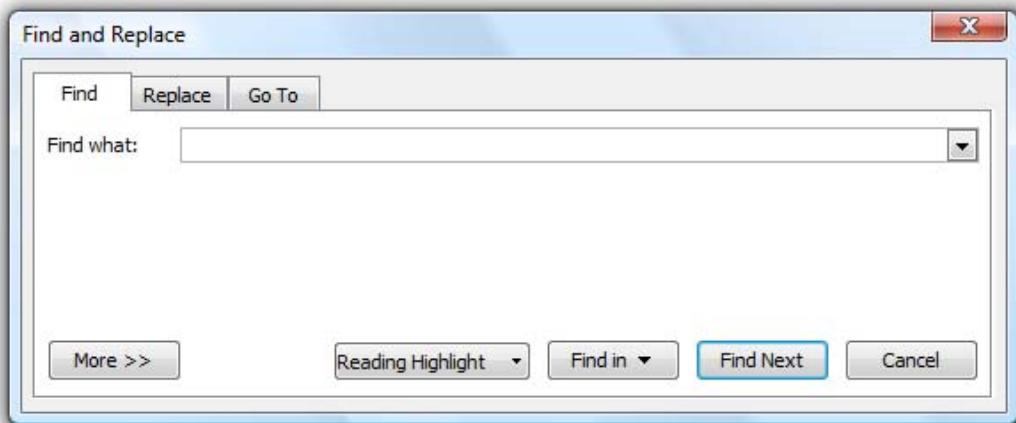
迷惑度: 既定の応答を問題なく推測できる、本当に選択肢がない、または選択肢の違いが不明確な場合は高くなります。

確認も質問として表示されますが、これについては「[確認](#)」で具体的に説明します。

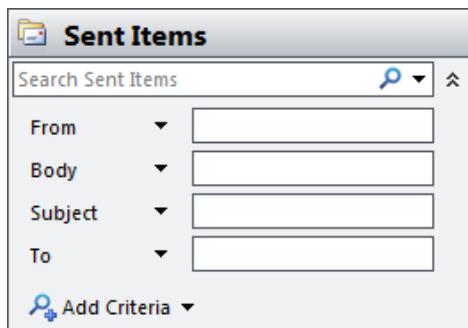
選択ダイアログ  
ボックス  
一連の選択肢を  
ユーザーに提示  
します。通常、  
コマンドをより  
正確に指定する  
ために使用しま  
す。質問ダイア  
ログ ボックスと  
は異なり、選択  
ダイアログ ボッ  
クスでは複数の  
質問をすること  
ができます。



この例では、モーダルダイアログ ボックスに、Microsoft Word の [改ページの挿入] コマンドを指定するためのオプションが表示されています。



この例では、モードレスダイアログ ボックスに、Word の [検索と置換] コマンドを指定するためのオプションが表示されています。



この例では、作業ウィンドウに、Microsoft Outlook® の [検索] コマンドを指定するためのオプションが表示されています。別のウィンドウを使用しないことで、コマンドがより直接的で、コンテキストに即しているという印象を与えます。

**種類:** モーダル、モードレス、および作業ウィンドウ。

**メイン指示テキスト:** ユーザーに作業内容を伝える、オプションの重要な指示テキスト。

**アイコン:** なし。

**コミット ボタン:** 次のいずれか。

- モーダルダイアログ ボックス: [OK]/[キャンセル] または "実行する"/[キャンセル]。"実行する"には、メイン指示テキストに対する具体的な応答が入ります。
- モードレスダイアログ ボックス: ダイアログ ボックスとタイトルバーの [閉じる] ボタン。

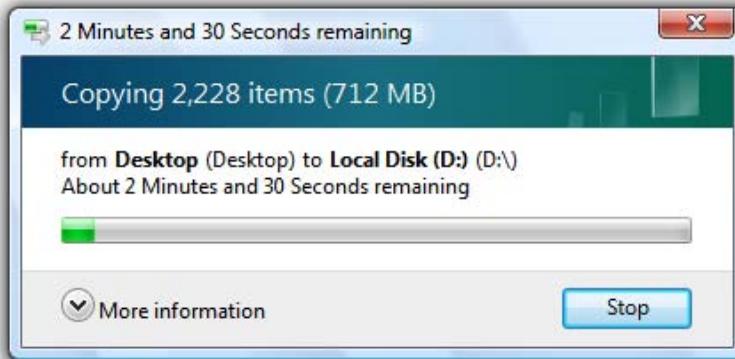
- 作業ウィンドウ: タイトルバーの [閉じる] ボタン。

その他のコントロール: 選択する際に役立つ補足説明およびあまり使用されないオプションを表示するシェブロンを使用できます。

迷惑度: ユーザーが開始し、応答を必要としているため、通常は低くなります。ただし、ユーザーが既定値をほとんど変更しない場合は高くなります。

**進行状況ダイアログ ボックス**  
長い時間がかかる処理(5秒を超える)の進行状況フィードバックと処理の取り消しや停止のためのコマンドをユーザーに提示します。

処理に時間がかかるタスク(30秒を超える)をバックグラウンドで実行できる場合は、ユーザーがその間にプログラムを継続して使用できるように、モードレスな進行状況ダイアログ ボックスを使用します。



この例では、モードレスな進行状況ダイアログ ボックスを使用して、ユーザーがプログラムを継続して使用している間に、フィードバックを提供しています。

種類: モーダルおよびモードレス。

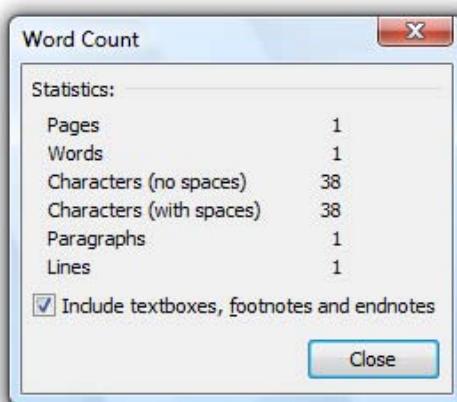
メイン指示テキスト: 末尾に省略記号を付けた、進行中の処理を簡潔に説明する語句。例: ダウンロード中...

アイコン: なし(ただし、アニメーションを表示できます)。

コミットボタン: 環境を(副次的な影響を残さず)元の状態に戻す場合は、[キャンセル]を使用します。それ以外の場合は、[停止]を使用します。

迷惑度: いつ処理が完了したかをユーザーに知らせる必要がある場合は低くなります。不必要にモーダルになっている、または処理が重要でない場合は高くなります。

**情報ダイアログ ボックス**  
ユーザーが要求した情報を表示します。



この例では、Word でモーダル ダイアログ ボックスを使用して、文字カウント情報を表示しています。

種類: モーダル。

メイン指示テキスト: 情報を説明する文。

アイコン: なし。

コミットボタン: [閉じる]。

その他のコントロール: 詳細情報を表示するシェブロンを使用できます。

迷惑度: ユーザーが要求した情報に関連性がある場合は低くなります。

## コモン ダイアログ ボックス

デザインコンセプト

適切なユーザーインターフェイスかどうかの判断基準

ガイドライン

全般

ファイルを開く

ファイルの保存

ファイルの種類の一覧

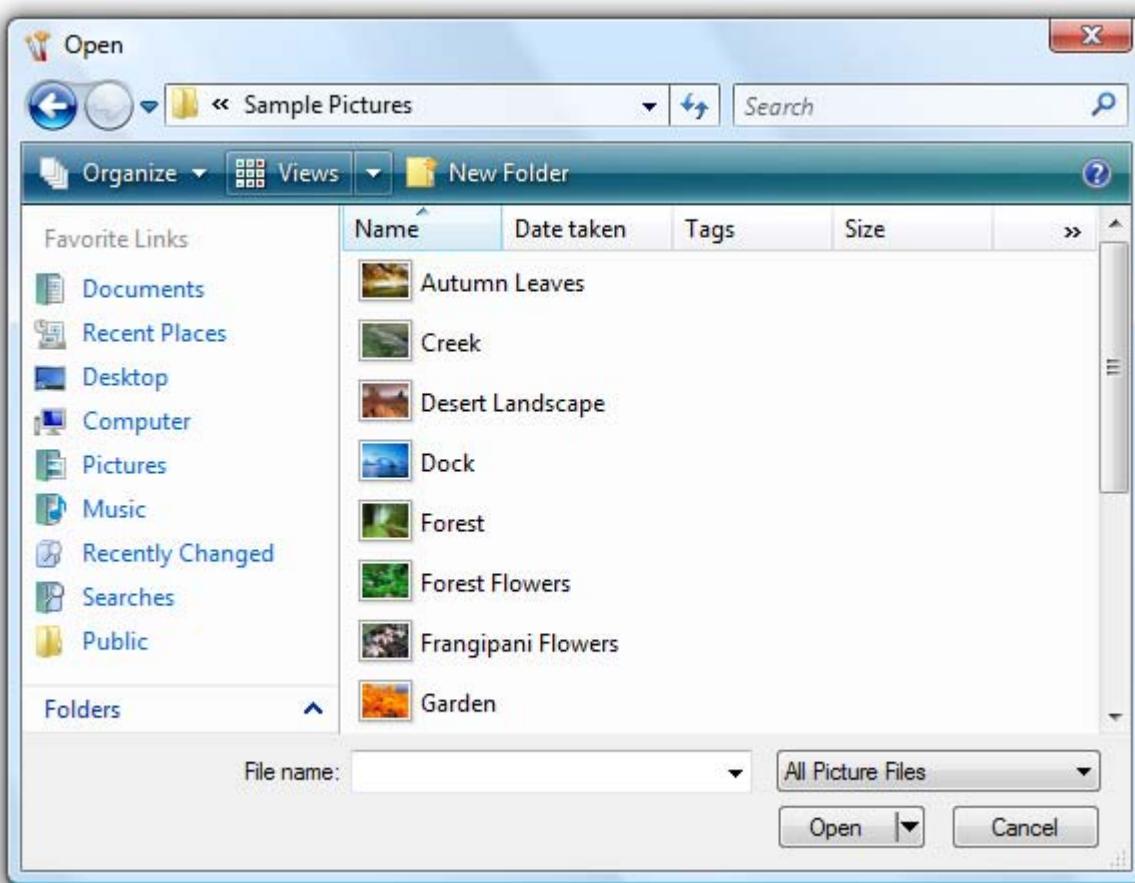
フォルダーを開く

フォント

値の保持

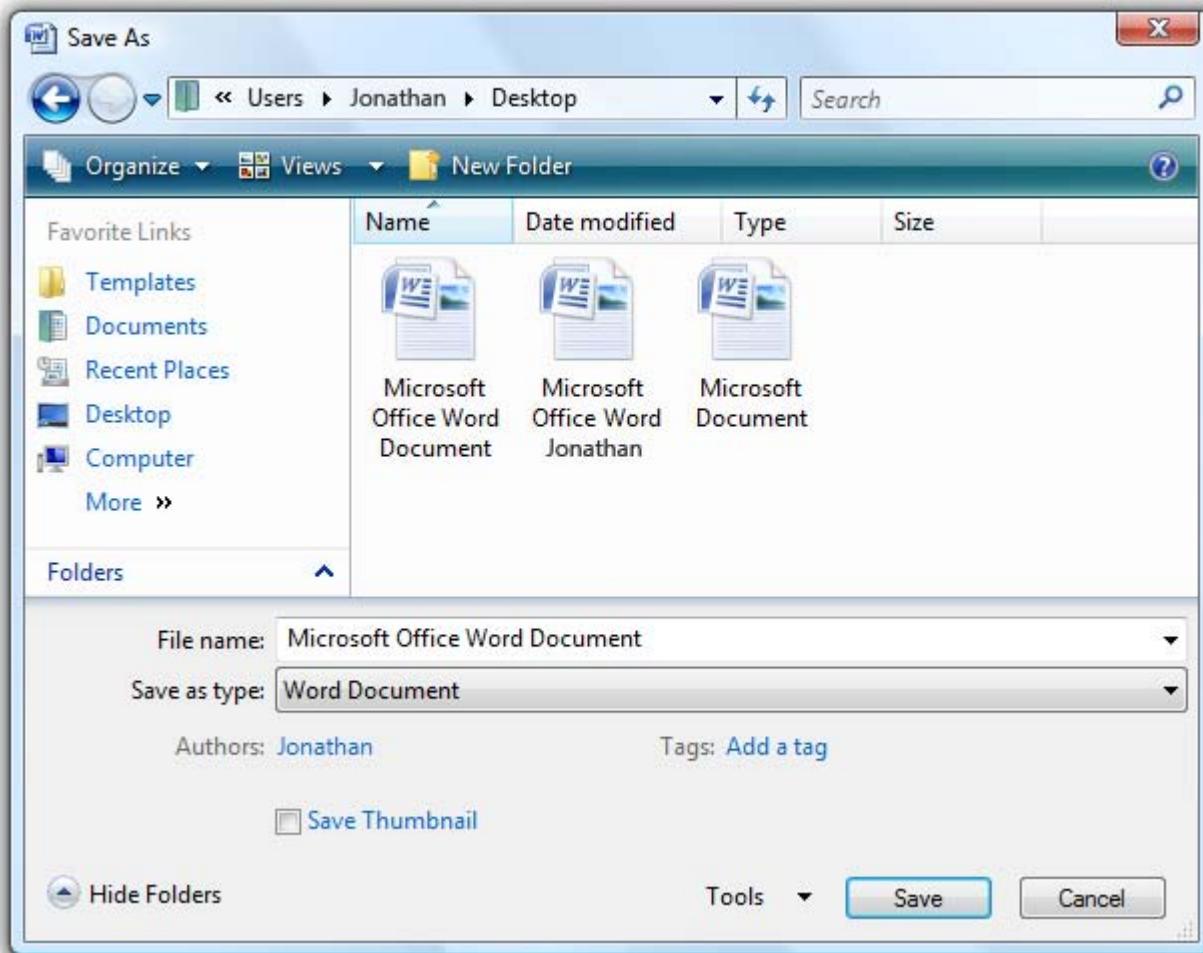
Microsoft® Windows® のコモン ダイアログ ボックスは、"ファイルを開く"、"ファイルの保存"、"フォルダーを開く"、"検索と置換"、"印刷"、"ページ設定"、"フォント"、"色" の各ダイアログ ボックスで構成されます。

ファイルを開く



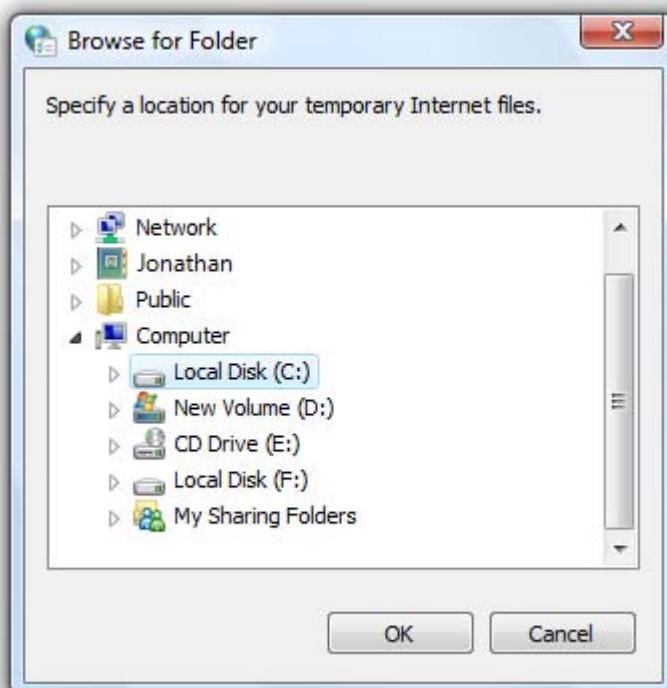
プログラムで使用するアイテムをすばやく見つけられるように最適化されています。

ファイルの保存



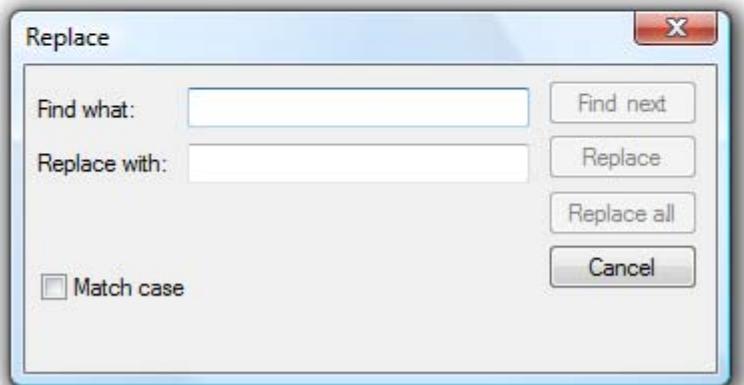
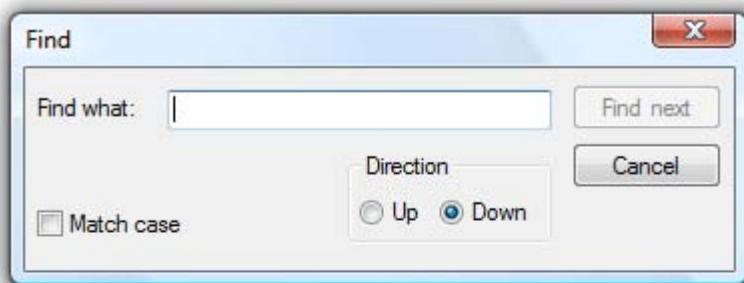
ファイルをメタデータと共に保存することで、ループを閉じます。

フォルダーを開く



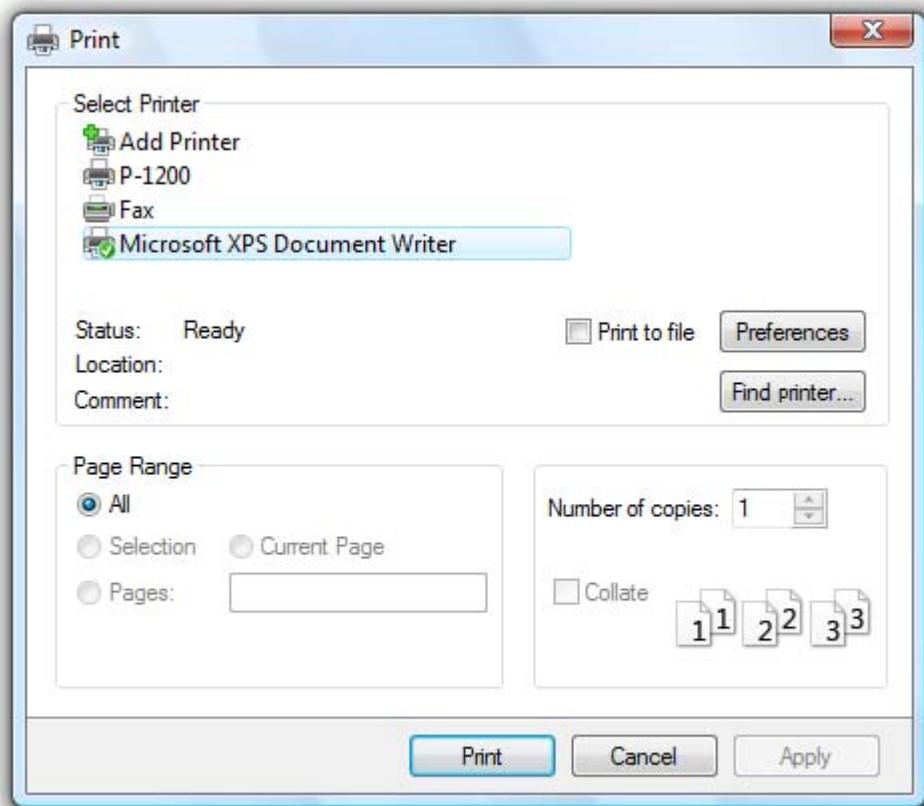
フォルダーを選択するための専用ダイアログ ボックスです。

検索と置換



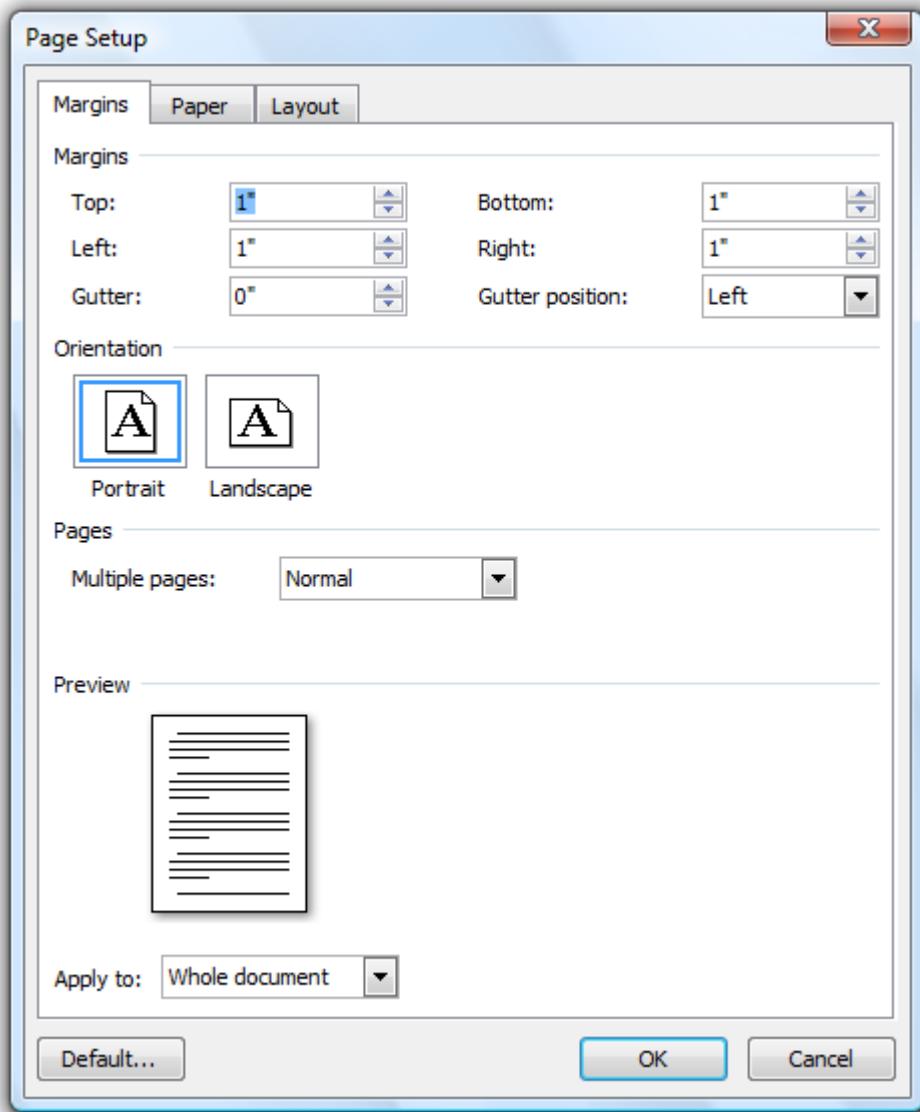
[検索] では、ユーザーはテキスト文字列を検索できます。検索のオプションである [置換] では、ユーザーは一致した文字列を別の文字列に置換できます。

## 印刷



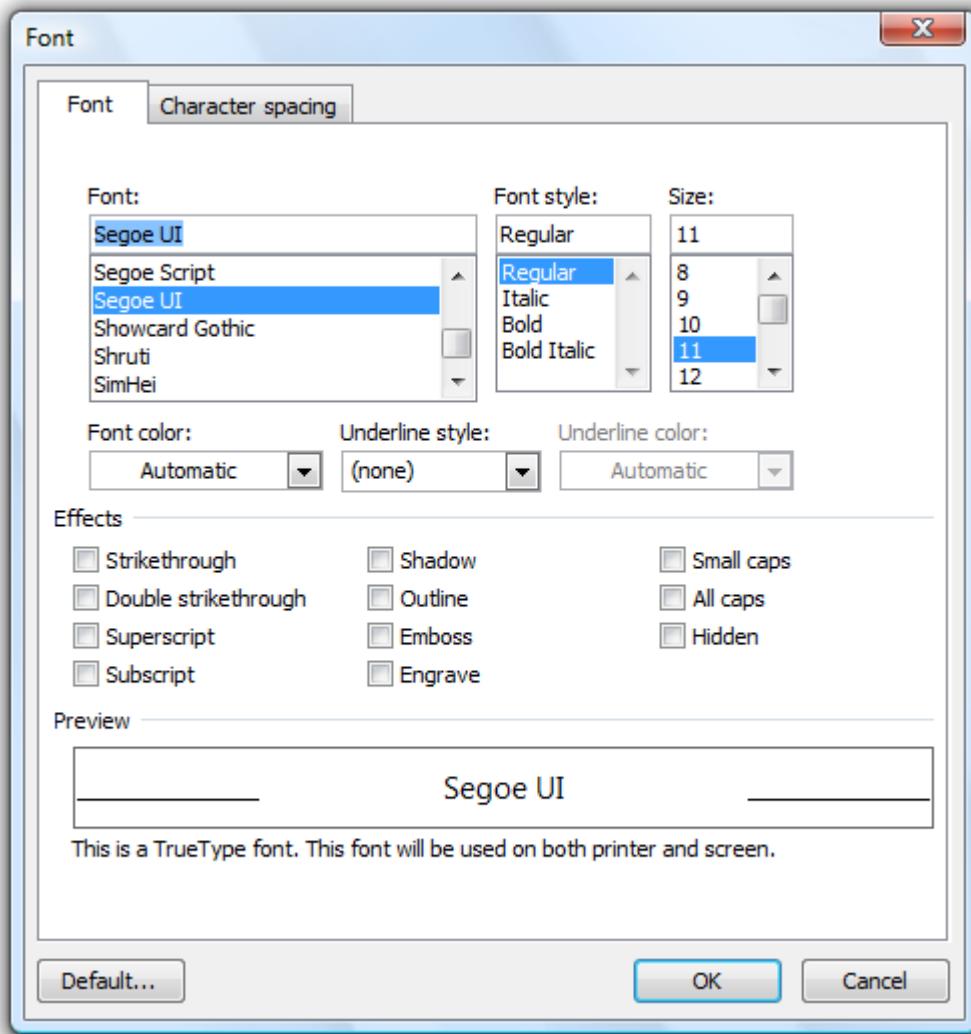
ユーザーは、プリンターを選択して構成できるほか、印刷対象、印刷部数、部単位で印刷するかどうかを選択できます。

## ページ設定



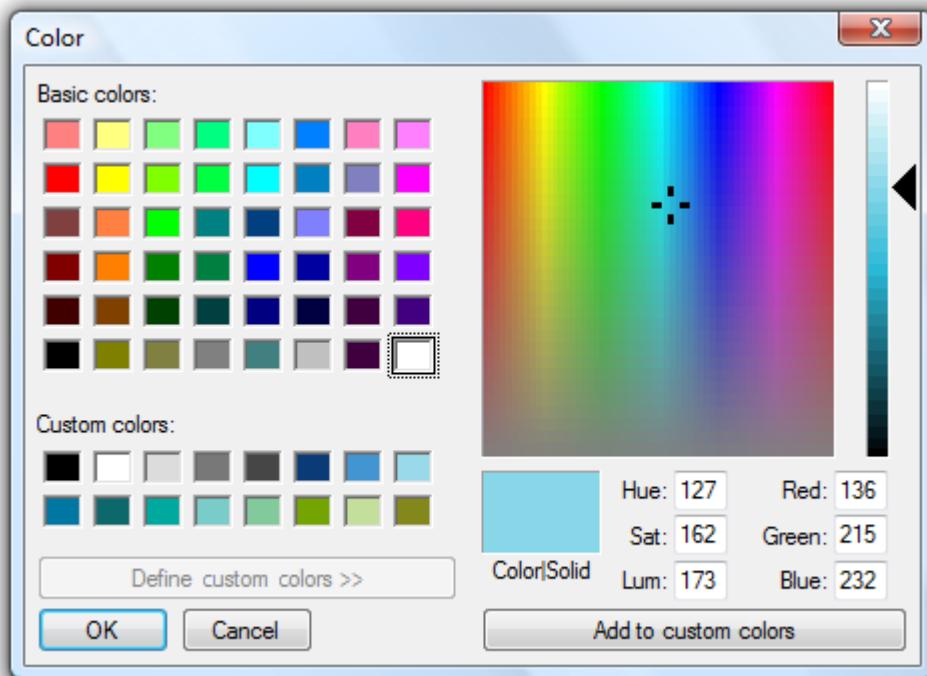
ユーザーは、用紙サイズと給紙方法、印刷の向き、余白を選択できます。

フォント



インストールされている使用可能フォントとポイント サイズが表示されます。

## 色



ユーザーは、事前定義された色のセットから色を選択するか、作成した色を選択できます。

## デザイン コンセプト

コモン ダイアログ ボックスを使用すると、異なるプログラム間でも一貫したエクスペリエンスをユーザーに提供できます。コモン ダイアログ ボックスを上手に使用することで、効率的で有意義なエクスペリエンスを提供することもできます。

以下について最適な既定値を選択したコモン ダイアログ ボックスを使用すると、ユーザーのエクスペリエンスを大幅に向上できます。

- 入力値(例: 既定のフォルダー、既定のファイル名)
- 選択済みオプション(例: 選択したプリンター、印刷オプション)
- ビュー(例: 縮小表示での画像表示、ファイル名なしでの画像表示、日付による並べ替え、列の幅)
- 提示方法(例: ウィンドウ サイズ、位置、コンテンツ)

"初回の" 既定値と、"2 回目以降の" 既定値の両方を決定する必要があります。初回の既定値はプログラムにより、ターゲット ユーザーに予想される使用方法に基づいて決定されます。一方、2 回目以降の既定値は、実際の使用方法に基づいて決定されます。過去の使用方法が、最も可能性のある将来の使用方法を示す指針になります。

プログラムの既定値が有用かどうかを判断するには、最もよく行うタスクについて、ユーザーが実行しなければならない手順の数を確認します。ユーザーがタスクを行うたびに同一の手順を繰り返す場合、その手順は不要な可能性があり、既定値を変更する余地があります。

### 最も重要な点

適切な初回の既定値と 2 回目以降の既定値を選択して、ユーザーに効率的で有意義なエクスペリエンスを提供します。

## 適切なユーザー インターフェイスかどうかの判断基準

コモン ダイアログ ボックスの使用を強く推奨します。一貫したユーザー エクスペリエンスのために、コモン ダイアログ ボックスを使用し、独自にダイアログ ボックスを作成しないようにします。名前空間で正しく安全に機能するようなカスタム UI を作成することは、とても困難です。コモン ダイアログ ボックスは、必要であればカスタマイズできます。

Windows Vista® では、"ファイルを開く" と "ファイルの保存" に拡張可能な新しいアーキテクチャが採用されており、追加機能の公開がより簡単になっています。このしくみは、大手の独立系ソフトウェア ベンダー (ISV) の最小要件に十分対応できるほど柔軟なものであり、今後リリースされる Windows でも存続します。

## ガイドライン

### 全般

- 適切な場合は、より直接的、または[モードレス](#)な代替方法を提供します。つまり、ユーザーが次の操作を実行できるようにします。
  - ファイルをプログラムにドロップして開く
  - [保存] コマンドを使用して、現在の名前で現在の場所にファイルを保存する
  - F3 キーを使用して、文字列の次の出現を検索する
  - [印刷] コマンドを使用して、ドキュメント全体を既定のプリンターに 1 部印刷する
  - ツールバーやパレット ウィンドウを使用して、フォントやフォント属性を変更する
  - ツールバーやパレット ウィンドウを使用して、色を変更する
- コモン ダイアログ ボックスを表示するには、次のコマンドを使用します(適切な[アクセス キー](#)と共に提供します)。

コモン ダイアログ ボックス	コマンド
ファイルを開く	[開く(O)...]
ファイルの保存	[名前を付けて保存(A)...]

フォルダーを開く	[フォルダーを開く(E)...] または [フォルダーの選択(E)...]
検索と置換	[検索(E)...] または [置換(R)...]
印刷	[印刷(P)...]
ページ設定	[ページ設定(U)...]
フォント	[フォント(E)...] または [フォントの選択(E)...]
色	[色(C)...] または [色の選択(C)...]

- 必要に応じて、より具体的なコマンドを使用することもできます。たとえば、ファイルをエクスポートするコマンドとして、[名前を付けて保存] の代わりに [ファイルのエクスポート] を使用します。
- ダイアログ ボックスのタイトルは、そのダイアログ ボックスを起動したコマンドを反映したタイトルにします。たとえば、[ファイルのエクスポート] コマンドから "ファイルの保存" が起動される場合、ダイアログ ボックスのタイトルを "ファイルのエクスポート" に変更します。

## ファイルを開く

- 初回の既定フォルダーには、ピクチャ、ミュージック、ビデオなどの専用フォルダーが適切な場合は専用フォルダーを使用し、それ以外の場合はドキュメント フォルダーを使用します。
- 2回目以降の既定フォルダーには、ユーザーがプログラムを使用して最後に開いたフォルダーを使用します。
- 写真ファイルを開くときは、既定でファイル名を非表示にします。通常、写真は縮小表示で識別できます。また、意味を成さない名前が付いていることもあります。

## ファイルの保存

- 新規ファイルを初めて保存する場合の初回の既定フォルダーには、ピクチャ、ミュージック、ビデオなどの専用フォルダーが適切な場合は専用フォルダーを使用し、それ以外の場合はドキュメント フォルダーを使用します。
- 一時ファイルには、現在のユーザーの一時フォルダーを使用します。ファイル名は、~DF1A92.tmp ではなく File0001.tmp のように、平易な一意のファイル名を使用します。
  - 開発者向け情報: 現在のユーザーの一時フォルダーは、GetTempPath API 関数を使用して取得できます。
- 初回の既定のファイル名には、次を基準にした一意の既定の名前を使用します。
  - ファイルの内容(認識できる場合)。たとえば、ドキュメントの最初の数語をファイル名にします。
  - ユーザーが選択したパターン。たとえば、前のファイル名が "Hawaii 1.jpg" の場合、次のファイル名には "Hawaii 2.jpg" を選択します。
  - ファイルの種類に基づいた一般的なパターン。たとえば、"Photo1.jpg" というファイル名にします。
- ファイルが既に存在する場合、2回目以降の既定値には、ファイルの現在のフォルダーと名前を使用します。
- ファイルを保存する際は、作成日を保持します。プログラムでファイルを保存する際に、一時ファイルを作成し、元のファイルを削除して、一時ファイルに元のファイル名を付ける作業を行う場合は、必ず元のファイルから作成日をコピーします。
- ユーザーがファイル名を指定しないで [保存] コマンドを選択した場合は、"ファイルの保存" を使用します。

## ファイルの種類の一覧

注: ファイルの種類の一覧は、"ファイルを開く" と "ファイルの保存" で使用され、表示されるファイルの種類と既定のファイル拡張子を決定します。

- ファイルの種類の一覧が短い(5項目以下)の場合は、使用頻度の高そうな順に項目を並べ替えます。リストが長い(6項目以上)の場合は、見つけやすいようにアルファベット順に並べ替えます。
- "ファイルの保存" の場合、サポートされているファイル拡張子を、一般的でないものも含めて全種類挙げ、先頭には最も一般的な拡張子を挙げます。ファイル処理ロジックでは、サポートされているファイル拡張子をユーザーが指定したかどうかを判断するときに、このリストが最初に確認されます。たとえば、JPEG というファイルの種類の一覧に .jpg と .jpeg のみが含まれている場合、test.jpe というファイルは test.jpe.jpg として保存される可能性があります。

- ・ "ファイルの保存" の場合、初回の既定のファイルの種類は、ターゲットユーザーが選択する可能性が最も高い種類にします。2回目以降の既定のファイルの種類は、ファイルの現在の種類にします。
- ・ "ファイルを開く" の場合、初回の既定のファイルの種類は、ターゲットユーザーが選択する可能性が最も高い種類にします。2回目以降の既定のファイルの種類は、最後に使用されたファイルの種類にします。
- ・ "ファイルを開く" の場合、ユーザーがどの種類のファイルも開けるようにするときや、ユーザーがフォルダー内の全ファイルを同時に確認する必要がありそうなときには、先頭の項目として "すべてのファイル" を挙げます。可能であれば、"すべての画像"、"すべての音楽"、"すべてのビデオ" のような他のメタ フィルターを、"すべてのファイル" のすぐ後に挙げます。
- ・ "ファイルの種類名 (\*.拡張子1; \*.拡張子2)" の形式を使用します。ファイルの種類名は登録されている種類名である必要があります。これは、コントロールパネルの [フォルダー オプション] (Windows Vista では [既定のプログラム]) で確認できます。たとえば、"HTML ドキュメント (\*.htm; \*.html)" のようにします。
  - ・ 例外: メタ フィルターの場合は、煩雑にならないようにファイル拡張子リストを削除します。たとえば、"すべてのファイル"、"すべての画像"、"すべての音楽"、"すべてのビデオ" のようにします。
- ・ ファイルの種類名には[センテンス スタイルの大文字化](#)を使用し、ファイル拡張子には小文字を使用します。

## フォルダーを開く

- ・ 新しいプログラムの場合は、"ファイルを開く" ダイアログ ボックスを "ピック フォルダー" モードで使用します。この場合は Windows Vista 以降が必要です。Vista 以前のバージョンの Windows 上で動作するプログラムの場合は、"フォルダーを開く" ダイアログ ボックスを使用します。
  - ・ 開発者向け情報: "ファイルを開く" ダイアログ ボックスを "ピック フォルダー" モードで使用するには、FOS\_PICKFOLDERS フラグを使用します。

## フォント

- ・ 必要であれば、フォント一覧をフィルタリングして、プログラムで使用できるフォントだけを表示できます。

## 値の保持

- ・ 2回目以降の既定値として使用するために、以下の値を保持するようにします。
  - ・ 入力値 (例: 既定のフォルダー、既定のファイル名)
  - ・ 選択済みオプション (例: 選択したプリンター、印刷オプション)
  - ・ ビュー (例: 縮小表示での画像表示、ファイル名なしでの画像表示、日付による並べ替え、列の幅)
  - ・ 提示方法 (例: ウィンドウ サイズ、位置、コンテンツ)

例外: ユーザーが1からやり直すような方法で使用する可能性が高いコモン ダイアログ ボックスについては、これらの値は保持しません。
- ・ 既定値を決定するにあたっては、重要なシナリオに基づき、ターゲットユーザーが必要とする可能性の最も高い値を考えます。また、1回のプログラム使用におけるシナリオ、連続使用および同時使用におけるシナリオ、複数のドキュメントにわたる場合のシナリオを考えます。役に立つと思われない状況では、値を保持しません。
  - ・ 例: 典型的なドキュメントベースのアプリケーションの場合、1回のプログラム使用におけるシナリオと連続使用におけるシナリオでは、"ファイルを開く" と "ファイルの保存" の設定を保持するのが便利です。同時使用におけるシナリオでは、それぞれの設定を優先する方が、ユーザーは一度に複数のドキュメントに対して効率的に作業できます。
- ・ プログラムごと、ユーザーごとに設定を保持します。

## ウィザード

このコンテンツはまだ完成していません。当面は以下の情報をご利用ください。

### Windows Vista の新機能

Microsoft Windows Vista のウィザードのデザインは、次のように変更されています。

- 柔軟性の高いページレイアウトおよびテキスト書式設定により、情報をより適切に表現できます。
- ページのサイズ変更が可能です。
- 効率を向上させるため、不要なウェルカムページおよび設定完了/終了ページが削除されました。
- ページのタイトルおよびサブタイトルの代わりに、目に留まりやすい[メイン指示テキスト](#)が表示されます。
- メイン指示テキストと柔軟性の高いレイアウトにより、Wizard '97 で見られた多数の繰り返しを排除できます。
- [コマンドリンク](#)を使用すると、即座に実行できる、わかりやすい選択肢を提示できるため、ラジオ ボタンと [次へ] ボタンを組み合わせて使用する必要がなくなります。
- [コミットボタン](#)とリンクは、それ自体がメイン指示テキストに対する応答を明確に示した説明であるため、効率的な意思決定と円滑で[誘導的な](#)操作フローを実現します。
- Web や Windows エクスプローラーの操作方法との一貫性が向上しています。
- [戻る] ボタンの標準的な位置が枠内の左上になり、コミット用の選択肢に注目が集まるようにしました。

### よくある逸脱

最もよくあるウィザードのガイドライン違反については、「[よくあるガイドラインからの逸脱](#)」を参照してください。

### その他の関連ガイドライン

以下の Windows ガイドラインには、ウィザードに関する資料が含まれています。

- [ダイアログ ボックス](#)
- [コントロール パネル](#)
- [レイアウト](#)
- [ウィンドウの管理](#)

### 以前のバージョンのガイドライン

[Windows のユーザー エクスペリエンス](#)のウィザードに関するガイドラインの一部も適用されます。これらのガイドラインは、前述のガイドラインと矛盾しない場合にのみ、慎重に適用してください。

## プロパティ ウィンドウ

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

プロパティシート

オプションダイアログボックス

プロパティページ

子プロパティウィンドウ

タブ

コマンドボタン

コミットボタン

ページコンテンツ

ヘルプ

標準ユーザーおよび保護された管理者

既定値

テキスト

ドキュメント

"プロパティ ウィンドウ" は、次の種類のユーザーインターフェイス (UI) の総称です。

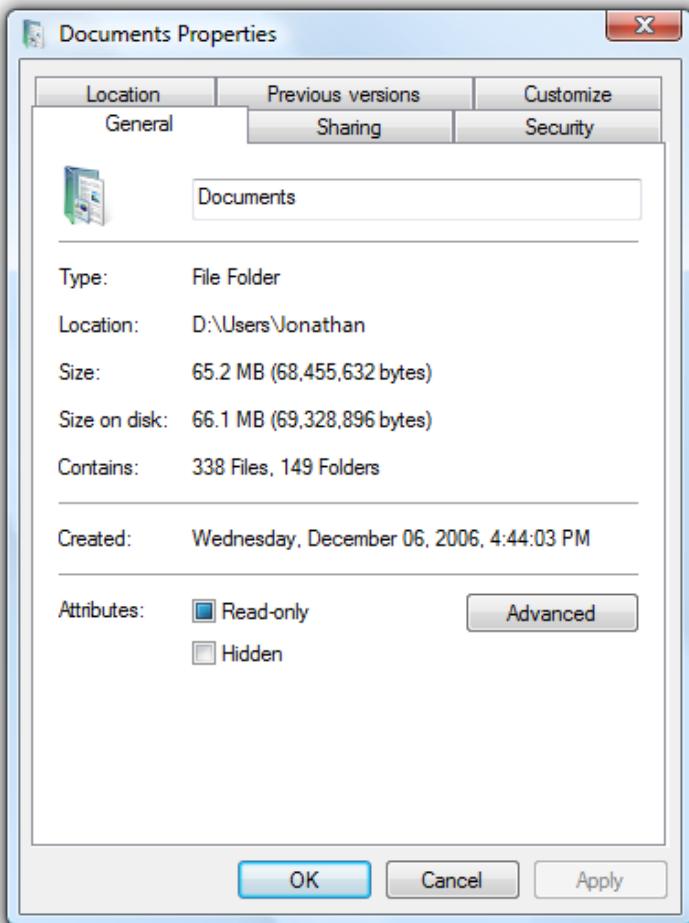
- プロパティシート: ダイアログボックスでオブジェクトやオブジェクトコレクションのプロパティを表示および変更するするために使用します。
- プロパティインスペクター: ウィンドウでオブジェクトやオブジェクトコレクションのプロパティを表示および変更するするために使用します。
- オプションダイアログボックス: アプリケーションのオプションを表示および変更するために使用します。

オブジェクトの "プロパティ" は、次のどちらかの要素です。

- ユーザーが変更できる "設定" (ファイルの名前や読み取り専用属性など)。
- ユーザーが直接変更できないオブジェクトの "属性" (ファイルのサイズや作成日など)。

ダイアログボックス (オプションダイアログボックスは除く) やウィザードとは異なり、プロパティ ウィンドウは、通常、単一のタスクではなく複数のタスクをサポートします。

一般に、プロパティ ウィンドウはページに分けられており、タブを使用してアクセスします。プロパティ ウィンドウというとタブが連想されますが (その逆の場合もあります)、タブはプロパティ ウィンドウの必須要素ではありません。



## 典型的なプロパティ シート

注: レイアウト、タブ、コントロール パネルに関するガイドラインは、それぞれ別の項目として記載しています。

## 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- ユーザーがプロパティを設定する際に、既定の複雑な手順を実行する必要があるかどうか。該当する場合は、代わりに[ウィザード](#)か[タスク フロー](#)を使用します。
- 全コンテンツがアプリケーションのオプションであるかどうか。該当する場合は、[オプション ダイアログ ボックス](#)を使用します。
- 全コンテンツがアプリケーションの属性であるかどうか。該当する場合は、[バージョン情報 ボックス](#)を使用します。
- コンテンツの大部分がオブジェクトのプロパティ (設定または属性) であるかどうか。該当しない場合は、標準的な[ダイアログ ボックス](#)か[タブ付きダイアログ ボックス](#)を使用します。
- ユーザーが、プロパティの確認と変更の作業を頻繁に、または長時間行う可能性があるかどうか。該当する場合は、[プロパティ インスペクター](#)を使用します。該当しない場合は、[プロパティ シート](#)を使用します。
- ユーザーが、何種類かのオブジェクトのプロパティを一度に確認または変更する可能性があるかどうか。該当する場合は、[プロパティ インスペクター](#)を使用します。該当しない場合は、[プロパティ シート](#)を使用します。

プロパティ シートとプロパティ インスペクターは併用できます。プロパティ インスペクターにはアクセス頻度の高いプロパティを表示し、プロパティ シートにはすべてのプロパティを表示するというように、使い分けが可能です。

## デザイン コンセプト

プロパティ ウィンドウは、テクノロジーに基づく低水準の設定を中途半端に寄せ集めた "ごみ置き場" となりがちです。こうしたプロパティはタブにまとめられてはいても、特定のタスクやユーザーを考慮せずにデザインされるケースが少なくありません。その結果、ユーザーがプロパティ ウィンドウで何らかのタスクを行うときに、何をすべきかわからなくなることがよくあります。

高い実用性と操作性を兼ね備えたプロパティ ウィンドウをデザインするには、次の手順に従ってください

さい。

- プロパティの必要性を確認します。
- テクノロジーではなく、ユーザーの目的という観点からプロパティを表示します。
- 適切なレベルのプロパティを表示します。
- 特定のタスク向けのページをデザインします。
- 特定のユーザー、特に制限付きユーザー(管理者以外)向けのページをデザインします。
- プロパティページを効率的にまとめます。

#### 最も重要な点

テクノロジーではなく、ユーザーの目的という観点からプロパティを表示します。プロパティとその用途を友人に説明する場面を想像してみてください。皆さんなら、どのように説明しますか。どのような言葉を使うでしょうか。その場面で使う言葉こそが、プロパティページで使用すべき言葉です。

詳細と例については、「[プロパティ ウィンドウのデザイン コンセプト](#)」を参照してください。

### 使用パターン

プロパティ ウィンドウにはいくつかの使用パターンがあります。

- **プロパティ シート**: 単一のオブジェクトのプロパティがモードレス ダイアログ ボックスに表示されます。
- **複数オブジェクトのプロパティ シート**: 複数のオブジェクトのプロパティがモードレス ダイアログ ボックスに表示されます。
- **有効な設定のプロパティ シート**: 単一のオブジェクトの有効なプロパティがモードレス ダイアログ ボックスに表示されます。
- **オプション ダイアログ ボックス**: アプリケーションのプロパティがモーダル ダイアログ ボックスに表示されます。
- **プロパティ インスペクター**: 現在の選択要素(単一のオブジェクトまたはオブジェクトのグループ)のプロパティがモードレス ウィンドウ ペインまたは非ドッキング ウィンドウに表示されます。

プロパティ インスペクター以外のプロパティ ウィンドウ パターンでは、いずれも "遅延型のコミット モデル" が使用されます。つまり、ユーザーが [OK] や [適用] をクリックした場合に限り、変更内容が有効になります。プロパティ インスペクターでは "即時型のコミット モデル"(ユーザーが変更を行うと、プロパティがすぐに変更される)が使用されるため、[OK]、[キャンセル]、および [適用] ボタンは不要です。

詳細と例については、「[プロパティ ウィンドウの使用パターン](#)」を参照してください。

### ガイドライン

#### プロパティ シート

- プロパティ シートは、ユーザーが次の操作を行ったときに表示します。
  - オブジェクトのプロパティ コマンドを選択したとき。
  - 入力フォーカスをオブジェクトに移し、Alt + Enter キーを押したとき。

#### 複数オブジェクトのプロパティ シート

- 選択されたすべてのオブジェクトに共通するプロパティを表示します。異なるプロパティ値が存在する場合、それらの値に関連付けられたコントロールの状態が混合された値を表示します(混在状態の値の使用については、各コントロールのガイドラインを参照してください)。
- 選択されているオブジェクトが複数の不連続オブジェクトである場合(ファイル フォルダーなど)、不連続オブジェクトについての複数オブジェクトのプロパティ シートではなく、グループ化された単一のオブジェクトのプロパティを表示します。

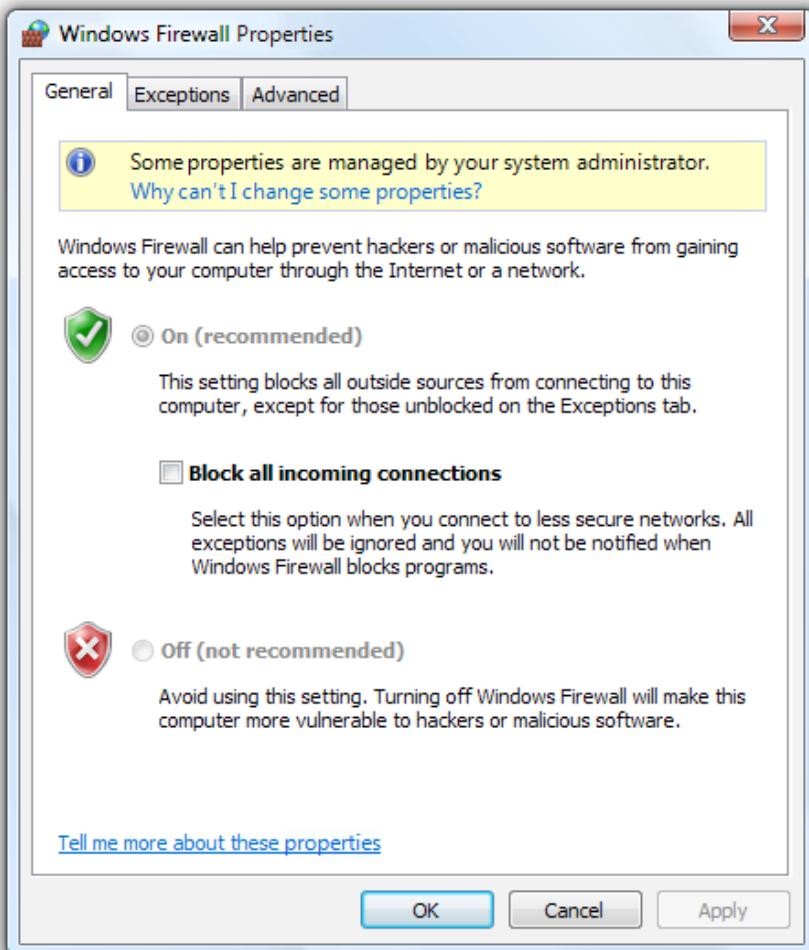
#### オプション ダイアログ ボックス

- オプションとカスタマイズを分けないようにします。つまり、[オプション] コマンドと [カスタマイズ] コマンドの両方を表示しないということです。こうした区別があると、ユーザーの混乱を招くおそれがあります。代わりに、オプションを通じてカスタマイズできるようにしてください。

#### プロパティ ページ

- 次のガイドラインに従ってページを配置します。
  - [全般] ページかそれに相当するページを最初のページにします。

- [詳細設定] ページかそれに相当するページを最後のページにします。
- 残りのページについては、次のガイドラインに従います。
  - 関連するページのグループにまとめます。
  - 使用される可能性が高いグループから順に配置します。
  - 各グループ内では、関連性または使用される可能性が高いページから順に配置します。
  - アルファベット順で表示しなくても済むように、ページの数は抑えてください。
- 各ページ上のすべてのプロパティをタスクベースの単一の具体的な用途に関連付けることで、ページの内容に一貫性を持たせます。
- プロパティ ウィンドウの用途が対象ユーザーにとってわかりにくい場合は、ページの上部に説明を付け加えます(ただし、スペースに余裕がある場合)。単一のタスクの専用ページの場合、そのタスクの実行方法を明確に示すテキストを追加します。文を使用し、末尾に句点を付けます。



この例では、Microsoft® Windows® ファイアウォールの用途が [全般] ページの上部で説明されています。

- コントロールの名前と位置を統一し、同様のコンテンツにはページ間で一貫性を持たせます。たとえば、複数のページに [名前] ボックスがある場合は、各ページの同じ位置に配置し、同じラベルを使用します。同様のコンテンツはページごとに異なる位置に配置しないようにしてください。
- アプリケーション全体で、同じプロパティは同じページに配置します。たとえば、オブジェクトの種類ごとに有効期限ポリシーの配置先を変えないようにします(一方は [全般] タブに配置し、もう一方は [詳細設定] タブに配置するなど)。
- ユーザーが最後に表示したページから作業を開始する可能性が高い場合は、そのページが既定で選択されるように、タブの状態を維持します。プロパティ ウィンドウごと、ユーザーごとに設定を維持し、該当しない場合は既定で最初のページが選択されるようにします。
- 1つのページの設定は、他のページの設定とは独立したものになります。相互に依存する設定は、1つのページ上にまとめます。あるページ上の設定が変更されても、他のページ上の設定が自動的に変更されないようにする必要があります。
  - 例外: 相互に依存する設定が2つの異なるプロパティ ウィンドウ上にある場合は、静的テキストラベルを使用して、両方のウィンドウでこの依存関係を説明します。
- プロパティ ページはスクロール型にしません。タブとスクロールバーはウィンドウの有効領域を拡大するためのものですが、どちらか一方で十分です。そこで、スクロールバーを使用する代わりに、プロパティ ページのサイズを拡大し、

ページを効率的に配置してください。

## 最初のページ

- オブジェクトのプロパティの場合、オブジェクトの名前を最初のページに表示します。
- オブジェクトに[アイコン](#)を関連付ける場合は(任意)、最初のページの左上隅に適切なアイコンを表示します。

## [全般] ページ

- [全般] ページはできるだけ使用しないようにします。[全般] ページは、必ずしも必要ではありません。[全般] ページは次の場合にのみ使用します。
  - プロパティが複数のタスクに適用され、ほとんどのユーザーに対して重要な場合。[全般] ページには、特殊なプロパティや詳細なプロパティを配置しません。ただし、[全般] ページのコマンド ボタンを通じてこれらにアクセスできるようにすることができます。
  - プロパティをより具体的なカテゴリに分けられない場合。カテゴリに分けられる場合は、その名前をページに使用します。

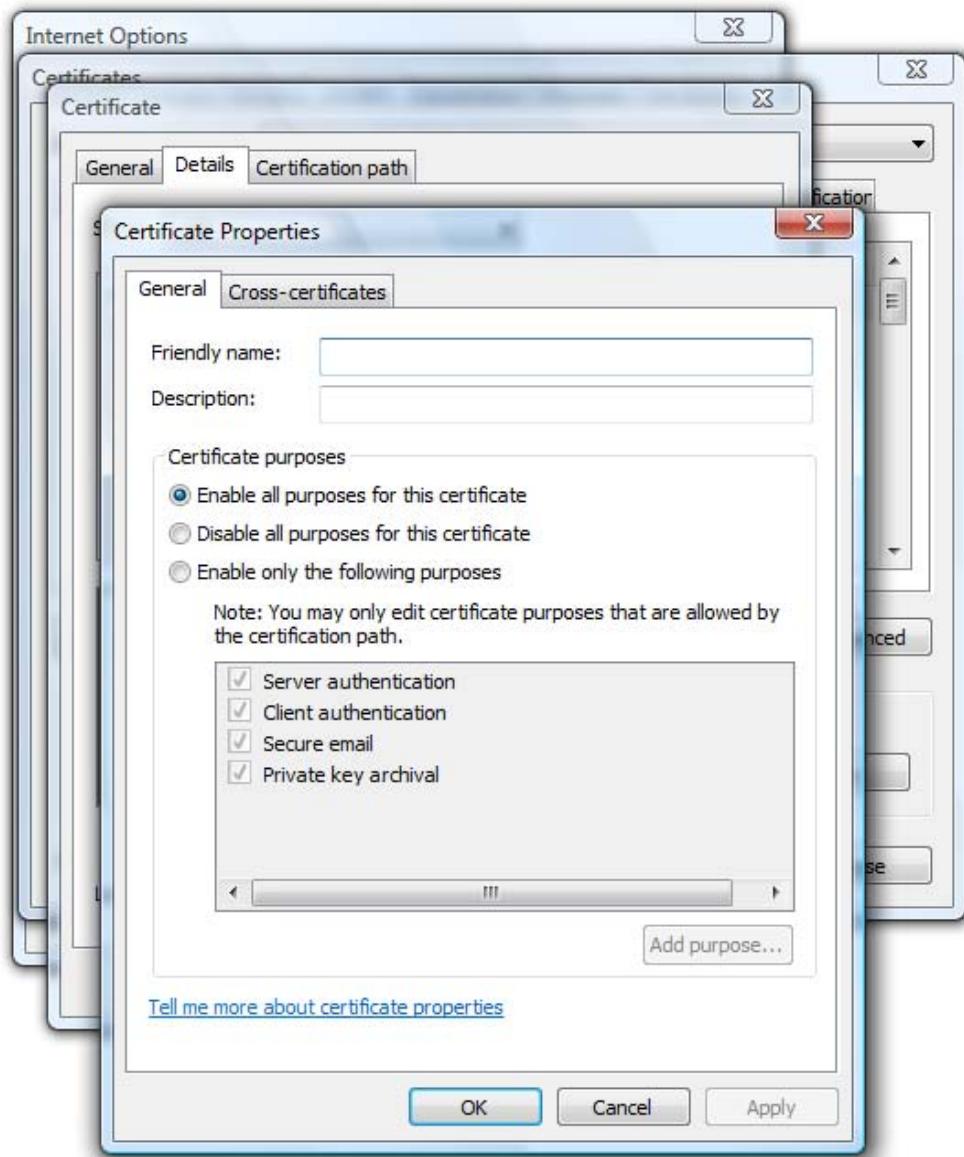
## [詳細設定] ページ

- [詳細設定] ページはできるだけ使用しないようにします。[詳細設定] ページは次の場合にのみ使用します。
  - プロパティが一般的ではないタスクに適用され、主に詳しい知識のあるユーザーにとって重要な場合。
  - プロパティをより具体的なカテゴリに分けられない場合。カテゴリに分けられる場合は、その名前をページに使用します。
- 技術的な側面だけを基に、詳細なプロパティであるかどうかを判断しないでください。たとえば、プリンターのホチキス止めオプションは高度なプリンター機能と考えられますが、すべてのユーザーにとって重要な機能なので、[詳細設定] ページに配置しないようにします。

## 子プロパティ ウィンドウ

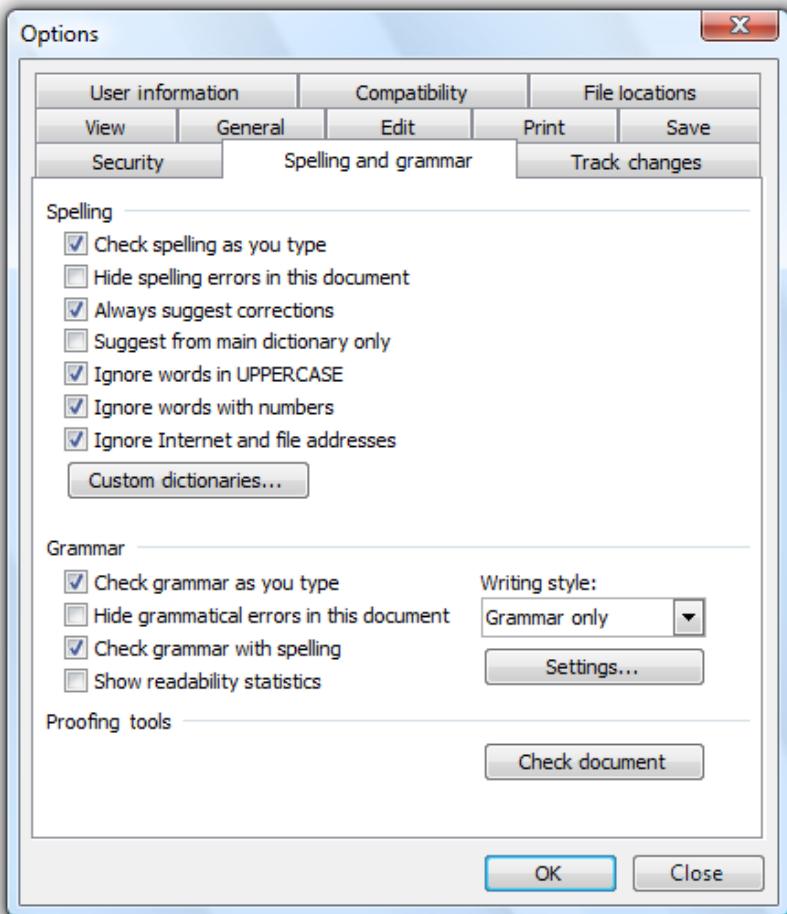
- プロパティ ウィンドウから、複数の子プロパティ ウィンドウを表示しないようにします。複数の子プロパティ ウィンドウを表示すると、[OK] ボタンや[キャンセル] ボタンの意味が理解しづらくなります。そこで、必要に応じて他の種類の補助ダイアログ ボックス(オブジェクトピッカーなど)を表示してください。

間違った例:



この例では、親オプション ダイアログ ボックスに 3 階層の子プロパティ ウィンドウがあります。この結果、[OK] と [キャンセル] の意味が不明瞭になっています。

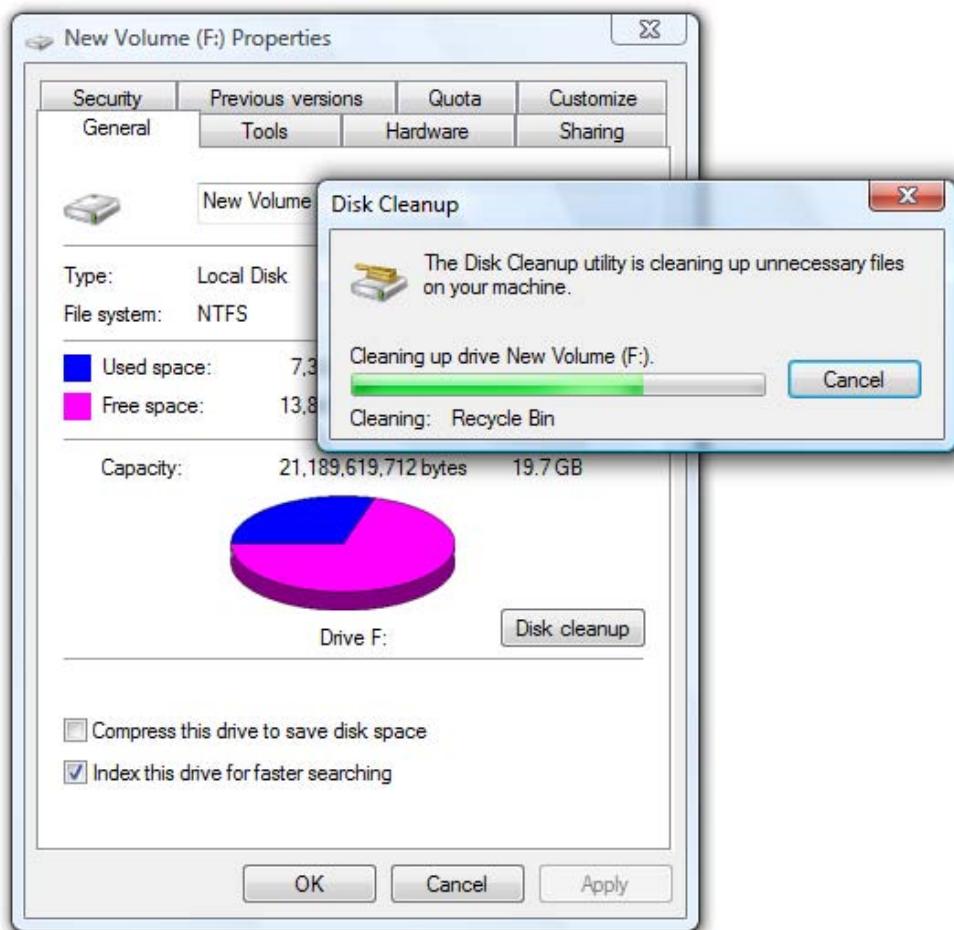
- ・遅延型のコミット モデルを使用するプロパティ ウィンドウでは、ユーザーが子プロパティ ウィンドウ上で行った変更を、オーナー ウィンドウ上で [キャンセル] をクリックして取り消すことができるようになります。
- ・子プロパティ ウィンドウで即時型のコミット モデルを採用する場合は、オーナー ウィンドウ上の [キャンセル] ボタンを [閉じる] ボタンに変更することで、変更が既にコミットされていることがわかるようになります。ユーザーが [適用] をクリックしたら、ボタンを [キャンセル] に戻します。



この例では、ユーザー辞書と文章構成の設定に対する変更を取り消すことはできません。[キャンセル] を [閉じる] に変更することで、それをユーザーに示すことができます。

#### 他の子ウィンドウ

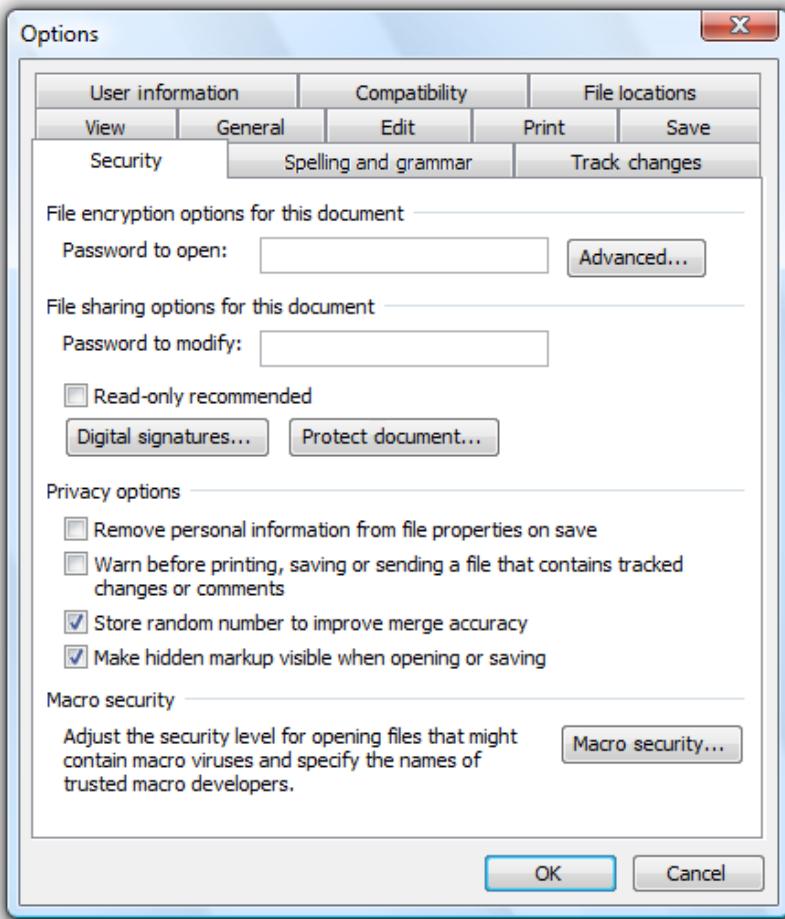
- 子ウィンドウが補助的なタスクの実行に使用される場合は、[キャンセル] ボタンの名前を変更しません。前に示したガイドラインが適用されるのは、補助的なタスクの実行に使用されるダイアログ ボックスではなく、子プロパティ ウィンドウです。



この例では、ディスクのクリーンアップは補助的なタスクなので、前のガイドラインは適用されません。たとえば、オーナー ウィンドウ上の [キャンセル] ボタンを [閉じる] に変更しないでください。

- 子ウィンドウが補助的なタスクの実行に使用される場合、コマンド ボタンがクリックされたときに親プロパティ ウィンドウを閉じないようにします。そうしないと、ユーザーの混乱を招きます。ユーザーは、そのコマンドを実行するためだけにプロパティ ウィンドウを開くわけではありません。

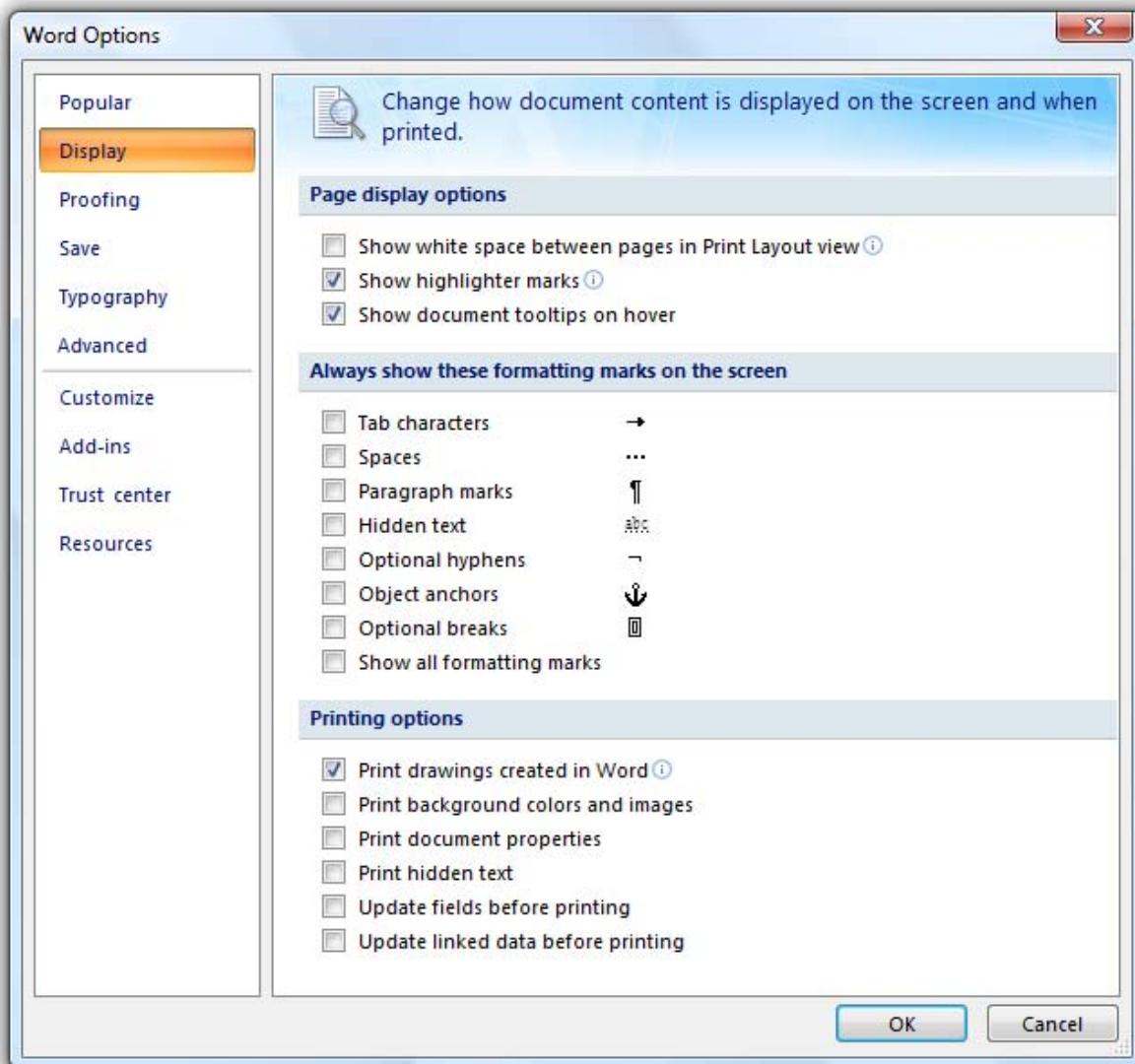
間違った例:



この例では、[文書の保護] をクリックすると [オプション] ダイアログ ボックスが閉じますが、これは不適切な動作です。

## タブ

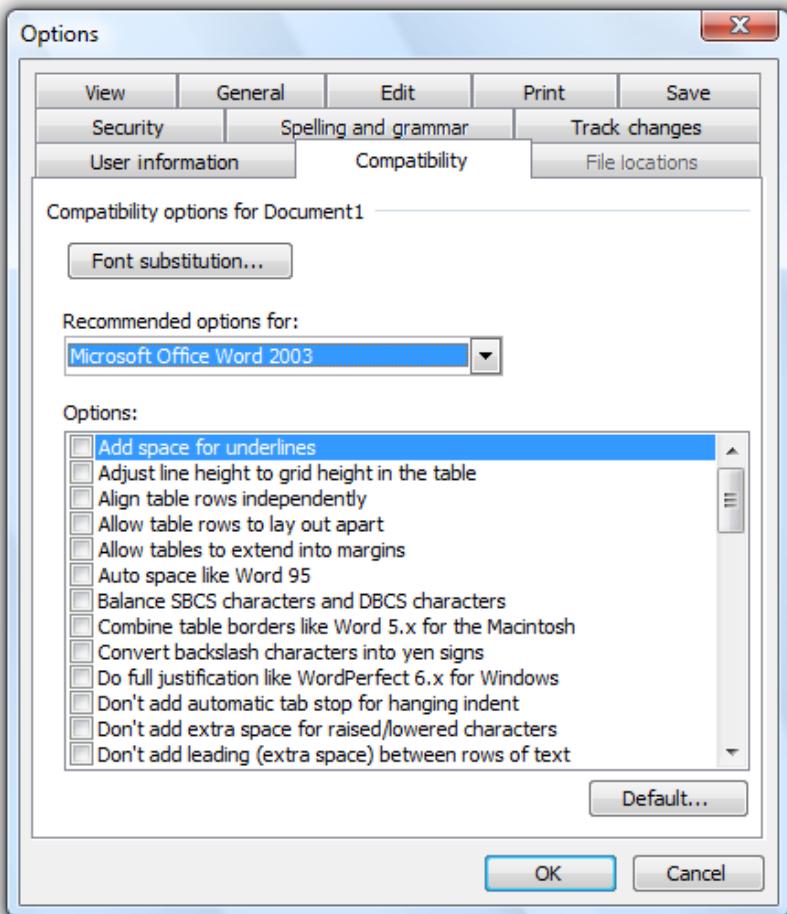
- 簡潔なタブ ラベルを使用します。ページのコンテンツを 1 ~ 2 語で明確に説明します。長いラベルを使うと、ラベルがローカライズされる場合は特に、無駄に画面領域を使用することになります。
- 限定的で意味のあるタブ ラベルを使用します。"全般"、"詳細設定"、"設定" など、どのタブにも当てはまるような汎用的なタブ ラベルの使用は避けます。
- 次のような場合は、水平タブを使用します。
  - プロパティ ウィンドウのタブが、サードパーティの拡張タブも含め 7 個以下の場合。
  - UI がローカライズされても、すべてのタブが 1 行に収まる場合。
  - 水平タブを、アプリケーションの他のプロパティ ウィンドウで使用する場合。
- 次のような場合は、垂直タブを使用します。
  - プロパティ ウィンドウのタブが、サードパーティの拡張タブも含め 8 個以上の場合。
  - 水平タブを使用すると 2 行以上になる場合。
  - 垂直タブを、アプリケーションの他のプロパティ ウィンドウで使用する場合。



この例では、垂直タブによって 8 個以上のタブが表示されています。

- プロパティインスペクターでは、スペースを節約するために、タブの代わりにドロップダウンリストを使用することを検討してください (特に、ユーザーが現在のタブをほとんど変更しない場合)。
- 現在のコンテキストに該当しない、ユーザーが想定しないタブは削除します。不要なタブを削除することで、簡潔な UI になります、見落としがなくなります。

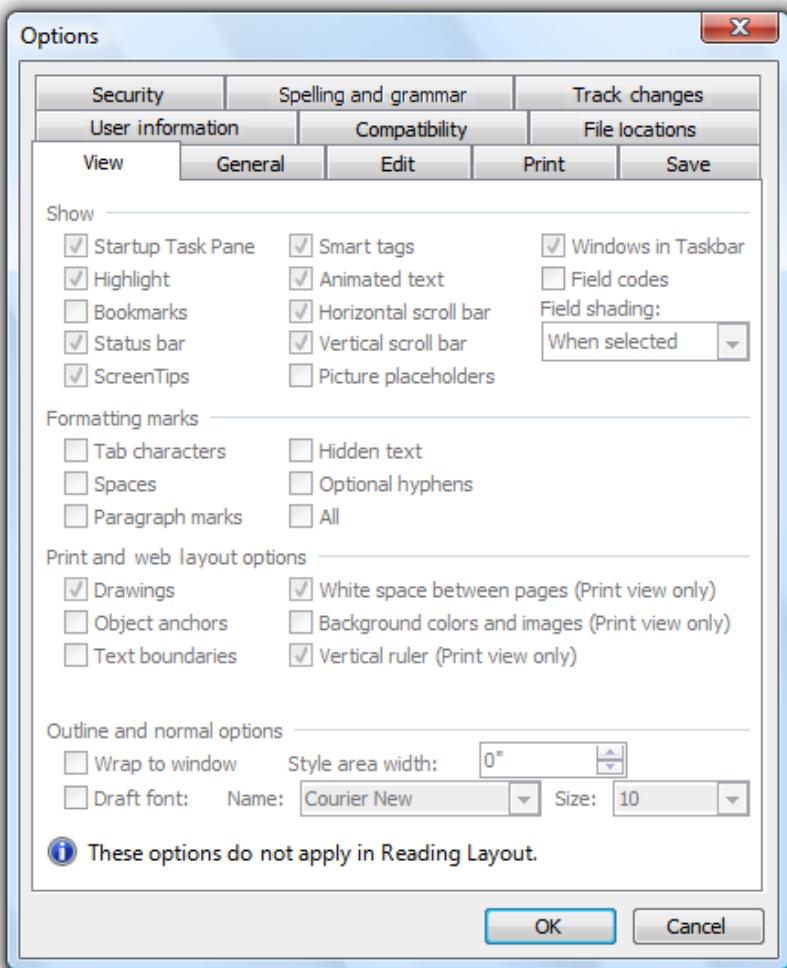
間違った例:



この例では、Microsoft Word 2003 が電子メールエディターとして使用されている場合に、[ファイルの場所] タブが無効にされていますが、これは間違います。ユーザーはこのコンテキストにおいて、ファイルの場所を確認したり変更する操作は想定していないため、ページを削除する必要があります。

- 現在のコンテキストに該当しなくても、ユーザーが次のことを想定している可能性がある場合は、タブを無効にしません。
  - タブが表示される。
  - ページ上のコントロールを無効にできる。
  - コントロールが無効になっている理由を説明するテキストが表示される。

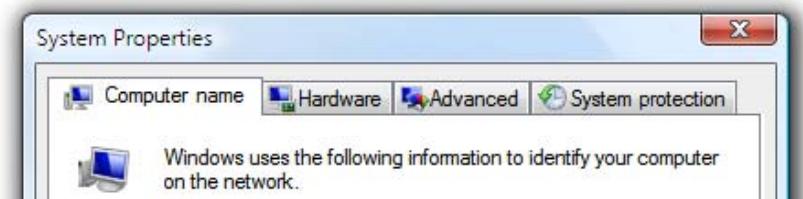
このような操作を行おうとするときにタブが無効になっていると、ユーザーにはその理由がわからず、また調べることもできず、特定のプロパティを探すときに他のすべてのタブを確認しなくてはならなくなります。



Word 2003 のこの例では、閲覧レイアウトでは [表示] のオプションはいずれも適用されませんが、ユーザーはタブラベルを見て何らかのオプションが適用されると考える可能性があります。このため、ページは表示され、オプションは無効になっています。

- タブの変更に対して効果を割り当てるることはしません。現在のタブを変更しても、副次的な影響が発生したり、設定が適用されたり、エラーメッセージが発生したりしないようにします。
- タブを入れ子にしたり、水平タブと垂直タブを組み合わせることはしません。代わりにタブの数を減らすか、垂直タブのみを使用するか、またはドロップダウンリストなど別のコントロールを使用します。
- プロパティ ウィンドウにタブが 1 つしかなく、拡張不可能である場合は、タブは使用しません。代わりに、[OK] ボタン、[キャンセル] ボタン、そして場合によっては [適用] ボタンを持つ通常のダイアログ ボックスを使用します。拡張可能なプロパティ ウィンドウ (サードパーティによる拡張が可能なウィンドウ) の場合は、必ずタブを使用する必要があります。
- タブ上にアイコンは配置しません。アイコンは、通常、煩雑になりがちで、画面領域を消費し、多くの場合は理解の向上につながりません。アイコンは、標準記号のように理解の補助になるものだけを使用します。

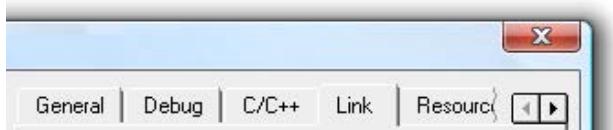
間違った例:



この例では、グラフィックによって見た目が乱雑になり、ユーザーの理解の助けになっていません。

- タブのグラフィックに製品ロゴは使用しません。タブはブランド化のためのものではありません。
- 水平タブはスクロール型にしません。水平スクロールは見つけにくいものです。垂直タブはスクロール型にしてもかまいません。

間違った例:



この例では、水平タブがスクロール型になっています。

#### コマンド ボタン

- すべてのプロパティ ページに適用されるコマンド ボタンは、プロパティ ウィンドウの下部に配置します。ボタンは右揃えにし、左から [OK]、[キャンセル]、[適用] の順に配置します。
- 個別のプロパティ ページにのみ適用されるコマンド ボタンは、そのプロパティ ページに直接配置します。

#### コミット ボタン

##### [OK] ボタン

- 親プロパティ ウィンドウの場合、[OK] ボタンのクリックによって行われる処理は、保留中の変更(ウィンドウが開かれた後や、前回 [適用] ボタンがクリックされた後に行われた変更)を適用し、ウィンドウを閉じるという処理です。
- 子プロパティ ウィンドウの場合、[OK] ボタンのクリックによって行われる処理は、変更を保留状態にし、ウィンドウを閉じて、オーナー ウィンドウの変更が適用される際に変更を適用するという処理です。
- [OK] ボタンの名前は変更しないでください。他のダイアログ ボックスとは異なり、プロパティ ウィンドウは特定のタスクを実行するために使用されるわけではありません。[OK] ボタンの名前を([印刷]などに)変更すると効果的なのは、プロパティ ウィンドウ以外のウィンドウです。
- アクセス キーは割り当てません。

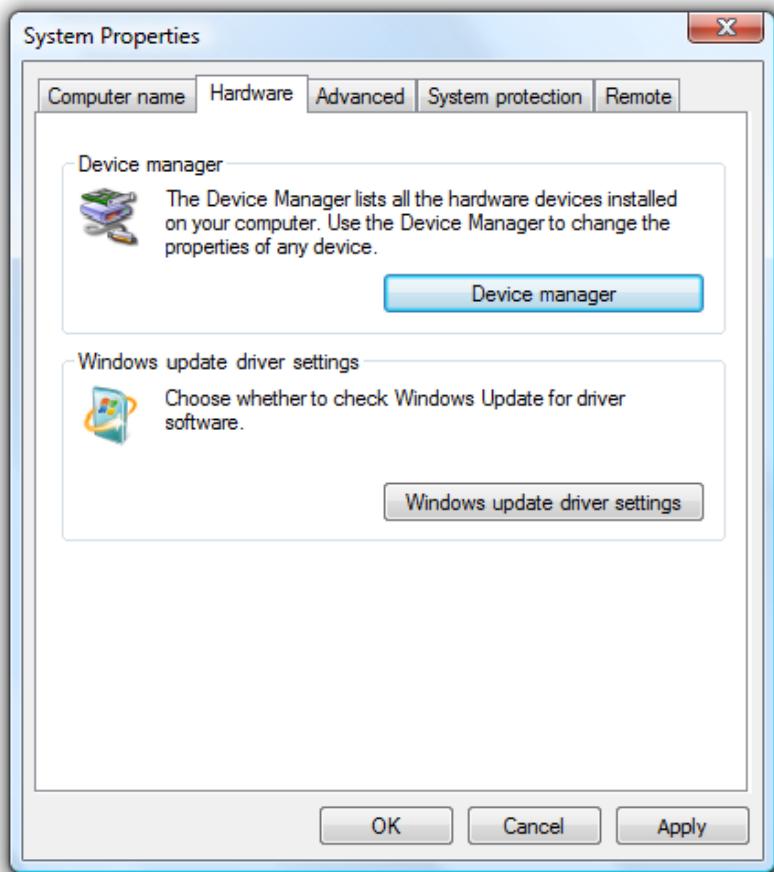
##### [キャンセル] ボタン

- [キャンセル] ボタンのクリックによって行われる処理は、保留中の変更(ウィンドウが開かれた後や、前回 [適用] ボタンがクリックされた後に行われた変更)をすべて破棄し、ウィンドウを閉じるという処理です。
- 保留中のすべての変更を破棄できない場合は、[キャンセル] ボタンを [閉じる] ボタンに変更します。[キャンセル] がクリックされたら、保留中のすべての変更を破棄する必要があります。
- 子プロパティ ウィンドウで即時型のコミットが必要な場合は、オーナー ウィンドウ上の [キャンセル] ボタンを [閉じる] ボタンに変更し、変更が既にコミットされていることを示します。
- アクセス キーは割り当てません。

##### [適用] ボタン

- 親プロパティ シートの場合、[適用] ボタンのクリックによって行われる処理は、保留中の変更(ウィンドウが開かれた後や、前回 [適用] ボタンがクリックされた後に行われた変更)を適用するが、ウィンドウは開いたままにするという処理です。こうすることで、ユーザーがプロパティ シートを閉じる前に変更内容を評価できるようになります。
- 子プロパティ シートでは使用しません。子プロパティ シートに [適用] ボタンを設けると、親プロパティ シート上のコミット ボタンの意味が理解しづらくなります。
- [適用] ボタンは、ユーザーが意味のある方法で効果を評価できる設定がプロパティ シートに(1つ以上)ある場合にのみ配置します。通常、[適用] ボタンは、設定によって目に見える変化が生じる場合に使用します。ユーザーが変更を適用し、その変更を評価して、その評価を基にさらに変更を行えるようにしてください。該当しない場合は、[適用] ボタンを無効にするのではなく削除します。

間違った例:



この例では、どのシステム プロパティにも視覚的な効果がなく、[適用] ボタンに意味がないので、削除する必要があります。

- ユーザーが適用する可能性のある設定はすべて親ページに配置します。混乱を招くおそれがあるので、子プロパティ シートには [適用] ボタンを配置しないでください。
- [適用] ボタンはオプション ダイアログ ボックスでは使用しません。プロパティ シートでのみ使用します。
- 保留中の変更がある場合にのみ、[適用] ボタンを有効にします。保留中の変更がない場合は無効にします。
- アクセス キーとして "A" を割り当てます。

#### [閉じる] ボタン

- 保留中のすべての変更を破棄できない場合は、[キャンセル] ボタンを [閉じる] ボタンに変更します。[キャンセル] がクリックされたら、保留中のすべての変更を破棄する必要があります。
- 変更を破棄するかどうかをユーザーに確認しません。
  - 例外: 構成に多大な労力を要する設定がプロパティ ウィンドウにあり、ユーザーがそれを変更している場合は、ユーザーがタイトルバーの [閉じる] ボタンをクリックしたときに確認を表示してもかまいません。タイトルバー上の [閉じる] ボタンに [OK] ボタンと同じ効果があると誤解しているユーザーもいるためです。
- 例外的に確認メッセージを表示する場合は、タイトルバーの [閉じる] ボタンに [キャンセル] または [閉じる] ボタンと同じ効果があることを確認してください。

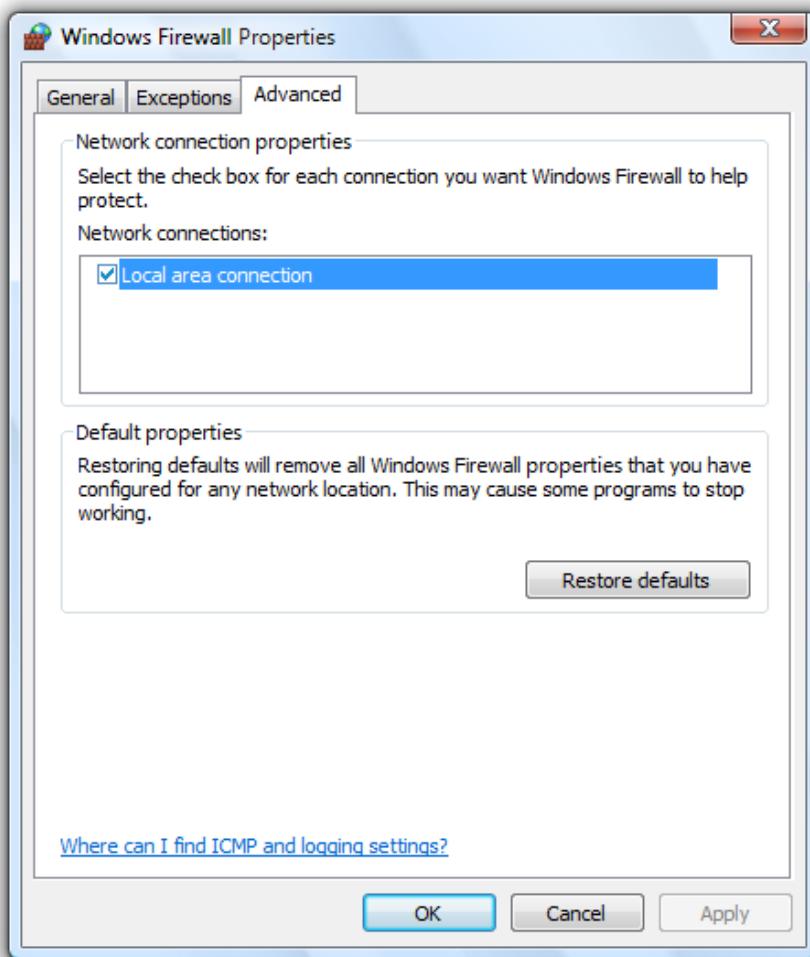
#### ページ コンテンツ

- プロパティの必要性を確認します。デザイン上の難しい判断を敬遠して、ページに不要なプロパティを詰め込まないようにしてください。
- テクノロジーではなく、ユーザーの目的という観点からプロパティを表示します。プロパティは特定のテクノロジーを構成するためのものですが、だからといって、そのテクノロジーの観点からプロパティを表示する必要はありません。
  - テクノロジーの観点から設定を表示する必要がある場合(ユーザーがテクノロジーの名前を把握している場合など)は、ユーザーがその設定からどのような利点を得られるかを簡潔に説明するテキストを追加してください。
- 適切なレベルのプロパティを表示します。個々の低水準の設定をプロパティ ページで表示する必要はありません。対象ユーザーが必要とするレベルのプロパティを表示してください。
- 特定のタスク向けのプロパティ ページをデザインします。ユーザーが実行するタスクを特定し、それらのタスクを実行するための明確なプロセスが用意されていることを確認します。
- プロパティ ページを効率的にまとめます。そのためには、タブの数を減らし、論理的なグループと関連性を基にページ

に残すタブを決め、ページの表示を簡素化します。

詳細については、「[プロパティ ウィンドウのデザイン コンセプト](#)」を参照してください。

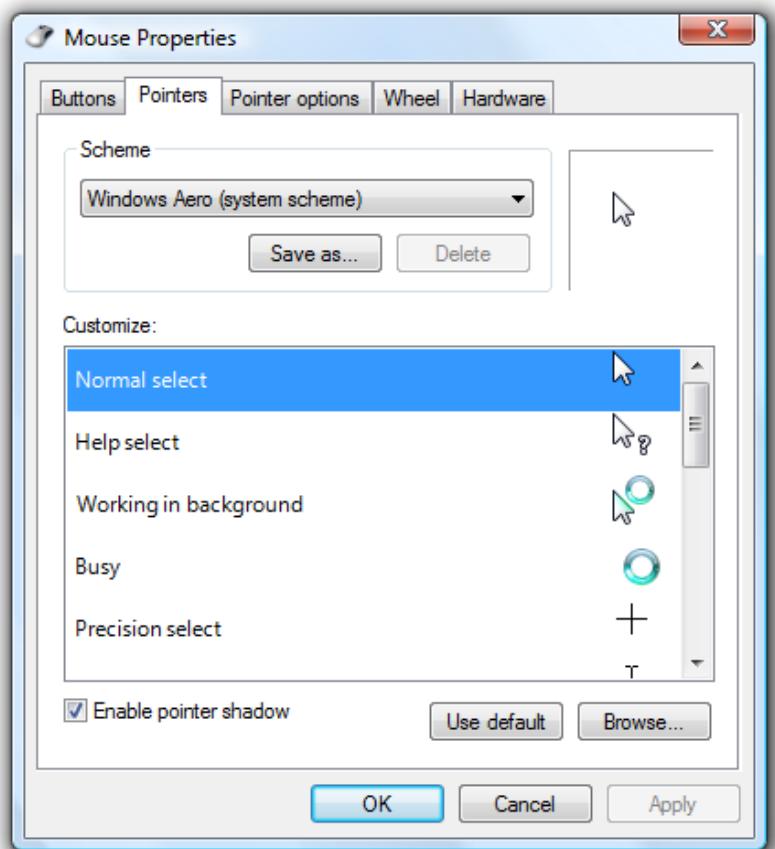
- 強く推奨されるオプションに対しては、ラベルに "(推奨)" を追加することを検討します。
- 次の場合には、プロパティ ページまたはプロパティ ウィンドウ全体に適用される [既定値に戻す] コマンド ボタンを配置します。
  - ユーザーが設定が複雑でわかりづらいと感じる可能性がある場合。
  - 設定を間違えると正しく機能しない可能性があるが、既定の設定に戻すと機能を復元できる場合。
  - オブジェクトが正しく構成されていない場合で、ユーザーにとって最初から構成をやり直す方が簡単な場合。



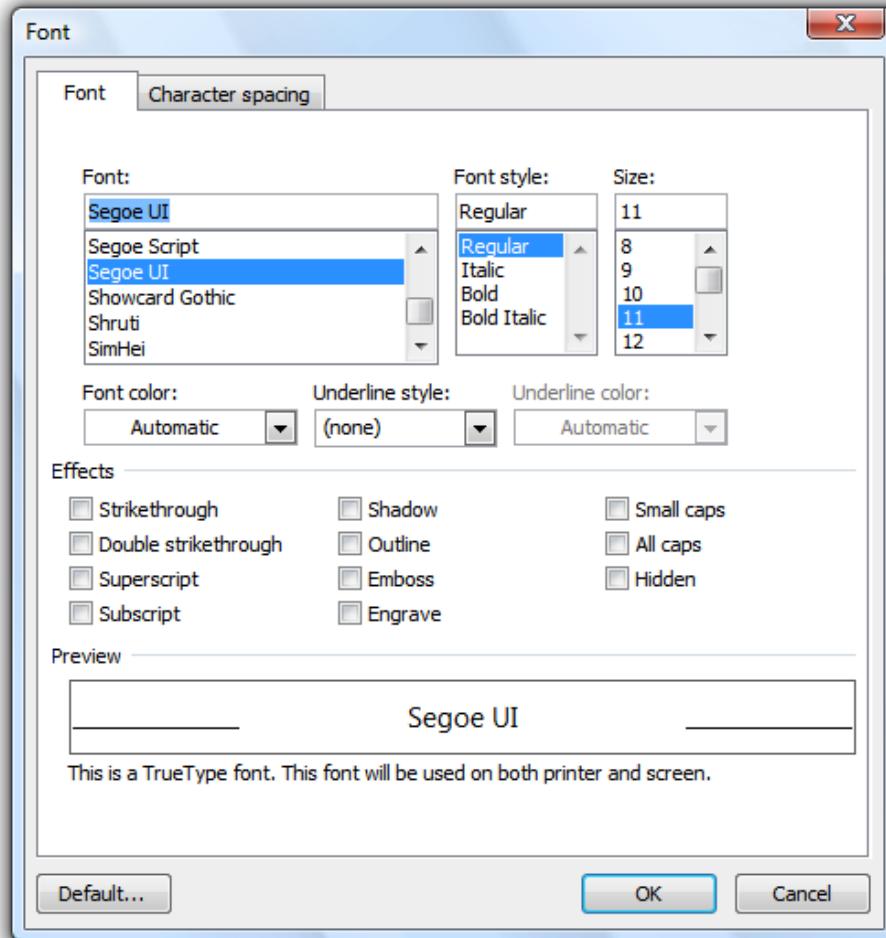
この例では、Windows ファイアウォールの設定が複雑で、正しく機能しなくなる可能性があります。何らかの問題があるときは、多くの場合、[既定値に戻す] をクリックして最初からやり直した方が簡単です。

[既定値に戻す] コマンドの効果がわかりにくい場合や、設定が複雑な場合は、確認メッセージを表示します。確認が必要であることを示すには、省略記号を使用します。

- 必要に応じて設定の結果のプレビューを表示します。



この例では、ページにポインター デザインのプレビューが表示されています。[適用] のクリックによってプレビューを表示させることもできますが、ページ上にプレビューを表示した方がユーザーにとって便利です。



この例では、[プレビュー] ボックスにフォント設定の結果が表示されています。この例は、グラフィック

ク要素以外の設定をプレビューできることを示しています。

## ヘルプ

- ユーザー アシスタンスを提供する場合は、次の方法を検討してください(望ましい順に列挙してあります)。
  - 対話型コントロールに内容を表すラベルを付けます。ユーザーは、他のどのテキストよりも、対話型コントロール上のラベルを読む可能性が高い傾向にあります。
  - 静的テキストラベルを使用し、コンテキストに即した説明を追加します。
  - 対応するヘルプトピックへの具体的な[リンク](#)を提供します。
- ヘルプリンクを各ページの下部に配置します。ヘルプトピックを持つ設定グループがページ(通常はグループボックス内)に複数ある場合は、ヘルプリンクをグループの下部に配置します。
- 一般的なヘルプトピックリンク、あいまいなヘルプトピックリンク、または汎用的なヘルプボタンは使用しません。ユーザーは、汎用的なヘルプを無視する傾向にあります。

詳細と例については、「[ヘルプ](#)」を参照してください。

## 標準ユーザーおよび保護された管理者

変更するうえで管理者特権が必要な設定は、数多くあります。処理に管理者特権が必要な場合、Windows Vista®以降では、[標準ユーザーと保護された管理者](#)は各自の特権を明示的に昇格する必要があります。これにより、危険なコードが管理者特権で実行されるのを防ぐことができます。

詳細と例については、「[ユーザー アカウント制御](#)」を参照してください。

## 既定値

- プロパティ ウィンドウ内の設定は、アプリケーション、オブジェクト、またはオブジェクトコレクションの現在の状態を反映している必要があります。そうしないと、誤解を招き、好ましくない結果が生じる可能性があります。たとえば、設定が現在の状態ではなく推奨事項を反映している場合、ユーザーは変更不要と見なし、変更作業を行わずに[キャンセル]をクリックする可能性があります。
- 最も安全(データの消失やシステムアクセスが失われることを防ぐため)で、最もセキュリティの高い初期状態を選択します。ほとんどのユーザーが設定を変更しないことを前提としてください。
- 安全性とセキュリティを判断材料として考える必要がない場合は、最もよく使用される初期状態か、最も便利な初期状態を選択します。

## テキスト

### コマンド

- プログラムオプションを表示するコマンドには、"オプション"という名前を付けます。
- オブジェクトのプロパティ ウィンドウを表示するコマンドには、"プロパティ"という名前を付けます。
- よく使用されるプログラムのカスタマイズ設定の概要を表示するコマンドには、"[個人用設定](#)"という名前を付けます。
- "設定"や"基本設定"は使用しません。
- これらのコマンドには、[省略記号](#)は使用しません。

### プロパティ シートのタイトル

- 単一のオブジェクトの場合は、"[オブジェクト名]のプロパティ"を使用します。
  - オブジェクトに名前がない場合は、オブジェクトの種類を名前に使用します(ユーザー アカウント プロパティなど)。
- 複数のオブジェクトの場合は、"[最初のオブジェクト名]、... のプロパティ"を使用します。
  - オブジェクトに名前がない場合は、オブジェクトの種類を名前に使用します(ユーザー アカウント プロパティなど)。
  - 種類の異なる複数のオブジェクトの場合は、"選択項目のプロパティ"を使用します。
- [タイトルスタイルの大文字化](#)を使用します。
- 末尾に句読点は付けません。
- "[オブジェクト名]-プロパティ"のようにハイフンは使用しません。

### プロパティ インスペクターのタイトル

- "プロパティ"を使用します。
- タイトルスタイルの大文字化を使用します。
- 末尾に句読点は付けません。

## オプション ダイアログ ボックスのタイトル

- "オプション" を使用します。
- タイトルスタイルの大文字化を使用します。
- 末尾に句読点は付けません。

## プロパティ ページのタブ名

- 簡潔なタブラベルを使用します。ページのコンテンツを 1 ~ 2 語で明確に説明します。長いタブ名を使うと、タブ名がローカライズされる場合は特に、無駄に画面領域を使用することになります。
- 限定的で意味のあるタブラベルを使用します。"全般"、"詳細設定"、"設定" など、どのタブにも当てはまるような汎用的なタブラベルの使用は避けます。
- ラベルには 1 ~ 2 語の語句を使用し、末尾に句読点は付けません。
- **センテンススタイルの大文字化**を使用します。
- 固有の **アクセスキー**は割り当てません。

## プロパティ ページのテキスト

- テキストが、大きなブロックにならないようにします。
- ローカライズされる場合は、30% の余白を残しておきます。
- テキストの語句をプロパティ ウィンドウ上のコマンドとしては使用しません。ユーザーが単に設定を確認するだけの場合も考えられるので、設定の変更を求めるメッセージを表示する必要はありません。
- センテンススタイルの大文字化を使用し、末尾に句読点を付けます。

## ドキュメント

### プロパティ ウィンドウを示す場合:

- プログラミングおよびその他の技術文書では、プロパティシートとオプション ダイアログ ボックスは "プロパティシート" と記述します。それ以外のドキュメント(特にユーザー向けドキュメント)では、"ダイアログ ボックス" を使用します。
- 大文字と小文字の区別を含め、タイトル テキストを正確に引用します。
- ユーザー操作を説明する場合は、"開く" と "閉じる" を使用します。
- タイトルを半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

### プロパティ ページを示す場合:

- プログラミングおよびその他の技術文書では、"プロパティ ページ" と記述します。それ以外のドキュメント(特にユーザー向けドキュメント)では、"タブ" を使用します。
- 大文字と小文字の区別を含め、タイトル テキストを正確に引用します。
- ユーザー操作を説明する場合は、タブのクリックを示す際に "クリック" を使用します。
- 名前を半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

# プロパティ ウィンドウのデザイン コンセプト

[プロパティ ウィンドウ](#)  
[プロパティ ウィンドウの使用パターン](#)

注: このセクションの内容は、主にプロパティ シートとオプション ダイアログ ボックス向けのものですが、プロパティ インスペクターにもある程度当てはまります。

## プロパティ ウィンドウを使いやくする

開発者は、多くの場合、プログラムの 80% の要素を明確に規定し、残りの 20% についてはユーザーがプロパティ ウィンドウを通じて操作できるようにすることで、プログラムのユーザーインターフェイス (UI) を簡素化します。これは場合によっては優れたアプローチですが、慎重に行わないと、使いにくいプロパティ ウィンドウができあがる可能性があります。

プロパティ ウィンドウは、テクノロジーに基づく低水準の設定を中途半端に寄せ集めた "ごみ置き場" となりがちです。こうしたプロパティはタブにまとめられてはいても、特定のタスクやユーザーを考慮せずにデザインされるケースが少なくありません。その結果、ユーザーがプロパティ ウィンドウで何らかのタスクを行うときに、何をすべきかわからなくなることがよくあります。

高い実用性と操作性を兼ね備えたプロパティ ウィンドウをデザインするには、次の手順に従ってください。

- [プロパティの必要性を確認します。](#)
- [テクノロジーではなく、ユーザーの目的という観点からプロパティを表示します。](#)
- [適切なレベルのプロパティを表示します。](#)
- [特定のタスク向けのページをデザインします。](#)
- [標準ユーザーと保護された管理者向けのページをデザインします。](#)
- [プロパティ ページを効率的にまとめます。](#)

## プロパティの必要性を確認する

開発者は強力で柔軟性に富んだプログラムを作成したいと考えるものですが、選択肢が多すぎると、ユーザーの手に負えなくなるおそれがあります。プロパティ ウィンドウ内のプロパティをすべて見直し、次の作業を行ってください。

- [プロパティを必要とするタスクを特定する。](#)
- [プロパティの対象ユーザーを特定する。](#)
- [対象ユーザーが実際にプロパティを使用する可能性を考える。](#)

使用される可能性が低い場合 (使用するとベスト プラクティスに準拠できなくなるなど)、プロパティを削除することを検討します。デザイン上の難しい判断を敬遠して、プロパティ ページに不要なプロパティを詰め込まないようにしてください。使用される可能性があるものではなく、使用される可能性が高いものを提供することが重要です。

## テクノロジーではなく、ユーザーの目的という観点からプロパティを表示する

プロパティは特定のテクノロジーを構成するためのものですが、そのテクノロジーの観点から表示する必要はありません。次の設定を比較してみてください。

テクノロジーに基づく表示:

- [Enable internet connection sharing host](#)
- [Manual duplex](#)

これらの例では、プロパティはテクノロジーの観点から表示されています。

目的に基づく表示:

Allow other network users to connect through this computer's internet connection

Print on both sides of paper

これらの例では、同じプロパティがユーザーの目的という観点から表示されています。

プロパティページのテキストを評価する簡単な方法は、プロパティとその用途を友人に説明する場面を想像してみることです。皆さんなら、どのように説明しますか。どのような言葉を使うでしょうか。おそらく、わかりにくいテクノロジー(手動両面印刷など)ではなく、ユーザーの目的(用紙の両面への印刷など)という観点から、平易な言葉で設定を説明すると思われます。その場面で使う言葉こそが、プロパティページで使用すべき言葉です。

テクノロジーの観点からプロパティを表示する必要がある場合(対象ユーザーがテクノロジーの名前を把握している場合など)は、プロパティのユーザーに対する実際の利点について簡潔に説明するテキストも付け加えてください。

#### 最も重要な点

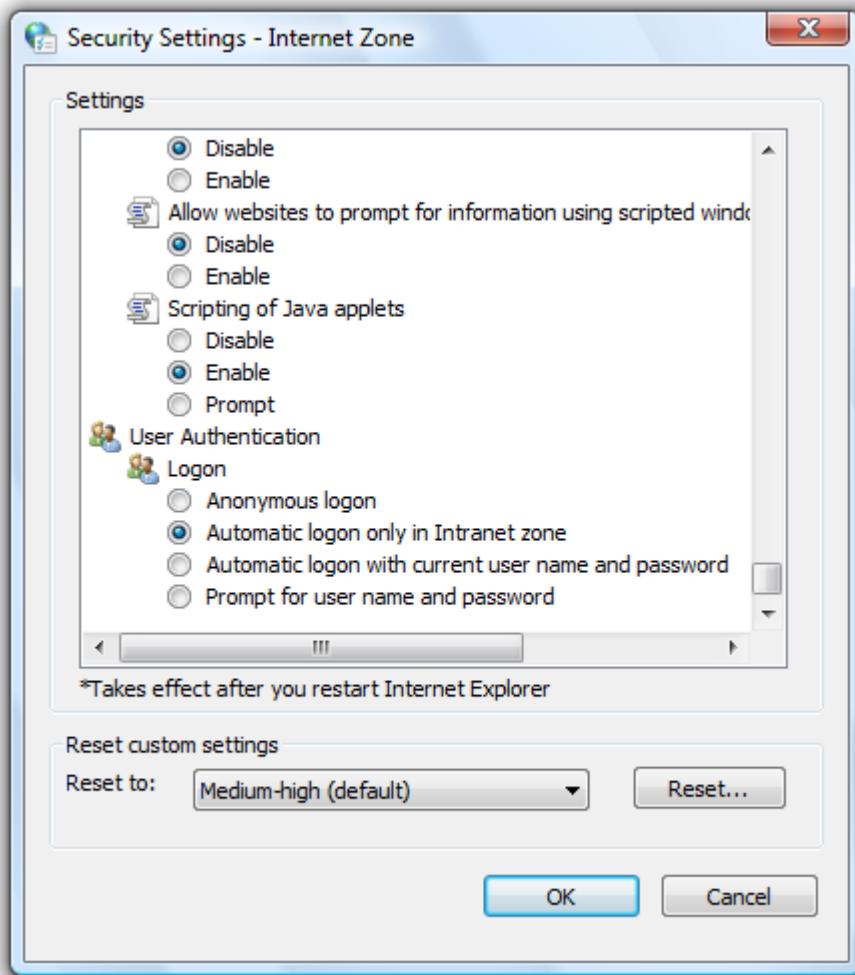
テクノロジーではなく、ユーザーの目的という観点からプロパティを表示します。プロパティとその用途を対象ユーザーに直接説明する場面を想像してみてください。皆さんなら、どのように説明しますか。どのような言葉を使うでしょうか。その場面で使う言葉こそが、プロパティページで使用すべき言葉です。

#### 適切なレベルのプロパティを表示する

個々の低水準の設定は、ユーザーの目的とは関係がない場合があります。個々の低水準の設定をプロパティページで表示する必要はありません。対象ユーザーが必要とするレベルのプロパティを表示してください。

次のサンプルは、Windows® Internet Explorer® のセキュリティ設定です。ここに示されている、テクノロジーに基づく低水準の設定について考えてみます。

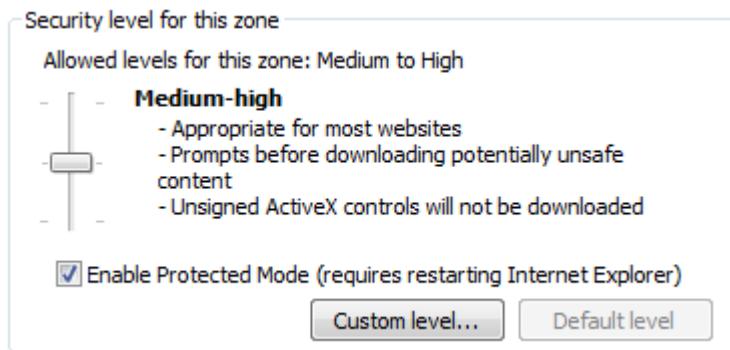
低水準の設定の例:



この例に示されているのは、Windows Internet Explorer の詳細なセキュリティ設定のごく一部(約 15%)です。

どの設定を有効にし、どの設定を無効にする必要があるのかを正確に把握しているユーザーはほとんどいないでしょう。とはいえ、ほとんどのユーザーに、大部分の機能を有効にしたままインターネットを安全に閲覧したいといった大まかな目的はあるはずです。したがって、Windows Internet Explorerには、目的に基づくレベルという観点から表示されるセキュリティ設定が用意されています。

より良い例:



この例では、一連の設定がユーザーの目的という観点から表示されています。

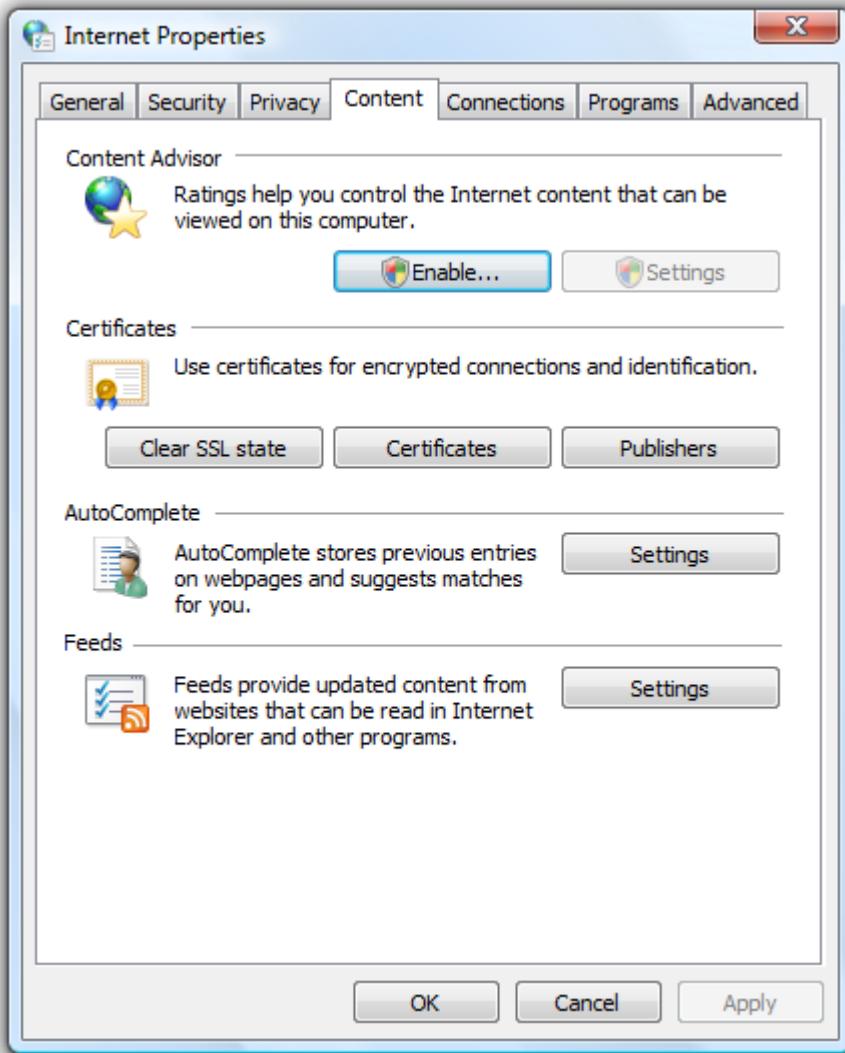
個々の設定を変更する必要がある場合、ユーザーは、既定のセキュリティ レベルから最適なレベルを選択したうえで、カスタム レベルを作成できます。

### 特定のタスク向けのページをデザインする

ページをデザインおよび評価する際には、ユーザーが実行する一般的なタスクを特定し、それらのタスクを実行するための明確なプロセスが用意されていることを確認してください。ユーザーは、通常、プロパティ ウィンドウで次の種類のタスクを実行します。

- ・オブジェクトの現在の設定と属性を表示する。
- ・オブジェクトの設定を変更する。
- ・オブジェクトの設定と属性に関係するタスクを実行する。
- ・オブジェクトの設定のトラブルシューティングを行い、予想どおりに機能しない理由を突き止める。
- ・オブジェクトの設定を既知の正常な状態(既定の設定など)に復元またはロールバックする。

プロパティ ウィンドウが特定のタスクをサポートしているかどうかを判断するには、各プロパティ グループの用途を 1 文で書き表し、サポートされるタスクがその用途の説明にどの程度対応しているかを確認します。こうした説明は、必ずしもプロパティ ウィンドウに追加する必要はありませんが、用途がはっきりしない場合は追加することをお勧めします。



この Windows Internet Explorer の例では、[コンテンツ] タブの各設定グループに説明が 1 文で追加されています。

### 標準ユーザーと保護された管理者向けのページをデザインする

変更するうえで管理者特権が必要な設定は、数多くあります。処理に管理者特権が必要な場合、Windows Vista 以降では、[標準ユーザーと保護された管理者](#)は各自の特権を明示的に昇格する必要があります。これにより、危険なコードが管理者特権で実行されるのを防ぐことができます。

詳細と例については、「[ユーザー アカウント制御](#)」を参照してください。

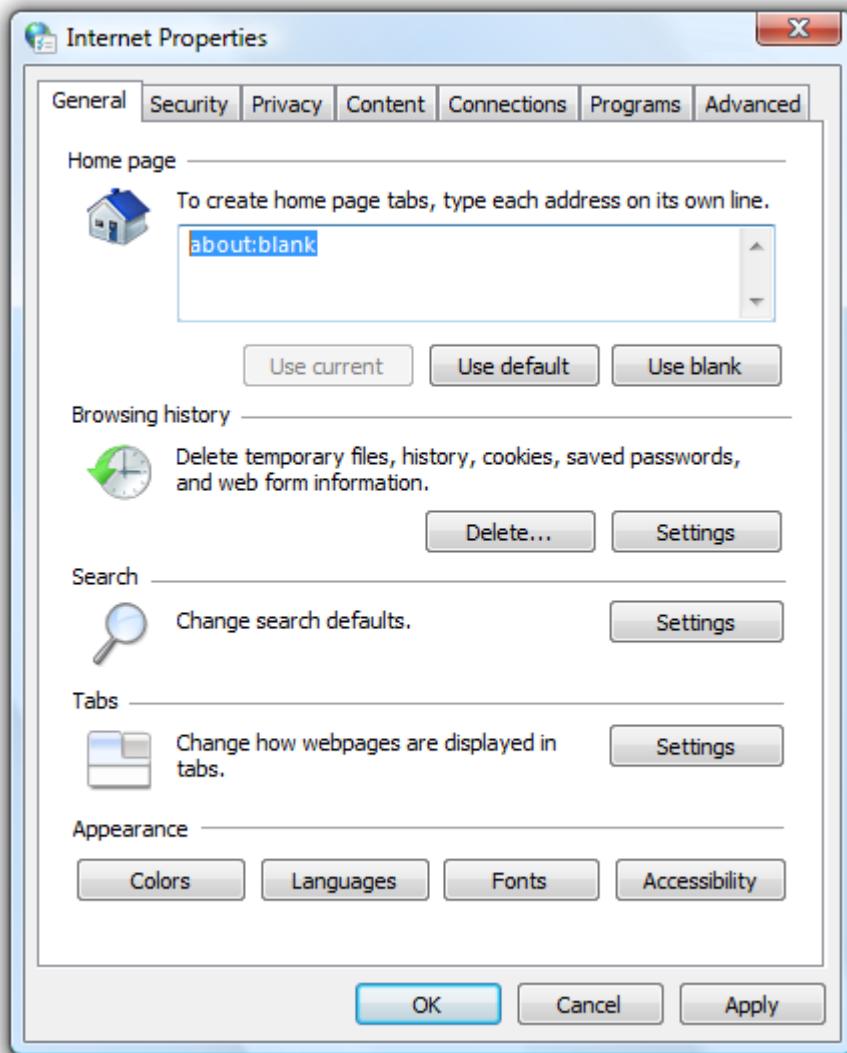
### プロパティ ウィンドウを効率的にまとめる

ユーザーがプロパティ ウィンドウに対して抱く、最も一般的な不満は、プロパティ ページにあまりに多くのタブとコンテンツが詰め込まれているという点です。これら両方の問題に対処するための手法をいくつか紹介します。

## タブの数を減らす

これまで、Windows では水平タブが使われてきました。水平タブは、タブが 1 行に収まる場合に最適です(最大で 5 ~ 7 タブ)。タブの数がそれ以上の場合には、垂直タブを使うとより効果的です。ただし、タブの数が 12 を上回るのは多すぎます。

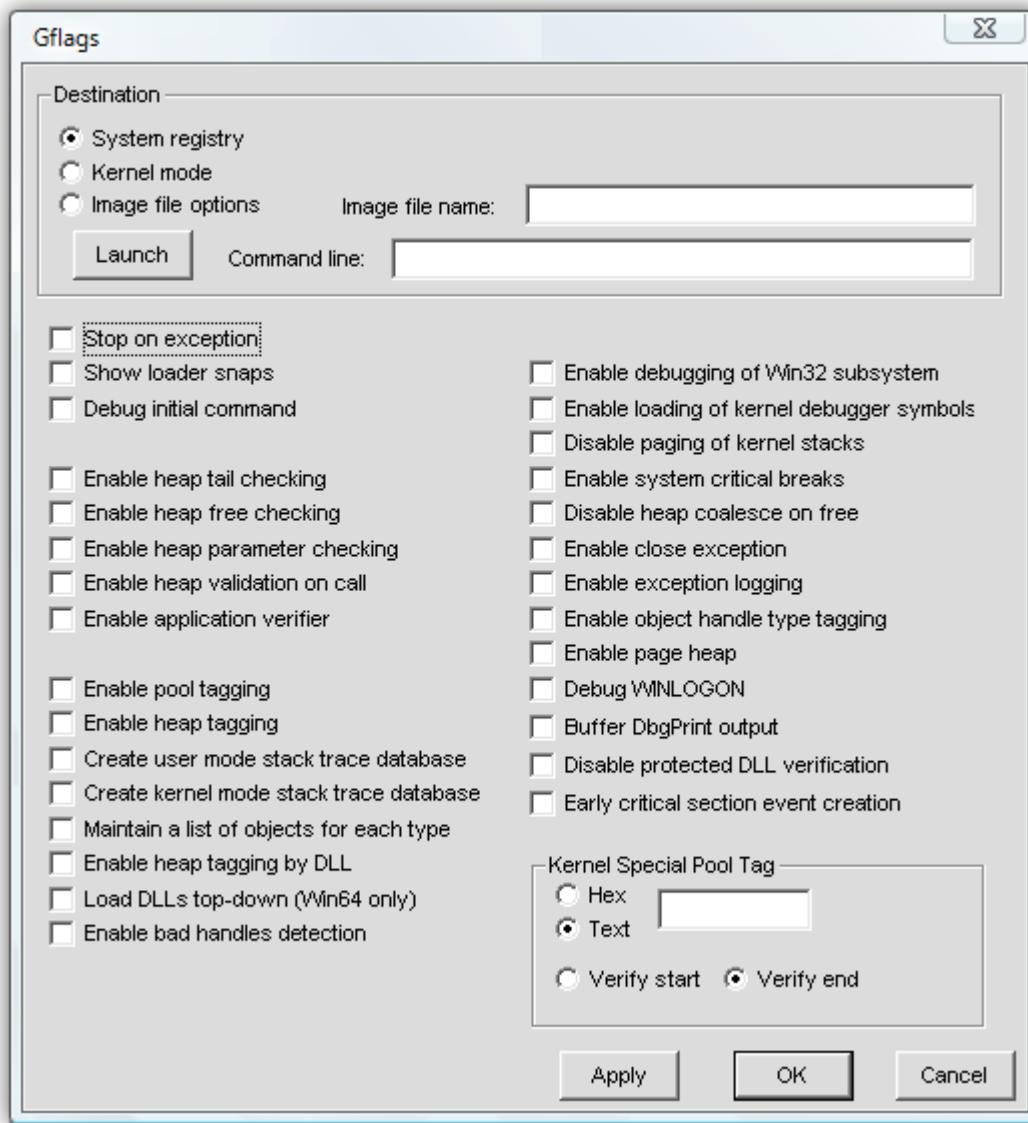
タブの数を減らすには、ユーザーによる使用頻度が最も少ないプロパティを特定します。次に、それらの設定をページ上に直接配置する代わりに、コマンド ボタンを使用して表示されるダイアログ ボックスに追加します。この手法は、ユーザーに対して既定の設定を維持することを推奨する場合に効果的です。



この例の Windows Internet Explorer の [全般] タブでは、変更頻度の少ない設定を別のダイアログ ボックスに移すことで、タブの数を減らしています。

もちろん、タブの数が少なすぎるケースもあります。

間違った例:



この例では、タブを使用して設定をまとめる必要があります。

#### ページに何を残すかを決定する

ページにどのプロパティを残すかを決めるには、プロパティのグループ分けと一貫性について検討します。次の場合、プロパティは適切にグループ分けされているといえます。

- ・ ページ上の設定に関連性があるか、相互に依存している。
- ・ ページ上の設定が他のページの設定に関連付けられていないか、依存していない。

あるページ上の設定が変更されても、他のページ上の設定が変更されないようにする必要があります。

次の場合、プロパティに一貫性があるといえます。

- ・ ページ上のすべてのプロパティが单一の概念に関係するものである。
- ・ 概念が具体的である。
- ・ 概念がテクノロジーではなく、タスクや目的に基づいている。

タブのラベルを見れば、設定にどのような関連性があるのかを確認できます。“全般”や“詳細設定”などの汎用的なラベルとは対照的に、具体的なラベルからは、内容に一貫性があることがはっきりとわかります。

#### ページを簡素化する

必要なプロパティを適切なレベルで表示する手法のほかに、次の手法を利用して、プロパティページの表示を簡素化することができます。

- プロパティ ページのサイズを大きくする。画面上の領域を占有するのをおそれる必要はありません。
- よりコンパクトなコントロールを使用する。
  - グループ ボックスではなく区切り記号でコントロールをグループ分けする。
  - 個別のラジオ ボタンを多数使用する代わりに、リスト ボックスを使用する。
  - 個別のチェック ボックスを多数使用する代わりに、チェック ボックスリストを使用する。
  - リスト ボックスの代わりにドロップダウン リストを使用する。
  - 関連性のあるコントロールのグループには、個別のコマンド ボタンの代わりに分割ボタンを使用する。
- 詳細な設定や使用頻度の少ない設定は別のダイアログ ボックスに移す。ダイアログ ボックスはコマンド ボタンを使用して表示します。
- ページ内のプロパティに関連性や一貫性があまりない場合は、ページを分割する。ページを分割して、プロパティを新しいページか、[全般] や [詳細設定] などの汎用的なページに移します。

## プロパティ ウィンドウの使用パターン

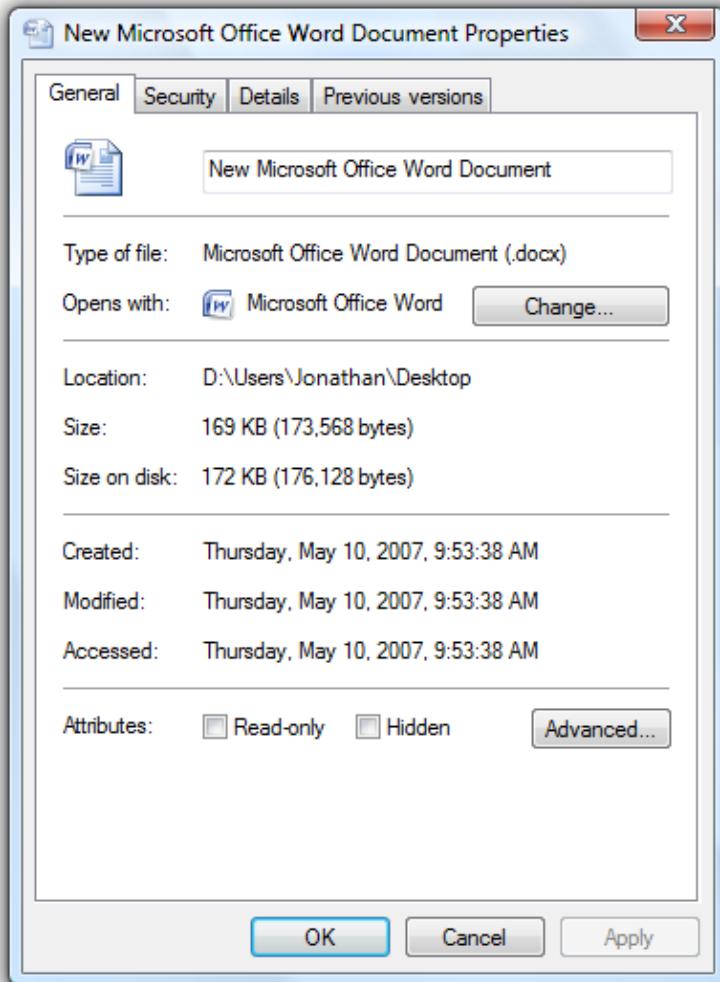
プロパティ ウィンドウ

プロパティ ウィンドウのデザインコンセプト

プロパティ ウィンドウにはいくつかの使用パターンがあります。

プロパティ シート  
ト  
单一のオブジエ  
クトのプロパ  
ティがモードレ  
ス ダイアログ  
ボックスに表示  
されます。

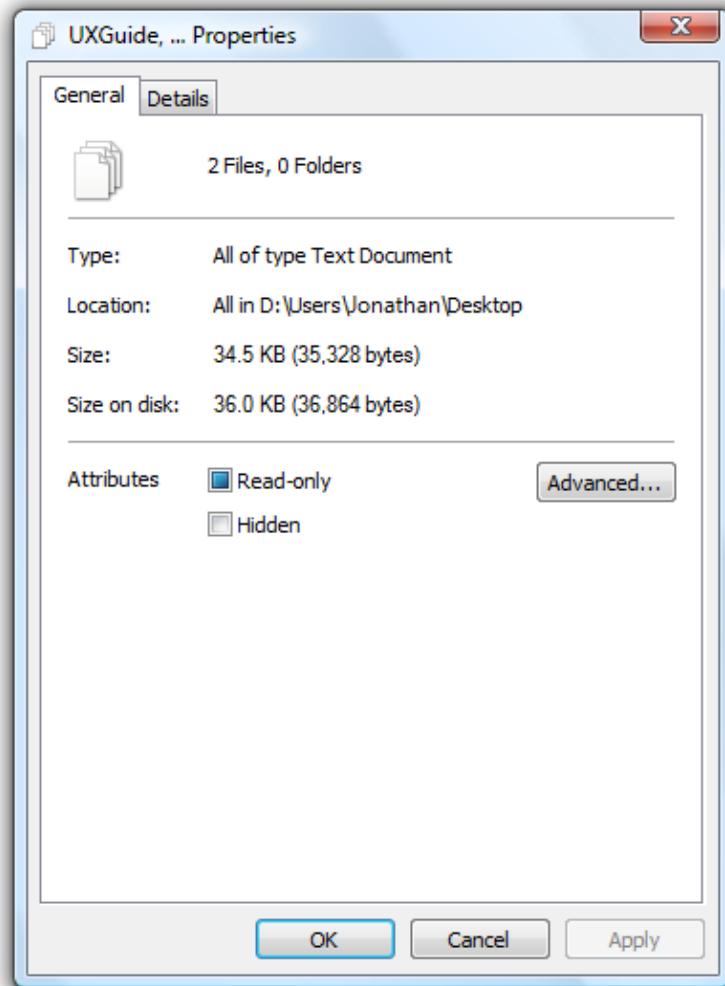
プロパティ シートは、オブジェクトに関してはモーダルです。ユーザーは、ダイアログ ボックスを閉じるか、再表示しないと、プロパティのオブジェクトを変更できません。プロパティ シートでは、遅延型のコミットが使用されます。つまり、ユーザーが [OK] や [適用] をクリックした場合に限り、変更内容が有効になります。



この例では、モードレス ダイアログ ボックスに单一のオブジェクトのプロパティが表示されています。

複数オブジェクトのプロパティ シート  
通常のプロパティ シートと似ていますが、このプロパティ シートには、オブジェクトのコレクションに関するプロパティだけが表示されます。設定の変更は、対象のすべてのオブジェクトに適用されます。

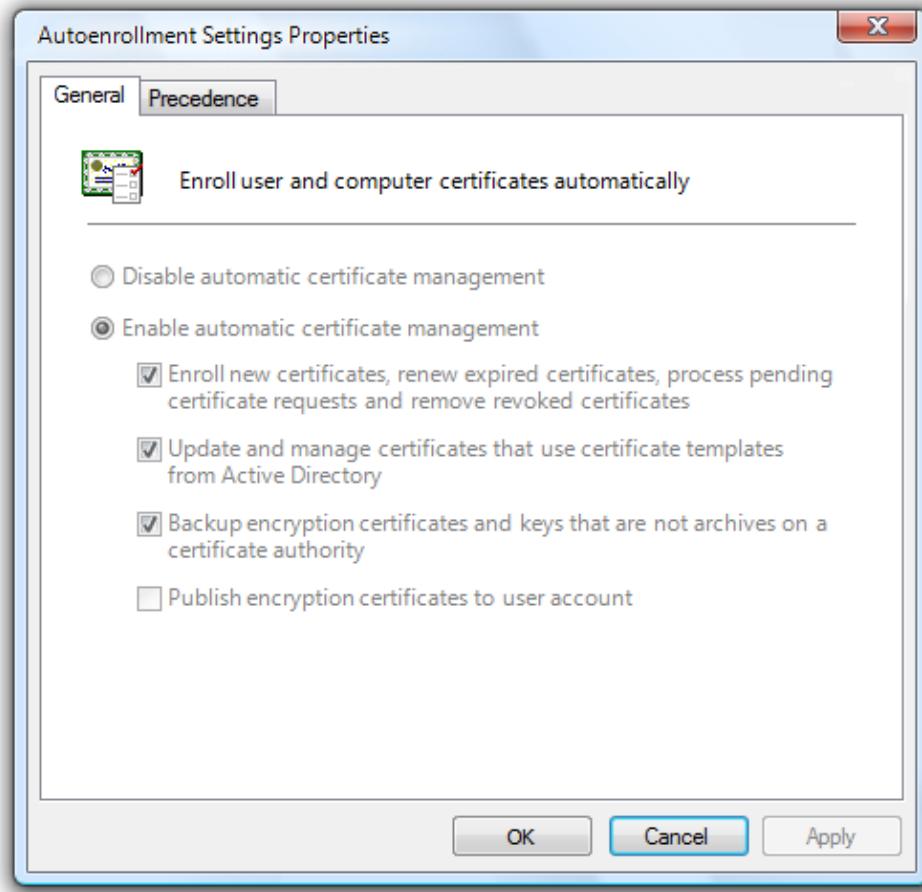
複数のオブジエ  
クトのプロパ  
ティがモードレ  
ス ダイアログ  
ボックスに表示  
されます。



この例では、プロパティ シートに複数のオブジェクトのプロパティが表示されています。混在状態のチェック ボックスは、選択されているファイルの一部が読み取り専用であることを示しています。この属性の変更内容は、対象のすべてのファイルに適用されます。

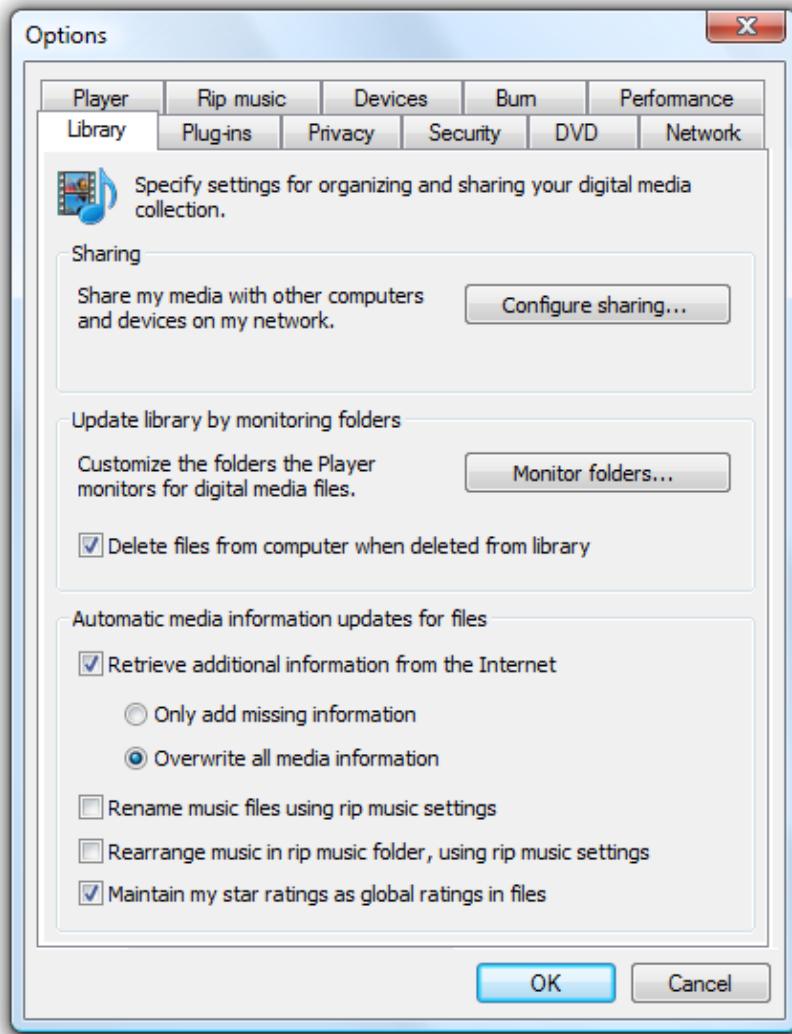
有効な設定のプロパティ シート オブジェクトの実際のプロパティが、そのオブジェクト自体の設定だけでなく、すべての親オブジェクトの設定によって決まる場合に使用されます(グループ ポリシーの場合など)。ユーザーによる変更が不可能なので、コントロールは無効になります。

单一のオブジェクトの有効なプロパティがモードレス ダイアログ ボックスに表示されます。



この例のプロパティ シートには、オブジェクト自体のプロパティと、そのすべての親オブジェクトのプロパティによって決まる有効なグループ ポリシーが表示されています。

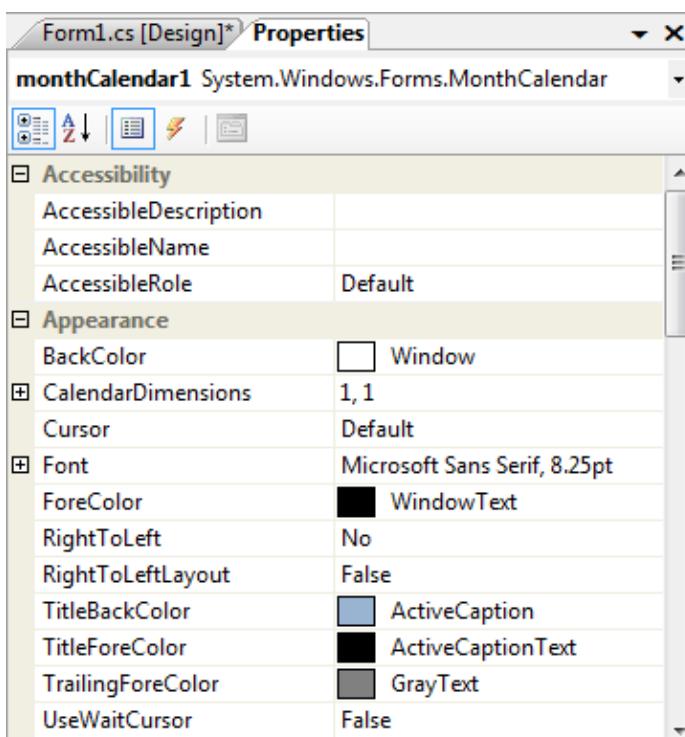
- オプション ダイアログ ボックスは、プロパティ シートと同様に、オプション ダイアログ ボックスでも遅延型のコミットが使用されます。そのため、ユーザーが [OK] をクリックした場合にのみ、変更内容が有効になります。オプション ダイアログ ボックスは、プロパティ シートとは次の点で異なります。
- アプリケーションの属性ではなく、アプリケーションの設定から構成される。アプリケーションの属性の表示には、[バージョン情報] ボックスが使用されます。
  - アプリケーションに対してモーダルである。
  - 通常、[適用] ボタンは不要。



この例では、アプリケーションのオプションがダイアログ ボックスに表示されています。

プロパティ シートとは異なり、プロパティ インスペクターには、現在選択されているオブジェクトのプロパティが表示されます。即時型のコミット(ユーザーが変更を行うと、プロパティがすぐに変更される)が使用されるため、[OK]、[キャンセル]、および [適用] ボタンは不要です。

プロパティ インスペクターには、オブジェクトのプロパティがモードレス ウィンドウペインまたは非ドッキング ウィンドウに表示されます。



この Microsoft® Visual Studio® の例では、オブジェクトのプロパティがモードレス ウィンドウ ペイン内

のグリッドに表示されています。



この例では、画像のプロパティがモードレス ウィンドウ ペインに表示されています。ユーザーは、サイズ以外のすべてのプロパティを変更できます。

## 外観

Windows® ベースのアプリケーションのユーザー エクスペリエンスを設計する際は、以下の外観に関する事項を検討する必要があります。それぞれの項目で、詳細を説明します。

- レイアウト
- ウィンドウ枠
- フォント
- 色
- アイコン
- 標準アイコン
- グラフィック要素
- サウンド

## レイアウト

デザインコンセプト

ガイドライン

画面の解像度と dpi

ウィンドウサイズ

コントロールのサイズ

コントロールの間隔

配置

フォーカス

配置

アクセシビリティ

推奨されるサイズと間隔

レイアウトメトリック

"レイアウト" とは、ウィンドウまたはページ内のコンテンツのサイズ、間隔、配置のことです。効果的なレイアウトは、魅力的な外観を実現するだけでなく、ユーザーがすばやく目的のものを探し出す際にきわめて重要な役割を果たします。ユーザーがすぐに理解できるデザインになるか、それとも困惑したり圧倒されるデザインになるかは、レイアウトが効果的かどうかで決まります。

注: ウィンドウの管理に関するガイドラインは、別の項目として記載しています。特定のコントロールに推奨されるサイズと間隔については、各コントロールのガイドライン項目を参照してください。

## デザイン コンセプト

視覚的な階層

要素の関連性と優先度が外観で示されているウィンドウまたはページでは、視覚的な階層が明確です。視覚的な階層がないと、ユーザーはこれらの関連性と優先度を自分で考えて見つけ出す必要があります。

視覚的な階層は、次のような属性を巧みに組み合わせることで実現できます。

- フォーカス。ユーザーが最初に見る必要がある場所がレイアウトで示されている。
- フロー。画面上に明確な道すじが示されていて、自然に目で追っていくと、使用に適した順番でユーザーインターフェイス (UI) の要素をスムーズに見つけ出すことができる。
- グループ化。論理的に関連する UI 要素には、関連性が視覚的に明確に示されている。関連するアイテムはまとめてグループ化され、関連しないアイテムは分けられている。
- 強調。UI 要素は、相対的な重要性に基づいて強調されている。
- 配置。UI 要素は適切に並べられ、流し読みしやすく、整頓されている。

さらに、効果的なレイアウトには次のような属性が含まれます。

- デバイス非依存。フォントの書体やサイズ、解像度 (dpi)、ディスプレイ、グラフィックアダプターに関係なく、レイアウトは意図したとおりに表示される。
- 流し読みのしやすさ。ユーザーは、目的のコンテンツをひとめで見つけることができる。
- 効率性。大きくする必要のある UI 要素は大きく、小さいサイズでうまく機能する要素は小さくなっている。
- サイズの可変性。有用な場合に、ウィンドウのサイズ変更が可能になっていて、そのコンテンツのレイアウトは画面の大小にかかわらず効果的である。
- バランス。コンテンツは、画面上に均等に分布している。
- 視覚的な単純さ。レイアウトが必要以上に複雑化されていない。レイアウトの外観を見てユーザーが圧倒されることがない。
- 一貫性。類似のウィンドウまたはページでは、類似のレイアウトが使用されていて、ユーザーが常に適応できるようになっている。

サイズ、間隔、および配置は単純な概念です。レイアウトの課題はこれらの各属性を適切に組み合わせることです。

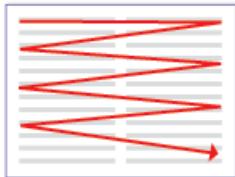
Microsoft® Windows® では、ダイアログ ユニット (DLU) や相対ピクセルなど、デバイスに依存しないメトリックを使用して、レイアウトが設計されています。これらのレイアウト メトリック、およびその測定方法と変換方法の詳細については、[レイアウト メトリック](#)を参照してください。

読み取りを考慮した設計モデル

ユーザーは、コンテンツの外観や編成から内容を読み取って選択します。効果的なレイアウトを作成するには、ユーザーが何を読み取る傾向にあるか、およびその理由を理解する必要があります。

以下の読み取りを考慮した設計モデルを使用して、レイアウトを決定できます。

- 人は左から右、上から下に向って読みます (西欧文化圏の場合)。
- 読み方としては、精読と流し読みの 2 つのモデルがあります。精読のモデルは、理解を目標とします。

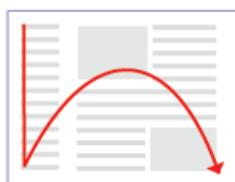


この図は、精読のモデルを示しています。

これに対し、流し読みのモデルは、場所の特定を目標とします。流し読みのモデルの全体的な視線の動きは、次のようにになります。

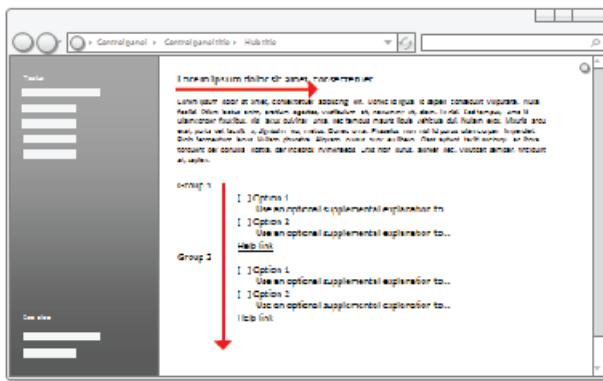


この図は、流し読みのモデルを示しています。



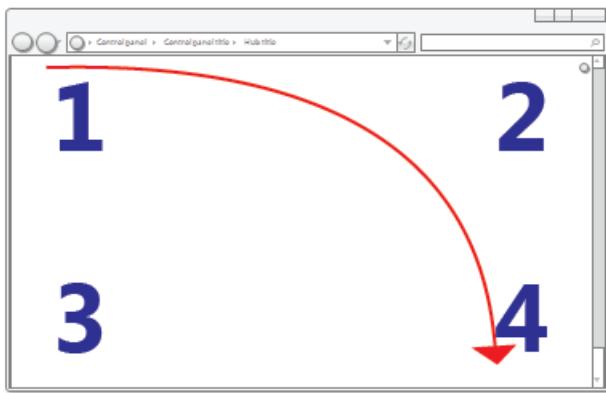
ページの左端に沿ってテキストがある場合は、ユーザーは最初に左端を流し読みします。

- ・ソフトウェアを使用するときは、ユーザーは UI 自体ではなく作業に没頭します。したがって、ユーザーは、通常、UI テキストを読まずに流し読みし、必要なときのみテキストの断片を詳しく読みます。
- ・ユーザーは、ページの左側や右側にあるナビゲーション ウィンドウを読み飛ばす傾向にあります。ナビゲーション ウィンドウがあることは認識しますが、移動するときだけナビゲーション ウィンドウを確認します。
- ・体裁が整えられていない大きなテキスト ブロックは、まったく読まれないおそれがあります。



大きなテキスト ブロックやナビゲーション ウィンドウは、流し読みのときに読み飛ばされる傾向にあります。

- ・均一な状態であれば、ユーザーはウィンドウの左上隅を見て、ページ全体を流し読みし、右下隅で終了します。左下隅は無視される傾向にあります。



均一な状態であれば、ユーザーは 1、2、4、3 の順番で読み取ります。

- ただし、対話型 UI では、要素が均一ではないため、注目される度合いは UI 要素ごとに異なります。ユーザーは、対話型コントロール(特にウィンドウの左上や中央のコントロール)および目立つテキストを最初に見る傾向にあります。

#### Use the computer without a display

When you enter these settings, they will automatically start each time you log on.

Your text and speech

Turn on Narrator

Narrator reads aloud any text on the screen. You will need speakers.

Turn on Audio Description

Hear descriptions of what's happening in video (when available).

Turn up Text to Speech

Adjust how links and floating windows

Turn off all unnecessary animations (when possible)

How long should Windows notification dialog boxes stay open?

7.0 seconds ▾

See also:

[Audio Devices and Sound Themes](#)

[Learn about additional assistive technologies online](#)

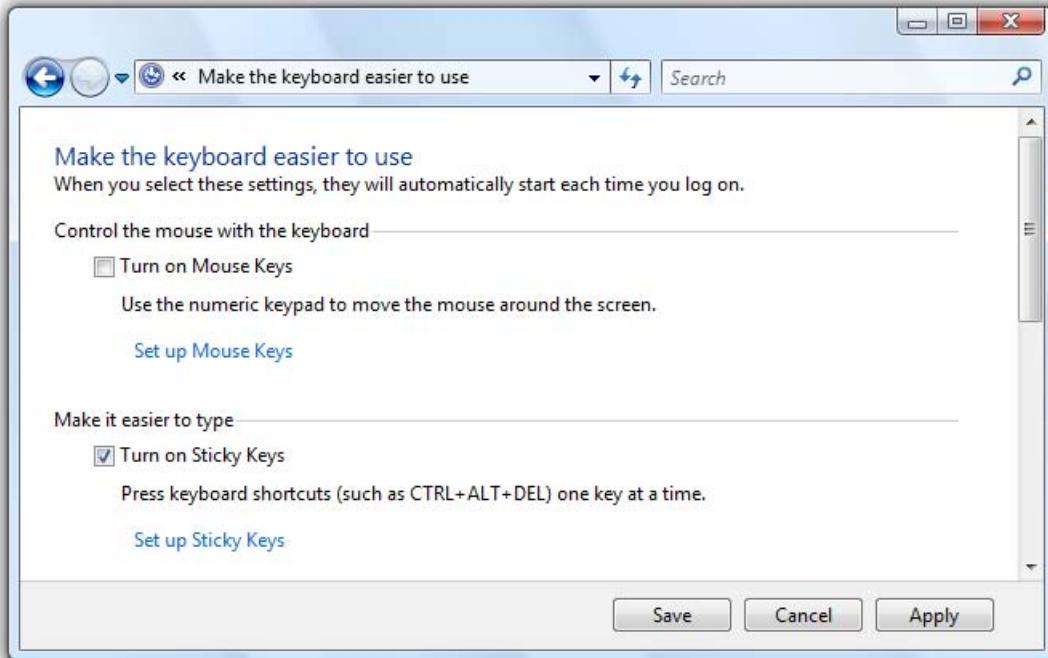
Save

Cancel

Apply

ユーザーは、メインの対話型コントロールおよび目立つメイン指示テキストに注目し、その他は必要なときだけ確認します。

- 対話型コントロールのラベル、特に作業中のタスクの完了に関連して表示されるコントロール ラベルは読まれる傾向にありますが、静的テキストはユーザーが必要と判断したときだけ読されます。
- 外観が他と異なるアイテムは注意を引き付けます。太字のテキストと大きなテキストは、標準のテキストより目立ちます。色付きの UI アイテム、または色付きの背景上の UI アイテムは目立ちます。アイコン付きのアイテムは、アイコンなしのアイテムより目立ちます。
- ユーザーは、理由がない限りスクロール操作をしません。一面トップにあるコンテンツ内で、スクロールする理由が見当たらなければ、ユーザーはスクロール操作をしません。
- 行動を決定したユーザーは、すぐに読むのをやめて操作を行います。
- ユーザーは、用が済めば読むのをやめるので、必要な操作を完了できる所より後に表示されるものはすべて無視される傾向にあります。



ユーザーは、用が済めば読むのをやめます。

もちろん、この一般的なモデルにも例外はあります。視線追跡デバイスを使ったテストでは、実際のユーザーの視線の動きには、かなりむらがあることが示されています。このモデルの目標は、ユーザーの動作を正確にモデル化することではなく、設計における適切な取捨選択に役立てることです。もちろん、ここに挙げたことから、自分自身の読み方のパターンを多く把握できるに越したことはありません。

#### 流し読みのための設計

ユーザーは精読ではなく流し読みをします。したがって、流し読みを想定して UI 画面を設計する必要があります。テキストが記述されているとおりに左から右、上から下に読まれるとは考えずに、ユーザーの注意を引く UI 要素が読まれると想定して設計します。

流し読みを想定して設計するには、以下に従います。

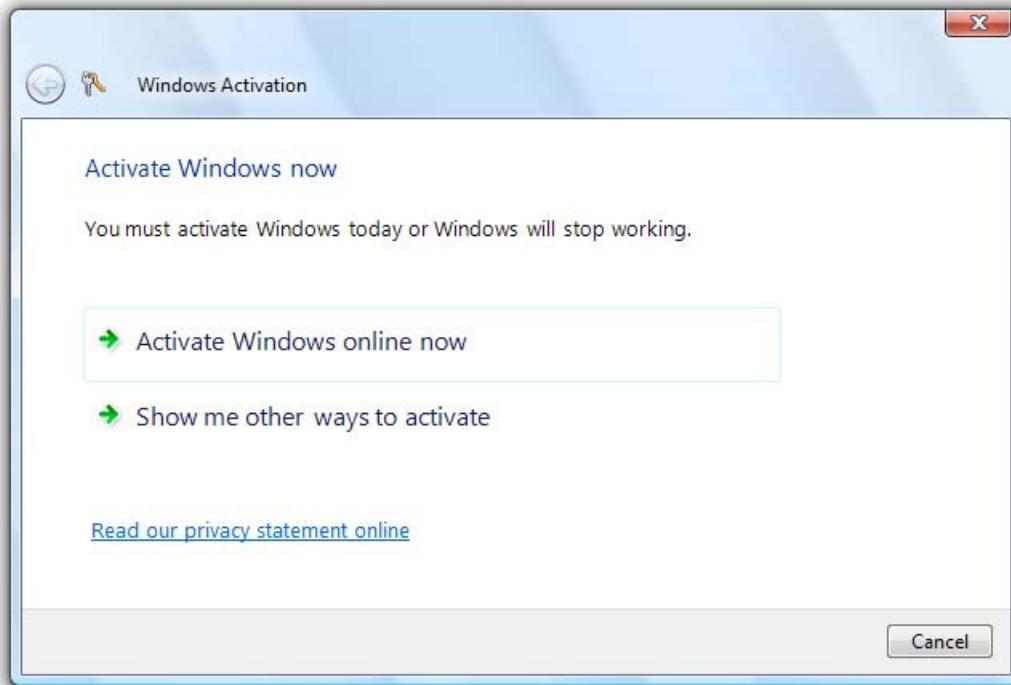
- ユーザーは、初めにウィンドウ全体をざっと見渡し、主に次のような順序で UI 要素を読み取ると想定します。
  - 中央の対話型コントロール
  - コミットボタン
  - 中央以外の対話型コントロール
  - メイン指示テキスト
  - 補足説明
  - 警告アイコンが付いたテキスト
  - ウィンドウ タイトル
  - UI 内のその他の静的テキスト
  - 脚注
- タスクを開始する UI 要素は、左上隅または中央上に配置します。
- タスクを完了する UI 要素は、右下隅に配置します。
- 可能な場合、重要なテキストは静的テキストにするのではなく、対話型コントロールに配置します。
- スクロール可能な長いコントロール/ページの左下角や下部には、重要な情報を配置しないようにします。
- 大きなテキストブロックは提示しないようにし、不要なテキストは削除します。**逆ピラミッド型**の提示スタイルを使用します。
- 何らかの手段でユーザーの注意を引き付ける場合は、注意を引くことに正当な根拠があることを確認します。

できる限りこのモデルに沿って設計します。ただし、特定の UI 要素を強調したり、目立たないようにする必要が出てくることもあります。

プライマリ UI 要素を強調するには、以下に従います。

- プライマリ UI 要素は、**視線の通り道**に配置します。
- タスクを開始する UI 要素は、左上隅または中央上に配置します。
- コミットボタンは、右下隅に配置します。
- 残りのプライマリ UI は、中央に配置します。
- コマンドボタン、コマンドリンク、アイコンなど、注意を引き付けるコントロールを使用します。
- 大きなテキストや太字のテキストなどの目立つテキストを使用します。

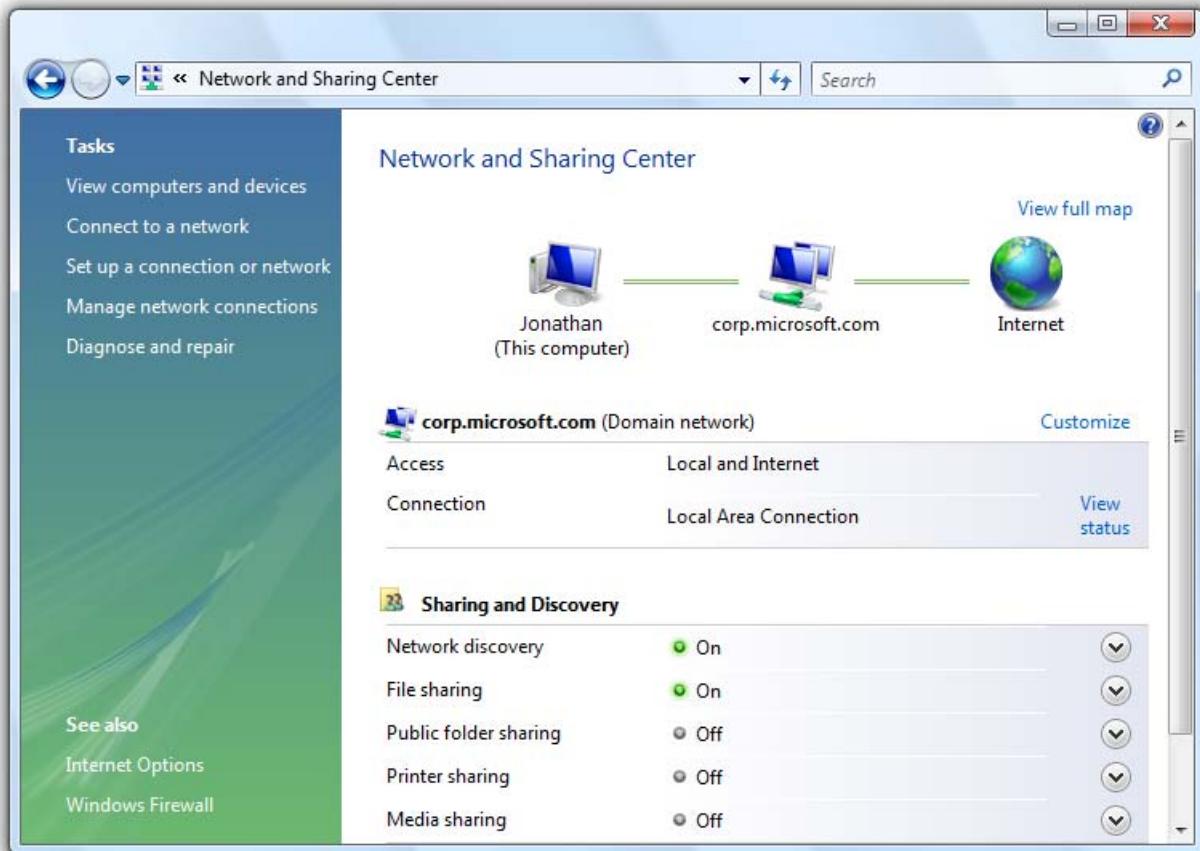
- ユーザーが必ず読む必要があるテキストは、対話型コントロール内に配置するか、アイコンを付けるか、またはバナー上に表示します。
- 明るい背景に濃い色のテキストを使用します。
- 要素の周りの間隔は広めにします。
- 強調している要素をユーザーが確認するために、マウスでポイントするなど、対話操作が必要になるような設計はしないでください。



この例では、プライマリ UI 要素を強調するさまざまな方法を示しています。

セカンダリ UI 要素を目立たなくするには、以下に従います。

- セカンダリ UI 要素は、視線の通り道の外に配置します。
- ユーザーが通常確認する必要がない要素は、ウィンドウの左下隅または下部に配置します。
- コマンド ボタンの代わりに、注意を引き付けないタスク リンクなどのコントロールを使用します。
- 通常のテキストまたは灰色のテキストを使用します。
- 暗い背景に明るい色のテキストを使用します。濃い灰色や青い背景に白いテキストを使用すると効果的です。
- 要素の周りの間隔は最小限にします。
- セカンダリ UI 要素を隠すために段階的表示の使用を検討します。

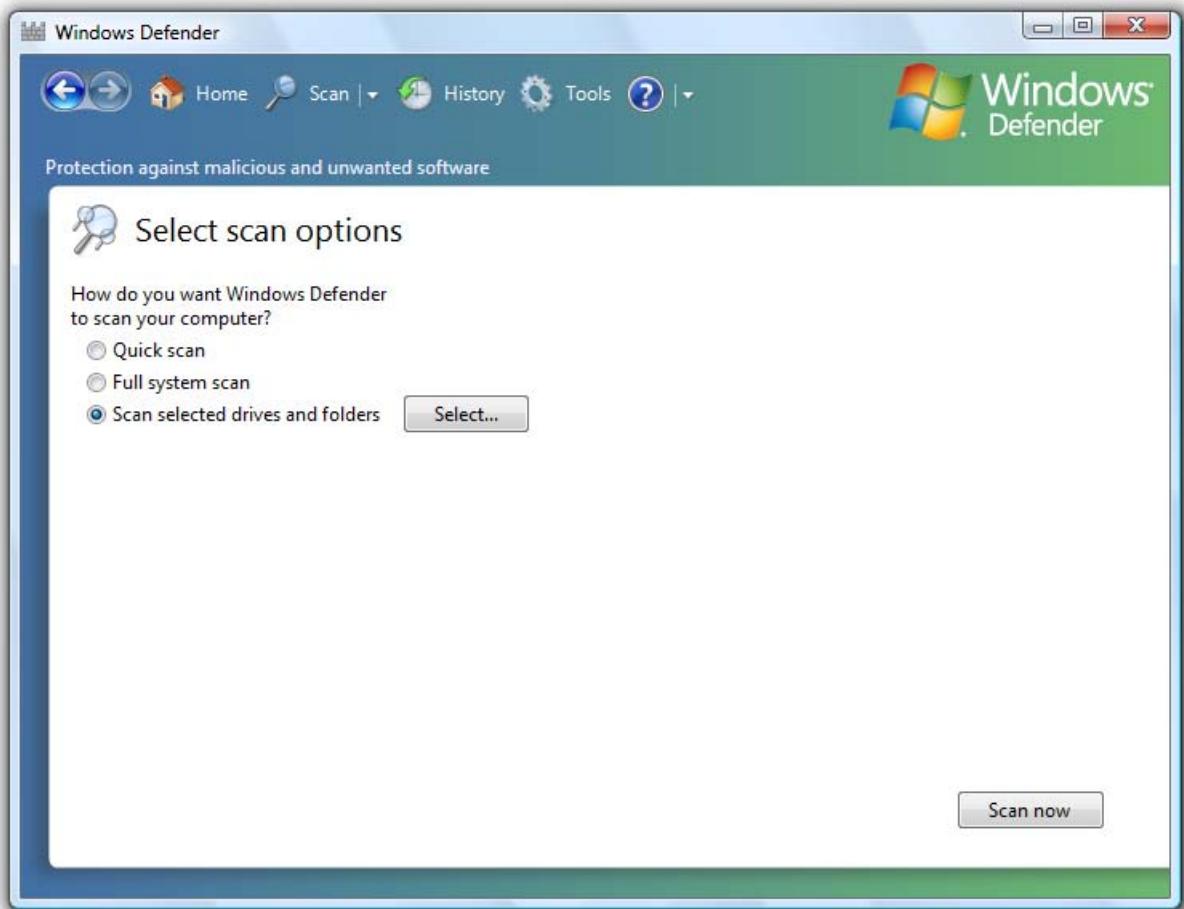


この例では、セカンダリ UI 要素を目立たなくするさまざまな方法を示しています。

#### 画面領域の有効利用

画面領域を有効に使用するには、いくつかの要因のバランスをとる必要があります。使用する領域が広すぎると、ウィンドウが重く感じられ、無駄に見えます。また、[フィットの法則](#)から判断すると、使いにくい UI になります。

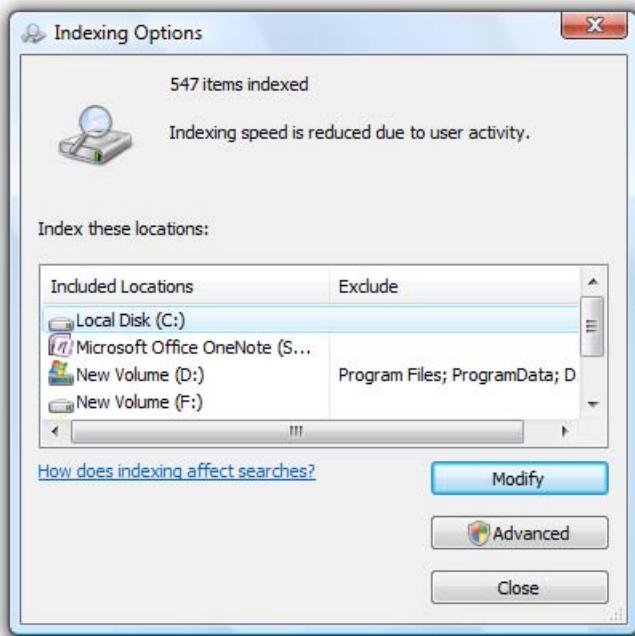
間違った例:



この例では、ウィンドウがコンテンツに対して大きすぎます。

一方、領域が狭すぎると、窮屈で落ち着かない、威圧的なウィンドウになります。また、スクロールや他の操作が必要となる場合は、使いづらくなります。

間違った例:



この例では、ウィンドウがコンテンツに対して小さすぎます。

重要な UI は、サポートされる最も低い有効解像度内に収める必要があります。ただし、画面領域の有効利用とは、ウィンドウの大きさを最小限にするという意味ではありません。ウィンドウの大きさを最小限にはしないでください。効果的なレイアウトとは、適切な空き領域を残したものであり、最小限の領域にすべてを詰め込んだものではありません。最近のディスプレイの画面領域は大きいので、その領域を有効に使用する方が理にかなっています。したがって、画面領域を節約しすぎるよりも、思い切って広めに使った方がうまく行きます。これにより、ウィンドウはより軽快な外観になり親しみやすいものになります。

レイアウトで画面領域が有効利用されているかどうかは、以下の点に基づいて判断します。

- サイズを変更せずに、ウィンドウおよびコントロールを使用できる。ユーザーが最初に行う操作がウィンドウやコントロールのサイズ変更であるとしたら、そのサイズは正しくありません。
- データが切り詰められておらず、リスト ビューやツリー ビューのほとんどのデータには省略記号が付いていない。また、他のコントロールでも、長すぎない限りデータが切り詰められていない。タスクの実行のために読む必要があるデータは、切り詰められることがないようにします。
- ウィンドウやコントロールは適切なサイズに設定され、不要なスクロールが省かれている。水平スクロールバーはほとんどなく、不要な垂直スクロールバーもない。
- ほとんどのコントロールで標準サイズが使用されている。たとえば、画面上で使用するコマンド ボタンの幅を 2 種類以内にとどめるなど、コントロール サイズの種類を減らすように努力している。
- UI 画面のバランスがとれており、未使用の大きな画面領域はない。

目的を果たすためにちょうど良い大きさのウィンドウ サイズを選択します(ウィンドウがサイズ変更可能な場合は、このサイズを既定にします)。データが切り詰められていたりスクロールバーが存在しているにもかかわらず、利用可能な大きい画面領域が残っているとしたら、それらは明らかに、領域を有效地に使用していないレイアウトです。

#### コントロールのサイズ

通常、画面領域を有效地に使用するための最初のステップは、さまざまな UI 要素の適切なサイズを判断することです。コントロール サイズの表および特定のコントロールのガイドライン項目内の推奨されるサイズを参照してください。

フィットの法則では、ターゲットが小さくなればなるほど、マウスで捕捉するための時間が長くなるとされています。さらに、Windows Tablet とタッチ テクノロジーを使用したコンピューターでは、"マウス"が実際にはペンやユーザーの指である場合があるため、小さいコントロールのサイズを決定する際には、別の入力デバイスも考慮する必要があります。すべての入力デバイスに対応可能で、適切なコントロールの最小サイズは、 $16 \times 16$  ピクセルです。標準で  $15 \times 9$  相対ピクセルのスピinnコントロールボタンは、小さすぎてペンではうまく操作できません。

#### 間隔

間隔を広めにとることで(ただし広すぎないように注意します)、レイアウトはゆったりとしたわかりやすいものになります。効果的な間隔とは、単なる未使用の領域ではありません。効果的な間隔によって、内容が見渡しやすくなり、デザインも視覚的な魅力も向上します。ガイドラインについては、間隔の表を参照してください。

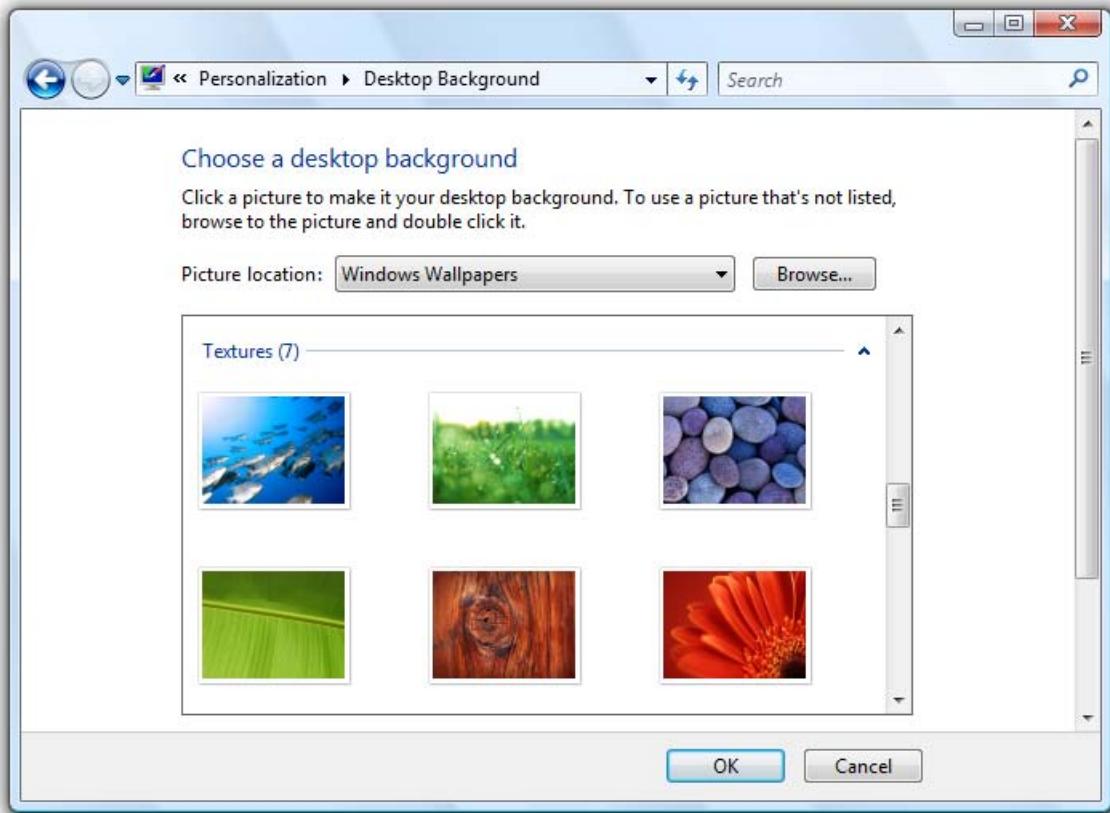
ここでも、Windows Tablet とタッチ テクノロジーを使用したコンピューターでは、"マウス"が実際にペンやユーザーの指となる場合があることを考慮する必要があります。ペンや指をポインティング デバイスとして使用するときは、ターゲットに合わせるのがより困難になり、意図するターゲットの外部をユーザーがタップすることができます。対話型コントロールが非常に近くに配置されているもの実際には接触していない場合、ユーザーがそれらのコントロールの間の非アクティブな領域をクリックすることができます。非アクティブな領域をクリックしても効果や視覚的なフィードバックが得られないで、多くの場合ユーザーは何が悪かったのかがわかりません。小さなコントロールが狭すぎる間隔で配置されていると、間違って別のオブジェクトをタップしないように、ユーザーは正確にタップしなければならなくなります。これらの問題に対処するには、対話型コントロールのターゲット領域を接触させるか、またはコントロール間に 3 DLU (5 相対ピクセル) 以上の間隔を確保します。

レイアウト内の間隔が適切かどうかは、以下の点に基づいて判断します。

- 全体的に UI 画面がゆったりとしていて窮屈ではない。
- 間隔が均一でバランス良く表示されている。
- 関連する要素どうしは近くにまとめて配置され、関連しない要素は相対的に大きく離されている。
- ツールバー ボタンなど、もともと密接することが意図されているコントロール間に、不必要的間隔がない。

#### サイズ変更可能なウィンドウ

サイズ変更可能なウィンドウも、画面領域を有效地に使用するうえでの重要な要素です。ウィンドウの中には、固定されたコンテンツで構成され、サイズ変更を可能にしてもメリットがないものもありますが、サイズ変更可能なコンテンツで構成されるウィンドウは、サイズを変更できるようにする必要があります。ユーザーがウィンドウを拡大する理由は、もちろん広い画面領域を活用するためであり、それには、必要な UI 要素の領域を広げることにより、コンテンツを広げる必要があります。動的コンテンツ、ドキュメント、イメージ、リスト、ツリーを含むウィンドウは、サイズ変更のメリットが最も得られるウィンドウです。

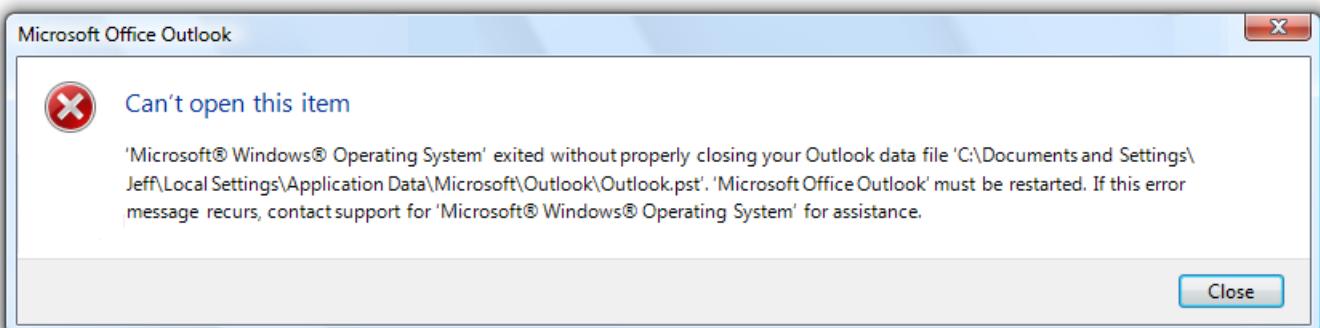


この例では、ウィンドウ サイズの変更によって、リスト ビュー コントロールのサイズが変更されています。

ウィンドウの幅が過剰に広げられる可能性も考慮する必要があります。たとえば、コンテンツの幅が 600 相対ピクセルより広いと、コントロール パネルのページの多くは扱いにくくなります。このような場合、この最大幅を越えてコンテンツ エリアを大きくできるようにしたり、ウィンドウの拡大に合わせてコンテンツの始点を変更したりせずに、最大幅を保持して左上の始点は固定します。

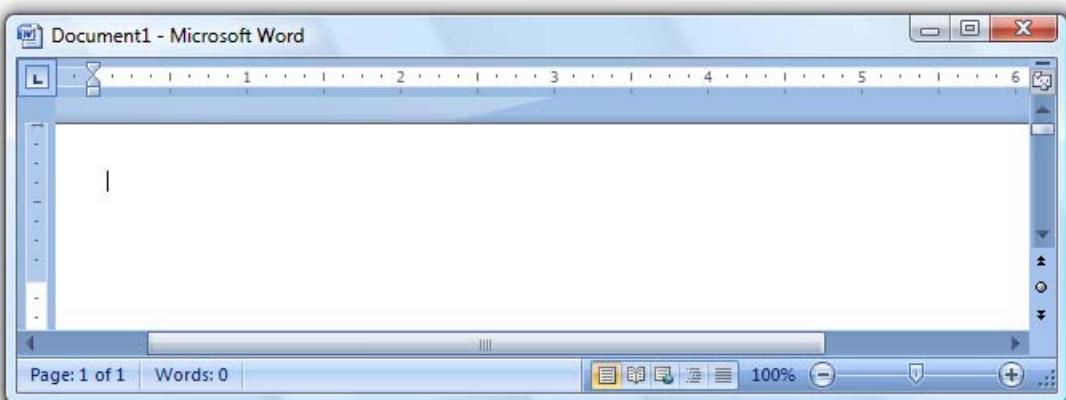
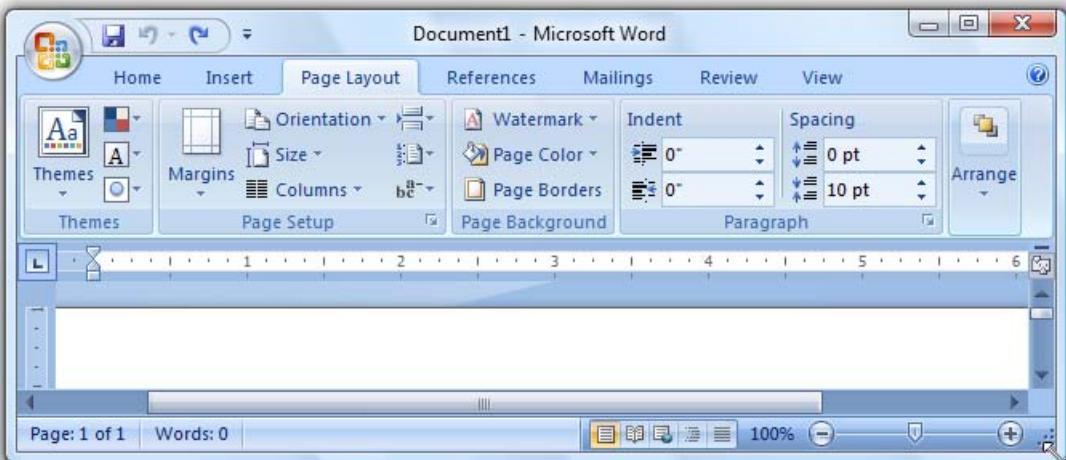
1 行の文字数が多くなるにつれて、テキストは読みにくくなります。テキスト ドキュメントでは、1 行の文字数は半角換算で 80 文字以内になるようにします。これより長いとテキストが読みにくくなります(句読点や空白文字も含めてカウントします)。

間違った例:



この例では、1 行の文字数が多く、読みにくくなっています。

最後に、サイズ変更可能なウィンドウを縮小する場合についても、サイズ変更可能なコンテンツを小さくしたり、間隔なしでも効果的に機能できる UI 要素は間隔を詰めたりして、画面領域を有効に使用する必要があります。縮小していくと、ある時点でウィンドウや UI 要素が小さすぎて利用できなくなります。この対処として、ウィンドウや UI 要素に最小サイズを割り当てたり、一部の要素を完全に削除する必要があります。



この例では、ウィンドウが最小サイズになっています。

コンテンツを小さいサイズで利用可能にするために、完全に異なる提示方法を使用するプログラムもあります。



この例では、Windows Media Player® が小さくされすぎて標準の形態を維持できなくなると、形態が変わることを示しています。

### フォーカス

最初に目につくはっきりとした場所が 1 つある場合、レイアウトには "フォーカス" があります。 フォーカスは、ウィンドウまたはページをどこから読めばいいかをユーザーに示すという重要な役割を果たします。明確なフォーカスなしでは、ユーザーの視線は当てもなくさまようことになります。 フォーカルポイントは、ユーザーがすばやく見つけ出して理解する必要のある重要な部分に置きます。また、視覚的に最も強調する必要があります。左上隅は、ほとんどのウィンドウにとって自然なフォーカルポイントです。

フォーカルポイントは、1 つだけにします。現実世界では、視線は一度に 1 つのものだけに焦点を合わせます。ユーザーは複数の場所に同時に焦点を合わせることはできません。

UI 要素をフォーカル ポイントにするには、以下の方法で視覚的に強調します。

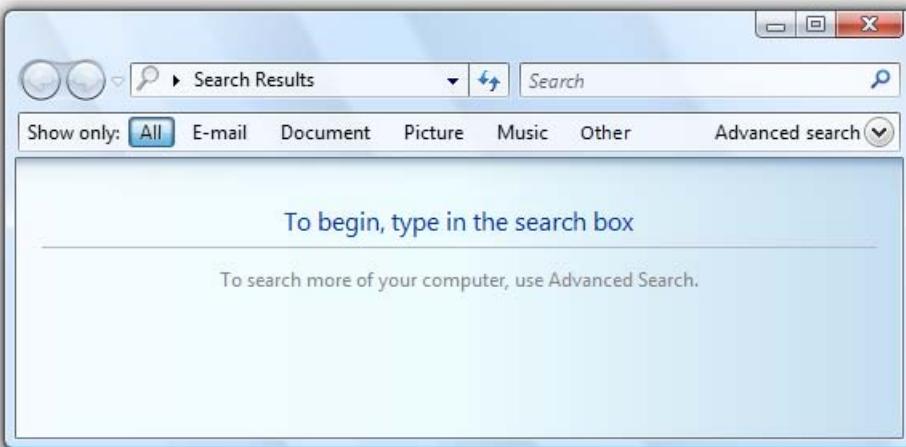
- UI 画面の左上または中央上に配置します。
- 重要かつわかりやすい対話型コントロールを使用します。
- メイン指示テキストなどの目立つテキストを使用します。
- コントロールが既定で選択され、初期入力フォーカスが置かれているようにします。
- コントロールの背景を異なる色にします。

ここで、Windows Search を例として挙げます。Windows Search のフォーカル ポイントは、タスクの始点である検索ボックスである必要があります。しかし、これは標準の検索ボックスの配置との一貫性を維持するために右上隅に配置されています。検索ボックスには入力フォーカスがありますが、視線の通

り道と合わせて考えると、これだけでは不十分です。

この問題に対処するために、ウィンドウの中央上の部分に、ユーザーを正しい場所に誘導する目立つ指示テキストを置いています。

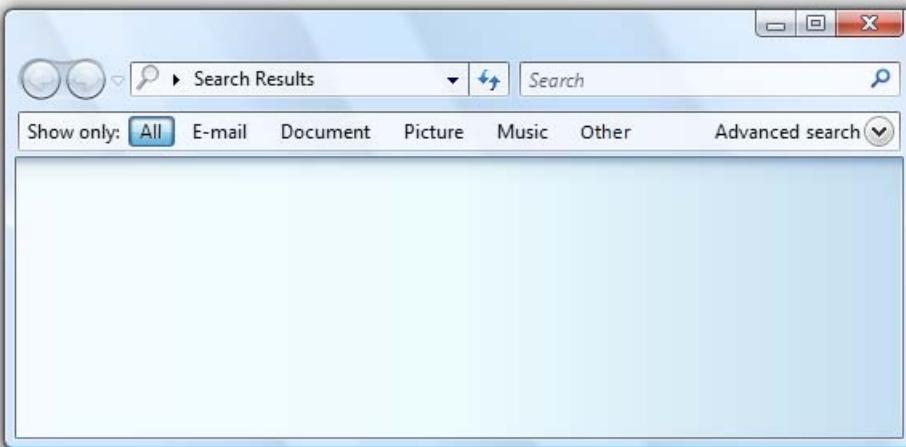
許容される例:



この例では、ウィンドウの中央上の目立つ指示テキストで、ユーザーを検索ボックスに誘導しています。

指示テキストなしでは、このウィンドウのフォーカル ポイントは明確にはなりません。

間違った例:



この例では、明確なフォーカル ポイントがなく、ユーザーはどこを見たら良いかわかりません。

UI 要素を視覚的に強調する場合は、注意を引くことに正当な根拠があることを確認します。前の Windows Search の間違った例では、強調された [すべて] ボタンが左上隅にあり、視覚的に最も強調されていますが、これは意図されるフォーカル ポイントではありません。ユーザーは、このボタンでできることを把握しようとして、行き詰まってしまう場合があります。

間違った例:



フォーカル ポイントとしての目立つ指示テキストがないと、強調された [すべて] ボタンが意図しない フォーカル ポイントとなります。

フロー

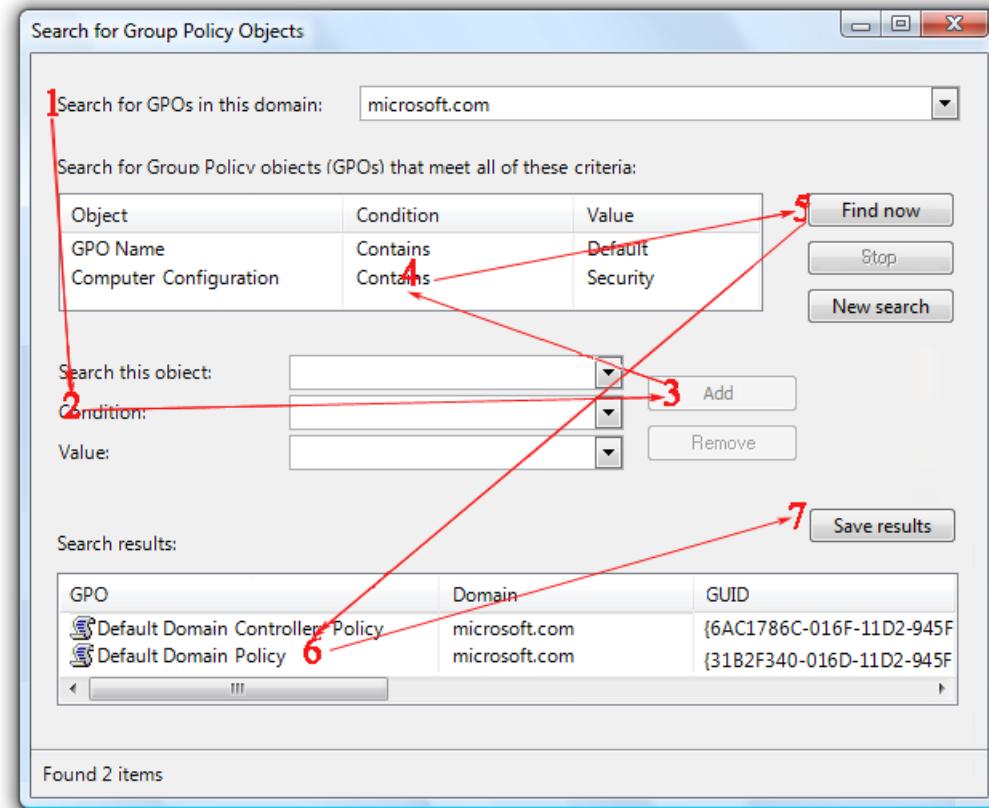
UI 画面上を通る明確な道筋によってユーザーがスムーズかつ自然に誘導され、使用に適した順序で UI 要素を確認できる場合、レイアウトには "フロー" があります。ユーザーは、フォーカル ポイントを特定したら、次はタスクを完了する方法を判断する必要があります。UI 要素の配置は、要素間の関連性を伝えるものであり、タスクの実行手順を反映している必要があります。つまり、タスクの手順を、左から右、上から下へと向かう (西欧文化圏の場合) 自然なフローにする必要があるということです。

レイアウトのフローが適切かどうかは、以下の点に基づいて判断します。

- UI 要素の配置に、ユーザーがタスクを実行するために必要な手順が反映されている。
- タスクを開始する UI 要素は、左上隅または中央上に配置されている。
- タスクを完了する UI 要素は、右下隅に配置されている。
- 関連する UI 要素はまとめられ、関連しない要素は分けられている。

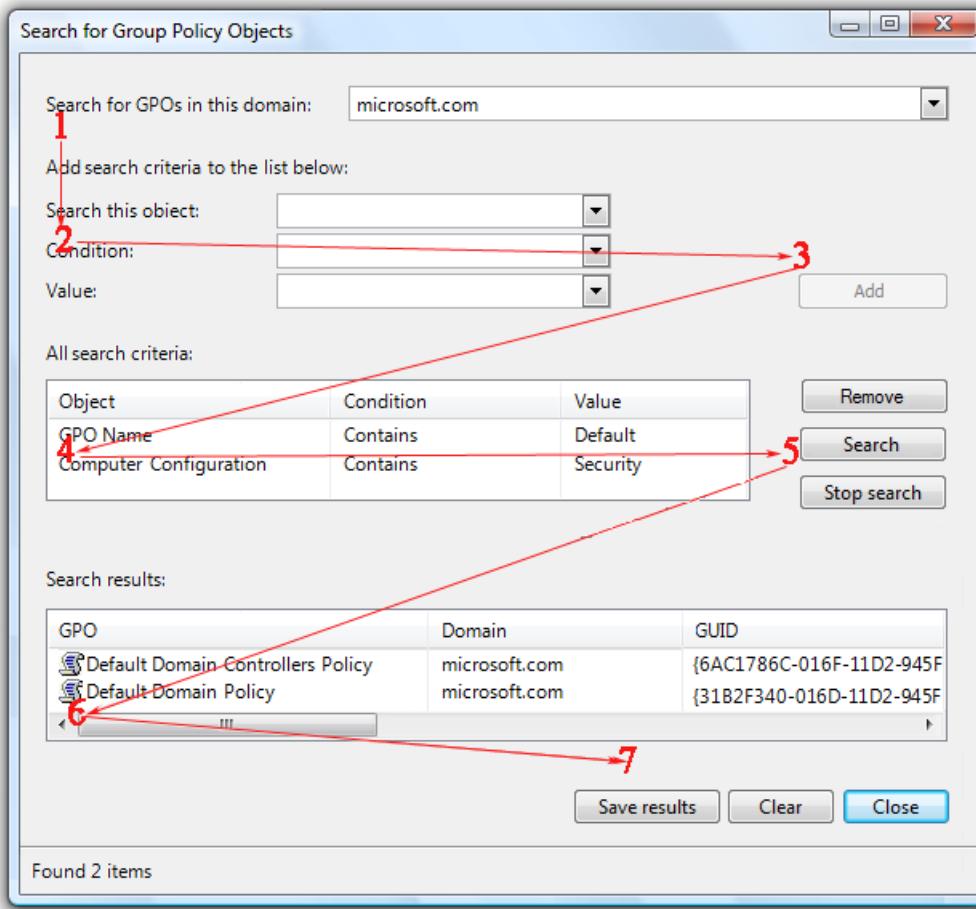
- 必要な手順は、メインフローの中に存在している。
- 任意選択の手順はメインフローの外部に配置され、適切な背景や段階的表示を使用してできる限り目立たなくされている。
- 視線の通り道上で、よく使用される要素はあまり使用されない要素の前に表示されている。
- ユーザーは常に、次の操作を把握していて、タスクフロー内に予期しない飛び越しや中断がない。

間違った例:



この例では、ユーザーには次の操作がわかりません。タスクフロー内に予期しない飛び越しや中断があります。

正しい例:



この例では、UI要素の提示方法に、タスクを実行する手順が反映されています。

#### グループ化

論理的に関連する UI要素が明確な視覚的関連性を持って示されていれば、レイアウトで“グループ化”が行われています。個別のアイテムを対象とするよりも、関連アイテムのグループを対象とした方が楽に理解して注目できるので、グループは重要です。グループを使用すると、レイアウトの表示がシンプルになり、構成を理解しやすくなります。

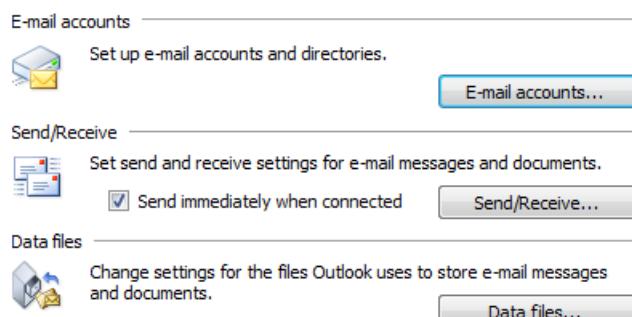
以下の方法でグループ化を表現できます(下に行くほどグループ化が強調されます)。

- レイアウト。コントロールが互いに関連する場合は近くに並べて配置し、関連しない場合は間隔を空けます。



この例では、レイアウトだけを使用してコントロールの関連性を示しています。

- 区切り記号。区切り記号とは、コントロールのグループをまとめる横線や縦線のことです。区切り記号を使用すると、外観がすっきりします。グループボックスとは異なり、区切り記号は領域いっぱいにすると最も効果的です。



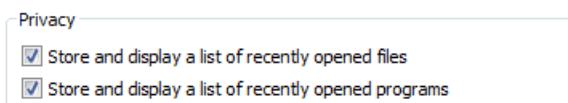
この例では、ラベル付けされた区切り記号を使用してコントロールの関連性を示しています。

- アグリゲーター。アグリゲーターは、関連の強いコントロールに対して視覚的な関連性を表現するグラフィックです。



この例では、境界線アグリゲーターを使用して、コントロール間の関連性を強調し、8つではなく単一のコントロールであるかのように表示しています。

- グループ ボックス。グループ ボックスとは、関連するコントロールのセットを囲むラベル付きの四角い枠のことです。



この例では、関連するコントロールのセットがグループ ボックス内に配置され、ラベルが付けられています。

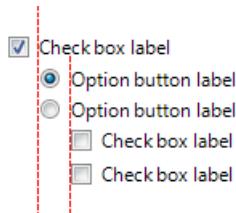
- 背景。背景を使用して、異なる種類のコンテンツを強調したり、逆に目立たなくしたりします。



この例では、コントロールパネルの作業ウィンドウが、関連するタスクおよびコントロールパネル アイテムのグループ化に使用されています。

見た目が煩雑にならないようにするには、グループ化の効果がある方法の中で、できるだけ軽快な外観のものを採用するのが最適です。詳細については、「[グループ ボックス](#)」、「[タブ](#)」、「[区切り記号](#)」、「[背景](#)」を参照してください。

グループ化の方法にかかわらず、インデントを使用して、グループ内のコントロールの関連性を示すことができます。対等なコントロールどうしは、左揃えにする必要があります。また、依存コントロールには 12 DLU または 18 相対ピクセルのインデントを設定します。



依存コントロールは 12 DLU または 18 相対ピクセルのインデントが設定されています。この間隔は、チェック ボックスおよびラジオ ボタンからそれぞれのラベルまでの間隔と同じになります。

レイアウトで適切なグループ化が行われているかどうかは、以下の点に基づいて判断します。

- ウィンドウまたはページ内のグループは 7 つ以内である。
- 各グループの目的が明確である。
- 各グループ内のコントロールの関連性、特にコントロールの依存関係が明確である。

- ・ グループ化でコンテンツが単純化されており、複雑化されていない。

## 配置

ここでいう配置とは、位置を調整して UI 要素を置くことです。配置は、コンテンツの流し読みのしやすさや、ユーザーが感じる視覚的な複雑さに影響を与えるので、重要です。

以下の目標に従って、配置を決定する必要があります。

- ・ 横方向に簡単に流し読みできるようにする。横方向に読んだときに、次にくる項目との間に不自然な隙間がなく、関連するものであることがわかるようにします。
- ・ 縦方向に簡単に流し読みできるようにする。視線をあまり横方向に動かすことなく、関連項目を縦方向に流し読みして、目的のものを見つけることができるようになります。
- ・ 視覚的な複雑さを最小限にする。不必要に縦揃え用グリッド線があると、レイアウトが複雑に見えます。

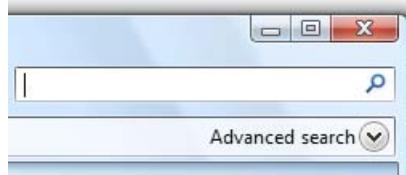
## 横位置

### 左揃え

読む順序が左から右の場合、左揃えはほとんどのコンテンツに効果的です。左揃えにすると、列形式のデータを縦方向に流し読みしやすくなります。

### 右揃え

右揃えは、数値データ、特に[数値データの列](#)に最適です。また、ウィンドウの右端に並ぶコントロールや[コミットボタン](#)にも適しています。



この例では、高度な検索用の段階的表示コントロールがウィンドウの右端にあり、右揃えとなっています。

### 中央揃え

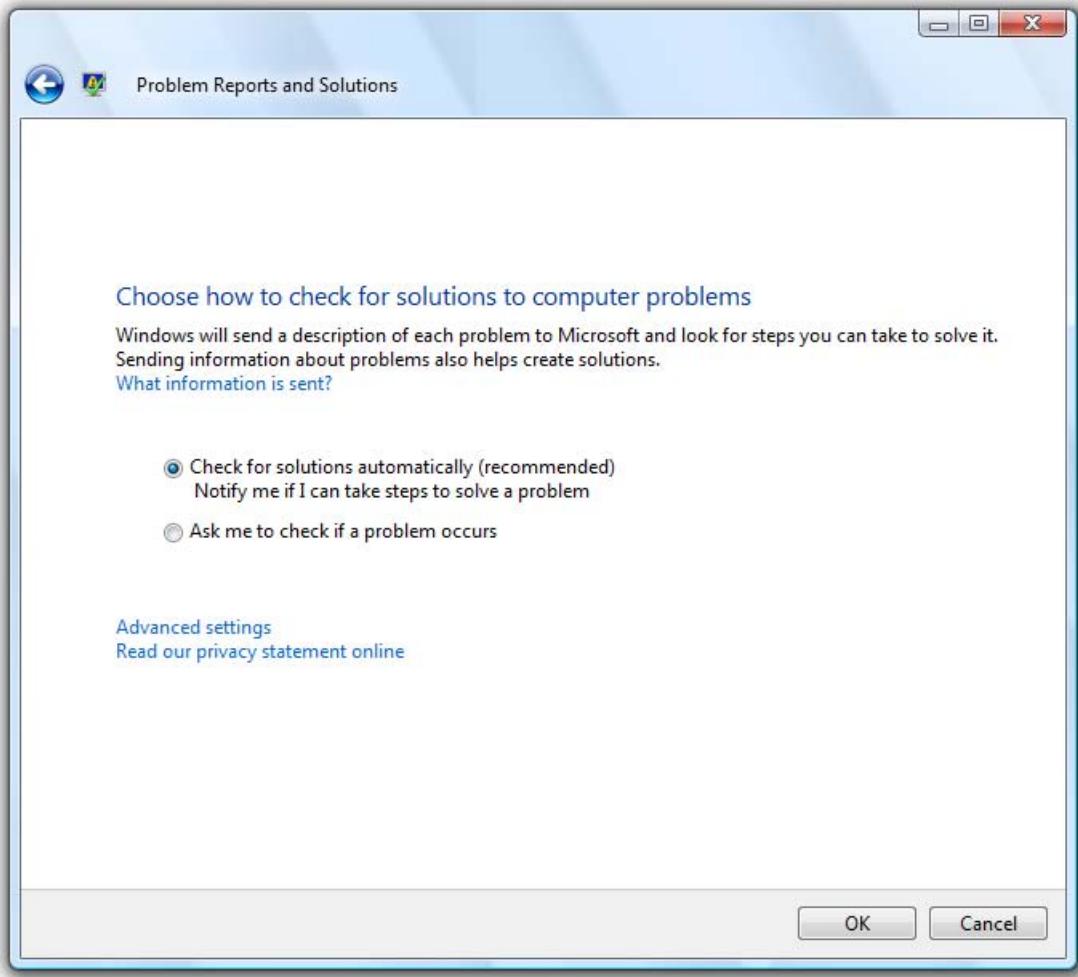
中央揃えは、左揃えや右揃えでは不適切な場合やバランスがとれない場合に最適です。



この例では、*Media Player* のコントロールが中央揃えになっており、バランスがとれています。

領域を埋めるためだけにウィンドウのコンテンツを中央揃えにしないでください。

間違った例:



この例では、サイズ変更可能なウィンドウ内の領域を埋めるために、コンテンツが不適切に中央揃えにされています。

#### 縦位置

#### 要素の上揃え

上から下に読む場合、上揃えはほとんどのコンテンツに効果的です。上揃えにすると、UI要素を横方向に流し読みしやすくなります。

#### テキストのベースライン揃え

テキスト付きのコントロールの縦位置を揃えるときは、テキストのベースラインを揃ると、横方向にスムーズな流れができます。

#### 正しい例:

HTML editor: Microsoft Office Word...

#### 間違った例:

HTML editor: Microsoft Office Word...

正しい例では、コントロールとそのラベルの縦位置がテキストのベースラインで揃えられています。

レイアウトの配置が適切かどうかは、以下の点に基づいて判断します。

- 横にも縦にも簡単に流し読みができる。
- 外観がシンプルである。

#### ラベルの配置

コントロール ラベルには一般的な配置のルールが適用されますが、その際に生じる一般的な問題に特別な注意を払う必要があります。ラベルを配置するときは、以下のことを目標にします。

- 縦方向に流し読みしたときに適切なコントロールが簡単に見つけられるようにする。
- 横方向に流し読みしたときにコントロールとラベルの関連が簡単にわかるようにする。
- 言語によってラベルの長さが変わるローカライズでの作業が簡単になるようにする。

- 長さの異なるラベルが混在している場合にも対応できるようする。
- テキストの表示が欠けないようにしながら、利用可能な領域を効率的に使用する。

全体的な目標は、ユーザーが探しそうな要素を最小限の視線の動きで見つけることができるようすることですが、ユーザーが探すコントロールの種類などはコンテキストによって異なります。

一般的なラベルの配置には以下のような4つのスタイルがあり、それぞれにメリットがあります。

- 左揃えのラベルをコントロールの上に配置
- 左揃えのラベルをコントロールの左に配置
- 左揃えのラベルをコントロールの左に配置、コントロールを左に寄せる
- 右揃えのラベルをコントロールの左に配置

#### 左揃えのラベルをコントロールの上に配置

このスタイルは、レイアウトがラベルの長さに依存しないのでローカライズが簡単ですが、縦方向の領域を最も消費します。

<b>Authors:</b>	<b>Manager:</b>
Jonathan	Jonathan
<b>Tags:</b>	<b>Company:</b>
vacation, desert	Microsoft
<b>Title:</b>	<b>Categories:</b>
Monument Valley	Family photos
<b>Subject:</b>	<b>Comments:</b>
Desert	With family

このスタイルでは、縦方向の領域が最も消費されますが、ローカライズは最も簡単です。ほとんどの対話型コントロールのラベル付けに適しています。

次のような場合に最適です。

- ラベル付けされたコントロールが対話型である場合(単なるテキストでない)。
- UIがローカライズされる場合。このスタイルでは、多くの場合、ラベルの長さが2倍さらには3倍になっても十分な領域を確保できます。
- UIに、Win32などの固定レイアウト テクノロジーが使用されている場合。
- コントロールの数が10個以下の場合。コントロールの数が10個を超えると、ラベルの流し読みが難しくなります。
- 縦方向に、ラベルが収まる十分な領域がある場合。
- レイアウトが列だけではなく、自由形式にする必要がある場合。

#### 左揃えのラベルをコントロールの左に配置

これは、縦方向に最も流し読みしやすいスタイルで、ラベルの長さが大きく異なる場合にも効果的です。ただし、ラベルとコントロールの関連付けがわかりにくくなります。このスタイルでは、必要に応じて複数行にまたがるラベルを使用できます。

<b>Authors:</b>	Jonathan	<b>Manager:</b>	Jonathan
<b>Tags:</b>	vacation, desert	<b>Company:</b>	Microsoft
<b>Title:</b>	Monument Valley	<b>Categories:</b>	Family photos
<b>Subject:</b>	Desert	<b>Comments:</b>	With family

このスタイルは便利ですが、実際には2列しかないものが4列のように見え、データの外観が複雑になっています。

次のような場合に最適です。

- ユーザーが特定のラベルを見つけるために縦方向に流し読みする可能性が高い場合。
- ユーザーがラベルとコントロールを左から右、上から下へと読み取る可能性が低い場合。
- 横方向に、ラベルが収まる十分な領域がある場合。
- ラベルの長さが大きく異なる場合。
- 複数のフォームなど、多くのコントロールが含まれている場合。
- 列がほとんどない場合。ラベルとコントロールは、それぞれ独立した列のように見えます。

#### 左揃えのラベルをコントロールの左に配置、コントロールを左に寄せる

このスタイルでは、縦方向にラベルを流し読みしてから、ラベルから横方向にコントロールをたどりやすくなります。領域の面では効率的ですが、コントロールの縦方向の流し読みは難しくなります。領域を最大限に活用するために、コントロールは右揃えにされます。

**Authors:** Jonathan

**Tags:** vacation, desert

**Title:** Monument Valley

**Subject:** Desert

**Manager:** Jonathan

**Company:** Microsoft

**Categories:** Family photos

**Comments:** With family

このスタイルは、コンパクトで読みやすい一方、コントロールの縦方向の流し読みは難しくなっています。

次のような場合に最適です。

- UI で Windows Presentation Foundation などのさまざまなレイアウト テクノロジーが使用されている場合。
- ユーザーが特定のラベルを見つけるために縦方向に流し読みする可能性が高い場合。
- ユーザーがラベルとコントロールを左から右、上から下へと読み取る可能性が高い場合。
- ユーザーが縦方向にコントロールを流し読みする可能性が低い場合。
- コントロールのテキストの長さが大きく異なり、別のスタイルでは表示が欠ける可能性が高い場合。
- 読み取り専用のテキスト ボックスなど、読み取り専用のコントロールを使用する場合。その他のコントロールにこのスタイルを使用すると、まとまりなく見えます。ただし、コントロールをクリック後に編集可能にする場合はこの限りではありません。
- 列が多くても、列内にコントロールがほとんどない場合。

右揃えのラベルをコントロールの左に配置

これは、横方向に読み取る際に、ラベルとコントロールの関連付けが最もわかりやすいスタイルです。ただし、ラベルの縦方向の流し読みは難しく、またラベルの長さが大きく異なる場合は不適切です。

**Authors:** Jonathan

**Tags:** vacation, desert

**Title:** Monument Valley

**Subject:** Desert

**Manager:** Jonathan

**Company:** Microsoft

**Categories:** Family photos

**Comments:** With family

このスタイルでは、コントロールの縦方向の流し読みは簡単にできますが、ラベルの縦方向の流し読みは難しくなります。

次のような場合に最適です。

- ユーザーがラベルとコントロールを左から右、上から下へと読み取る可能性が高い場合。
- 以下のような状況のため、ユーザーが特定のラベルを見つけるために縦方向に流し読みする可能性が低い場合。
  - コントロールがほとんど存在しない。
  - 一般的なラベルが使用されている。
  - コントロールが何を示しているか一目瞭然であり、また、空白のままで表示されることがほとんどない(空白のままにされることを避けるために既定値が設定されているなど)。
- 横方向に、ラベルが収まる十分な領域がある場合。
- ラベルの長さに大きな違いがない場合。
- 列が多い場合。ラベルとコントロールは、まとめて 1 列に見えます。

これらのスタイルのいずれかを採用する前に、さらに以下の 2 つの要因を検討します。

- プログラム全体で一貫して使用できるスタイルを優先します。
- 左揃えのラベルをコントロールの上または左に配置するスタイルは、最もよく使われるスタイルであり、優先的に使用する必要があります。

## バランス

UI 画面にコンテンツが平均的に分散して表示されているウィンドウまたはページは、バランスがとれています。バランスのとれたレイアウトは、たとえ UI 画面の外観上の重みが物理的な重みになったとしても、傾くことはありません。

バランスをとる上で最もよくある問題は、ウィンドウまたはページの左側のコンテンツが多すぎることです。次のような方法でバランスをとることができます。

- 左側に右側より大きな余白を使用します。
- タスクの完了に使用する UI 要素を右側に配置します。
- タスク全体で使用する UI 要素を中央に配置します。
- サイズ変更可能なコントロールや複数行にまたがるコントロールを長くします。
- 中央揃えは、その効果を十分検討した上で使用します。



このウィザードページのレイアウトはバランスがよくとれています。バランスを向上させるため、右の余白より左の余白が大きくなっています。

これらの手法でバランスがとれない場合は、ウィンドウまたはページの幅を狭くして、コンテンツの釣り合いをとることを検討します。

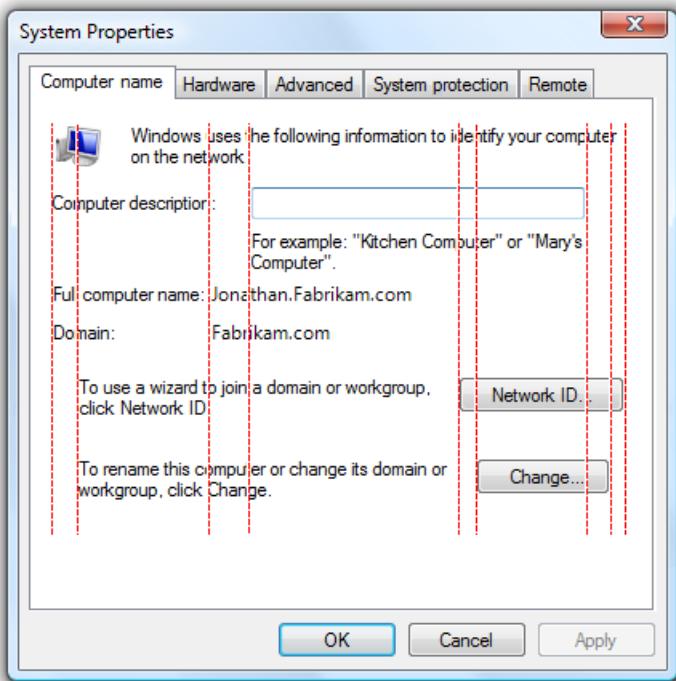
サイズ変更可能なUI画面では、バランスをとるためにコンテンツを中央揃えにしないでください。代わりに、左上の始点を固定して維持し、UI画面領域の最大サイズを定義して、その領域内でコンテンツのバランスをとります。

#### グリッド

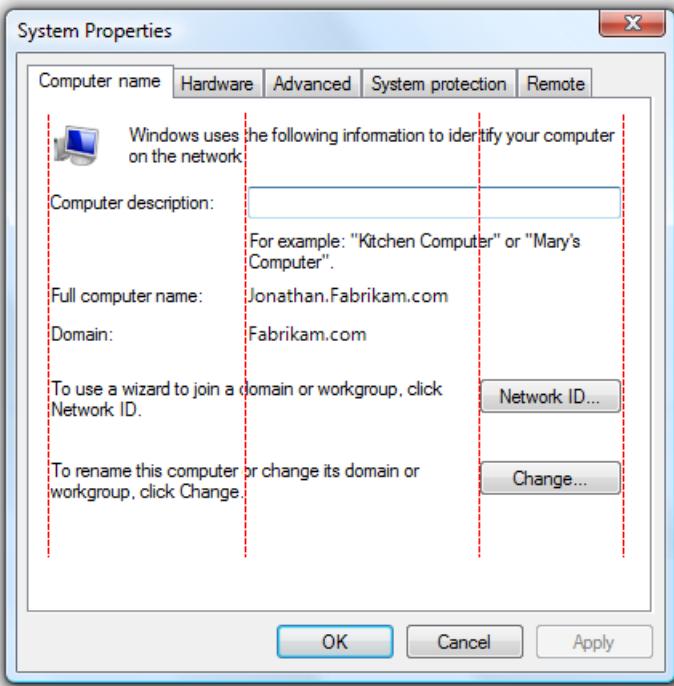
グリッドとは、配置の際の基盤となる機構ですが、ユーザーには見えません。グリッドは対照型にもできますが、非対照型グリッドの方が便利です。単一のウィンドウ/ページにグリッドを使用すると、画面内のコンテンツの整理に役立ちます。各画面に同じグリッドを使用すれば、各画面のレイアウトに一貫性を持たせることができます。

グリッド線の数は、外観の複雑さに影響します。グリッド線の少ないレイアウトは、グリッド線の多いレイアウトよりも単純に見えます。

複雑に見える例:



単純に見える例:



不必要的グリッド線は、外観を複雑にします。

レイアウトでグリッドが効果的に使用されているかどうかは、以下の点に基づいて判断します。

- 類似のコンテンツや機能を含むウィンドウ/ページのレイアウトが似ている。
- 繰り返し使用されるデザイン要素は、ウィンドウ/ページの同じような位置に表示されている。
- 縦および横のグリッド線に不要なものがいる。

視覚的な単純さ

視覚的な単純さが備わっていると、レイアウトが必要以上に複雑化されていないと感じます。

レイアウトが視覚的に単純かどうかは、以下の点に基づいて判断できます。

- 不必要的ウィンドウ クロムの層が除去されている。
- 最大 7 つまでの、識別が簡単なグループを使用して、コンテンツが提示されている。
- グループ ボックスではなく、軽快な外觀を持つレイアウトや区切り記号などでグループ化されている。
- 副コマンドにはコマンド ボタンではなくリンクを使用し、選択肢にはリストではなくドロップダウン リストを使用するなど、軽量コントロールが使用されている。
- 縦および横のグリッド線の数が削減されている。
- たとえば、画面上で使用するコマンド ボタンの幅を 2 種類までにとどめるなど、コントロール サイズの数が削減されている。
- 段階的表示が使用されており、必要になるまで UI 要素が非表示になっている。
- ウィンドウ/ページが窮屈にならない、十分な領域がある。
- ウィンドウやコントロールは適切なサイズに設定され、不要なスクロールが省かれている。
- 単一のフォントが使用され、そのサイズおよび色の数は少なく抑えられている。

原則的には、UI の効果を損ねることなく削除できるレイアウト要素があれば、ほとんどの場合、その要素を削除する方が適切です。

## ガイドライン

### 画面の解像度と dpi

- Windows の最小有効解像度 (800 × 600 ピクセル) をサポートします。セーフ モードで動作させる必要がある重要な UI では、640 × 480 ピクセルの有効解像度をサポートします。タスク バーと共に表示するウィンドウについては、縦方向に 48 相対ピクセル 短くして、タスク バー用の領域を確保します。
- サイズ変更可能なウィンドウのレイアウトを 1024 × 768 ピクセルの有効解像度に最適化します。より低い解像度の場合にも機能するように、これらのウィンドウのサイズが自動的に変更されるようにします。
- 96 dpi (ドット/インチ) の 800 × 600 ピクセル モード、120 dpi の 1024 × 768 ピクセル モード、144 dpi の 1280 × 1024 ピクセル モードで、必ずウィンドウをテストします。コントロール、テキスト、ウィンドウなどが切り詰められていないか、アイコンやビットマップが引き伸ばされたりしていないかなど、レイアウト上の問題をチェックします。
- タッチPC やモバイルPC 向けシナリオのプログラムでは、120 dpi に最適化します。現在、タッチPC やモバイルPC では高dpi 画面が普及しています。

## ウィンドウ サイズ

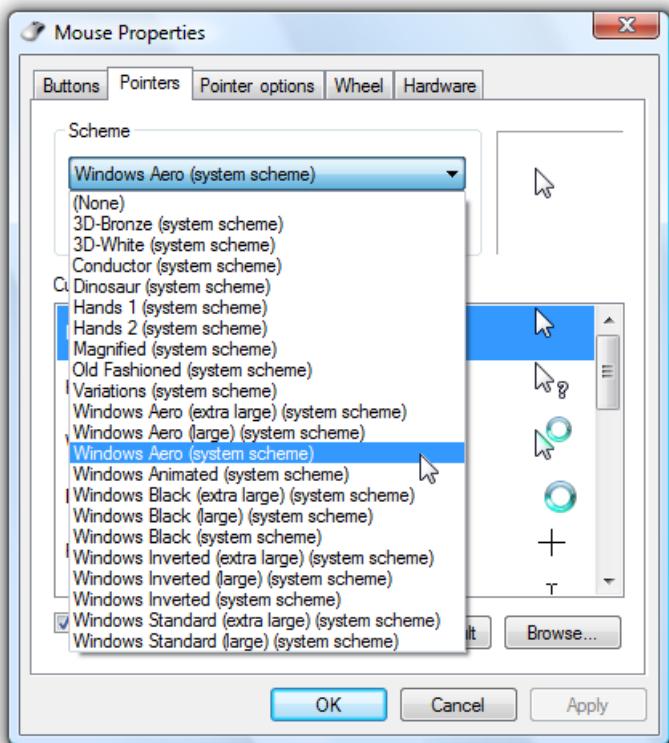
- ・コンテンツに適した既定のウィンドウ サイズを選択します。画面を効率的に使用できる場合は、ウィンドウの初期サイズを大きくします。
- ・バランスのとれた、縦と横のアスペクト比を使用します。望ましいアスペクト比は 3:5 から 5:3 ですが、エラーや警告のメッセージ ダイアログ ボックスには 1:3 のアスペクト比を使用できます。
- ・サイズ変更可能なウィンドウを使用するとスクロールバーの使用やデータの切り詰めを避けることができる場合は、常にサイズ変更可能なウィンドウを使用します。動的コンテンツ、ドキュメント、イメージ、リスト、ツリーを含むウィンドウは、サイズ変更のメリットが最も得られるウィンドウです。
- ・テキスト ドキュメントでは、1 行の文字数は半角換算で 80 文字以内になるようにします。これより長いとテキストが読みにくくなります(句読点や空白文字も含めてカウントします)。
- ・固定サイズのウィンドウの場合:
  - ・固定サイズのウィンドウの場合、全体が表示され、[作業領域](#)に収まるサイズにする必要があります。
- ・サイズ変更可能なウィンドウの場合:
  - ・サイズ変更可能なウィンドウは高解像度に最適化できますが、表示する際の実際の画面解像度に応じてサイズを縮小します。
  - ・ウィンドウのサイズを大きくするにつれて、より多くの情報が表示されていくようにする必要があります。少なくともウィンドウ領域の一部または 1 つのコントロールに、サイズ変更可能なコンテンツを含めるようにします。
  - ・ウィンドウのサイズを変更しても、コンテンツの左上の始点は固定したままにします。コンテンツのバランスをとるために、ウィンドウのサイズの変化に合わせて始点を移動しないでください。
  - ・コンテンツの幅が過剰に引き伸ばされるおそれがある場合は、コンテンツの最大サイズを設定します。コンテンツが引き伸ばされると扱いにくくなる場合は、ウィンドウのサイズが広げられても、コンテンツ エリアが最大幅を超えて広げられたり、コンテンツの始点が変更されたりしないようにします。最大幅を保持して左上の始点は固定します。
  - ・コンテンツが利用できなくなる下限のサイズがある場合は、ウィンドウの最小サイズを設定します。サイズ変更可能なコントロールに対しては、機能を維持できる最小サイズを設定します(リスト ビューが役目を果たすことができる最小の列幅など)。最小サイズでは、省略可能な UI 要素は完全に削除します。
  - ・小さいサイズでコンテンツを使用できるようにするために、提示方法の変更を検討します。



この例では、Windows Media Player が小さくされすぎて標準の形態を維持できなくなると、形態が変わることを示しています。

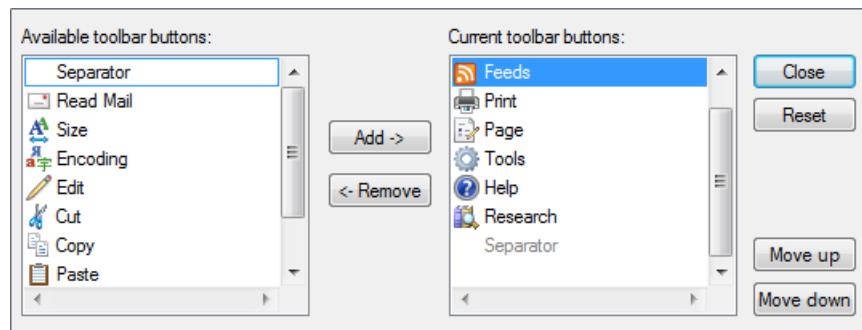
## コントロールのサイズ

- ・すべての対話型コントロールは、最小でも 16 × 16 相対ピクセルにする必要があります。こうすることで、Windows Tablet とタッチ テクノロジーをはじめとする、すべての入力デバイスで使用できます。
- ・データが切り詰められないように、コントロールのサイズを設定します。タスクを実行するために読む必要のあるデータを切り詰めないでください。リスト ビューの列は、データが切り詰められないサイズにします。
- ・不要なスクロールを省くことができるよう、コントロールのサイズを設定します。コントロールを少し大きくするだけでスクロールバーが不要になるのであれば、コントロールを大きくします。垂直スクロールバーがほとんど存在せず、不要な水平スクロールバーがない状態が最適です。



この例では、スクロールバーが不要になるように、ドロップダウンリストのサイズが決められています。

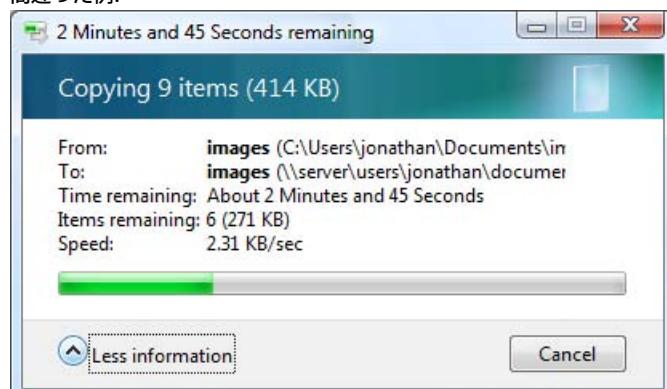
- UI 画面上のコントロールのサイズの種類を減らします。推奨されるコントロールの標準サイズを優先的に使用します。必要であれば、他のサイズもいくつか使用できますが、サイズに一貫性を持たせます。リストボックスとツリービューには単一の幅を使用し、コマンドボタンとドロップダウンリストに使用する幅は 3 種類以内にするように心掛けます。ただし、テキストボックスとコンボボックスの幅は、最長の入力や期待される入力の長さを示唆する長さにします。



この例では、リストボックスとコマンドボタンで 1 つのサイズが一貫して使用されています。

- テキストに基づいてサイズが決められるコントロールの場合は、ローカライズの対象となるすべてのテキストについて、30% (短いテキストの場合は最大 200%) の余白を追加します。このガイドラインでは、英語のテキストを使用してレイアウトが設計されていると仮定しています。また、このガイドラインで言及しているのは、ローカライズされる数字ではなく、ローカライズされるテキストであることにも注意してください。
- 静的テキストコントロール、チェックボックス、およびラジオボタンは、レイアウトに収まる最大幅に拡張します。こうすることで、可変長のテキストの切り詰めや、ローカライズの際の切り詰めを回避できます。

間違った例:



この例では、コントロールのテキストの表示が不必要に欠けています。

## コントロールの間隔

- コントロールどうしが接触していない場合は、コントロール間に少なくとも 3 DLU (5 相対ピクセル) の間隔を確保します。このようにしないと、ユーザーがコントロールの間の非アクティブな領域をクリックすることができます。非アクティブな領域をクリックしても効果や視覚的なフィードバックが得られないので、多くの場合ユーザーは何が悪かったのかがわかりません。

## 配置

- 左から右、上から下へと向かう(西欧文化圏の場合)自然なフローになるように、UI 画面内の UI 要素を並べます。UI 要素の配置は、要素間の関連性を伝えるものであり、タスクの実行手順を反映している必要があります。
- タスクを開始する UI 要素は、左上隅または中央上に配置します。ユーザーが最初に確認する必要がある UI 要素を、視覚的に最も強調します。
- タスクを完了する UI 要素は、右下隅に配置します。
- 関連する UI 要素はまとめて配置し、関連しない要素は分けます。
- 必要な手順は、メイン フローの中に配置します。
- 任意選択の手順はメイン フローの外に配置します。この際、適切な背景や段階的表示を使用してできる限り目立たないようにします。
- 視線の通り道上で、よく使用される要素を、あまり使用されない要素の前に配置します。

## フォーカス

- ユーザーが最初に確認する必要がある単一の UI 要素を、フォーカル ポイントに選択します。フォーカル ポイントは、ユーザーがすばやく見つけ出して理解する必要のある重要な部分に置きます。
- フォーカル ポイントは、左上隅や中央上に配置します。
- フォーカル ポイントを視覚的に最も強調する必要があります。目立つテキスト、既定の選択、初期入力フォーカスなどを使用して強調します。

## 配置

- 通常は、左揃えを使用します。
- 数値データ、特に数値データの列の場合には、右揃えを使用します。
- コミットボタンおよびウィンドウの右端に揃えられたコントロールには、右揃えを使用します。
- 中央揃えは、左揃えや右揃えでは不適切な場合やバランスがとれない場合に使用します。
- テキスト付きのコントロールの縦位置を揃えるときは、テキストのベースラインを揃ると、横方向にスムーズな流れができます。
- ラベルの配置については、「[デザインコンセプト](#)」の「[ラベルの配置](#)」を参照してください。

## アクセシビリティ

- レイアウトを、UI に関する重要な情報を伝える唯一の手段にはしないでください。視覚障碍のあるユーザーが、この提示方法を解釈できない場合があります。これ为了避免るために、たとえば、コントロール ラベルでその他のアイテムとの関連性がわかるようにします。
- コントロール ラベルに従属コントロールを埋め込んで、文や句を作成しないでください。このように意味を表現しても、これは単なるレイアウトなので、キーボード ナビゲーションやアクセシビリティ支援技術でうまく処理されません。さらに、文の構造は言語ごとに異なるので、この手法を使用するとローカライズできなくなります。

間違った例:

Create a computer account in the  domain

この例では、チェック ボックスのラベル内にテキスト ボックスが配置されているので不適切です。

正しい例:

Create a computer account in the following domain:

この例では、チェック ボックスのラベルの後にテキスト ボックスが配置されています。

- アクセシビリティに対応したグループ化を使用します。ウィンドウ、グループ ボックス、区切り記号、テキスト ラベル、アグリゲーターで定義されたグループは、アクセシビリティ製品で自動的に処理されます。配置と背景だけで定義されたグループは自動的には処理されないので、アクセシビリティのためにプログラム側でグループ定義する必要があります。

その他のガイドラインについては、「[アクセシビリティ](#)」を参照してください。

## 推奨されるサイズと間隔

### コントロールのサイズ

次の表に、一般的な UI 要素に推奨されるサイズ(幅 × 高さの形式、または単独の数値の場合は高さのみ)を示します(9 ポイント Segoe UI、96 dpi 時)。ローカライズの対象となるすべてのテキスト(数値以外)について、英文で最も長いテキストを基にして、ローカライズ用にさらに 30% (短いテキストには最大 200%) の余白を追加します。

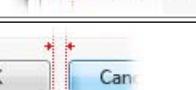
	コントロール	ダイアログ ユニット	相対ピクセル
	チェック	10	17

<input checked="" type="checkbox"/> Clock	ク ボッ クス		
<input type="checkbox"/> Volume			
11	コンボ ボック ス	最長アイテムの幅 + 30% × 14	最長アイテムの幅 + 30% × 23
	コマン ドボタ ン	50 × 14	75 × 23
 	コマン ドリン ク	25 (1 行の場合) または 35 (2 行の場合)	41 (1 行の場合) または 58 (2 行の場合)
Highest (32 bit) ▾	ドロップ ダウ ンリス ト	有効な最長データの幅 + 30% × 14	最長アイテムの幅 + 30% × 23
	リスト ボック ス	最長アイテムの幅 + 30% × 個別アイテムの高さの整数倍(最低 3 アイテム)	
   	リスト ビュ ー	データの切り詰めを回避できる列幅 × 個別アイテムの高さの整数倍	
	進行状 況バー	107 または 237 × 8	160 または 355 × 15
<input checked="" type="radio"/> Display as a link <input type="radio"/> Display as a menu	ラジオ ボタン	10	17
	スライ ダー	15	24
Select time zone:	テキス ト(スタ ティッ ク)	8	13
	テキス トボッ クス	最長の入力や予測される入力の幅 + 30% × 14 (1 行の場合) + 10 (追加 1 行ごと)	有効な最長データの幅 + 30% × 23 相対ピクセル (1 行の場合) + 16 (追加 1 行ごと)
 Desktop Jonathan Public Computer Local Disk (C:)	ツリー ビュ ー	最長アイテムの幅 + 30% × 個別アイテムの高さの整数倍(最低 5 アイテム)	

## 間隔

次の表に、一般的な UI 要素に推奨される間隔を示します (9 ポイント Segoe UI、96 dpi 時)。

要素	ダイアログ ユニット	相対ピクセル	
ダイアログ ボックスの余白	4 辺とも 7	4 辺とも 11	
* Label: 	テキスト ラベルとそれに関連付けられているコントロール(テキスト ボックス、リスト ボックスなど)との間隔	3	5
*  Check box *  Check box	関連するコントロールどうしの間隔	4	7
	関連しないコントロールどうしの間隔	7	11

			
	グループボックス内の最初のコントロール	グループボックスの一一番上から 11、グループボックスのタイトルと縦位置を揃える	グループボックスの一一番上から 16、グループボックスのタイトルと縦位置を揃える
	グループボックス内のコントロールどうしの間隔	4	7
	横または縦に並んだボタンどうしの間隔	4	7
	グループボックス内の最後のコントロール	グループボックスの一一番下から 7	グループボックスの一一番下から 11
	グループボックスの左端からの間隔	6	9
	コントロールに隣接するテキストラベル	コントロールの一番上から 3	コントロールの一番上から 5
	テキストの段落どうしの間隔	7	11
	対話型コントロールどうしの最小間隔	3 または間隔なし	5 または間隔なし
	非対話型コントロールと他のコントロールとの最小間隔	2	3

## レイアウト メトリック

### レイアウト

#### デバイスに依存しないレイアウト

フォントの書体やサイズ、解像度 (dpi)、ディスプレイ、グラフィック アダプターに関係なく、意図したとおりに表示されるレイアウトは、デバイスに依存しないレイアウトです。問題を理解するために、まずはダイアログ ボックスのレイアウトが物理的なピクセルに基づいている場合の動作を検討してみます。9 ポイントの Segoe UI を使用した次のダイアログ ボックスのレイアウトは、96 dpi では設計者が意図したとおりに表示されます。



9 ポイントの Segoe UI を使用した物理ピクセルベースのレイアウトのダイアログ ボックスを 96 dpi で表示した場合

しかし、120 dpi で表示するとこのダイアログ ボックスは小さくなり、拡大しなければテキストが読みづらくなります。



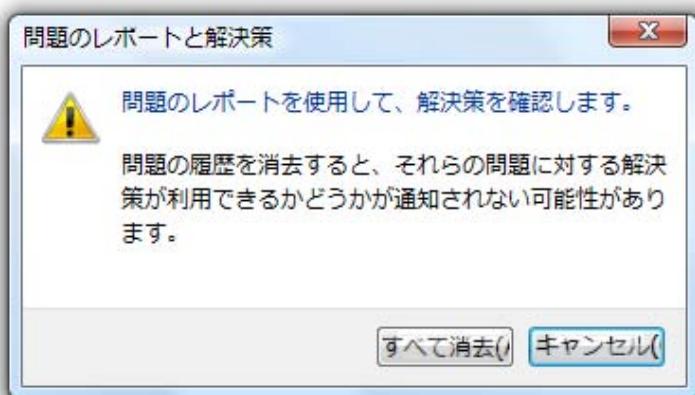
9 ポイントの Segoe UI を使用した物理ピクセルベースのレイアウトの同じダイアログ ボックスを、120 dpi で表示した場合

600 dpi のプリンターで印刷すると、拡大しなければまったく読み取れなくなります。



9 ポイントの Segoe UI を使用した物理ピクセルベースのレイアウトの同じダイアログ ボックスを、600 dpi で表示した場合

また、メイリオで表示すると、次のようにテキストがコントロール内に収まらなくなります。

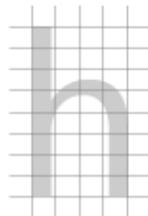


ピクセルベースのレイアウトの同じダイアログ ボックスを、9 ポイントのメイリオを使用して表示した場合

これらの例からわかるることは、デバイスに依存しないレイアウトにするには、テキストの読みやすさが他の表示パラメーターに影響されないようにする必要があるということです。そのため、ダイアログ ボックスのレイアウトとサイズは、テキストとの比例関係を維持する必要があります。Microsoft® Windows® はで従来から、ダイアログ ユニットを使用することによってこうした目標を達成しています。

## ダイアログ ユニット

ダイアログ ユニット (DLU) は、デバイス非依存のメトリックです。DLU では、水平方向の 1 ダイアログ ユニットが、使用中のフォントの平均文字幅の 4 分の 1 に相当します。また、垂直方向の 1 ダイアログ ユニットは、使用中のフォントの文字の高さの 8 分の 1 に相当します。文字の高さは幅の約 2 倍なので、水平方向の DLU は垂直方向の DLU とほぼ同じ大きさですが、DLU は正方形のユニットではないことを理解しておくことが重要です。



ダイアログ ユニットは、現在のフォントの平均文字幅の 4 分の 1、文字の高さの 8 分の 1 に相当します。

DLU は、Win32 リソース ファイルでのサイズ変更と配置に使用されます。リソース ファイル内で、各ダイアログ ボックス テンプレートのフォントが定義されており、DLU を決定する際に使用されます。したがって、複数のフォントを単一のプログラムで使用しても問題ありません。

## 相対ピクセル

前に示したように、物理ピクセルはデバイスに依存するメトリックですが、わかりやすく使いやすいというメリットがあります。デバイスに依存しないようにするために、新しい UI テクノロジーでは相対ピクセルが使用されています。

相対ピクセルはデバイスに依存しないメトリックであり、96 dpi では物理ピクセルでの表示と同じですが、他の dpi では比率を保って拡大縮小が行われます。そのため、たとえば、120 dpi での 1 相対ピクセルは、1.3125 物理ピクセルに相当します。相対ピクセルは正方形のユニットです。

相対ピクセルは、Windows Presentation Foundation (WPF) と WinForms で、ダイアログ ボックスのサイズ変更とレイアウトに使われています。

## 有効解像度

ほとんどの Windows ベースのコンピューターでは、従来より既定で 96 dpi となっていましたが、現在多くのラップトップでは、既定で 120 dpi 以上となっています。高 dpi が使用されることで、大きめのフォント、アイコン、グラフィックなど、より多くの物理ピクセルで描画した再現性の高い UI 要素が Windows で使用されるようになっています。結果として、UI の表示により多くのピクセルが必要とな

るため、画面の有効解像度は低くなります。

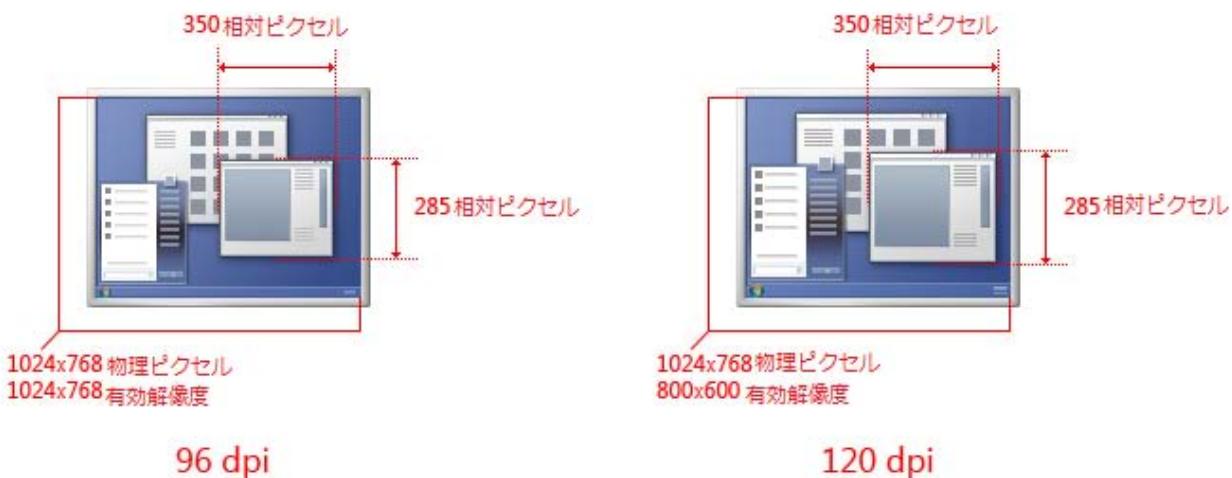
他の dpi を説明するために、画面解像度は、"有効解像度" という言葉で示されています。有効解像度は 96 dpi での解像度で、他の dpi に拡大縮小されます。次の表は、Windows の最小有効解像度 800 × 600 ピクセルおよび Windows の推奨有効解像度 1024 × 768 ピクセルでの一般的な dpi 設定の物理解像度を示しています。

dpi	最小物理解像度(ピクセル)	推奨される最小物理解像度(ピクセル)
96 dpi (100%)	800 × 600	1024 × 768
120 dpi (125%)	1024 × 768	1280 × 960
144 dpi (150%)	1200 × 900	1600 × 1200
192 dpi (200%)	1600 × 1200	2500 × 1600

これにより、Windows の最小有効解像度が 800 × 600 ピクセルの場合、Windows を 120 dpi でサポートするための最小物理解像度は、1024 × 768 ピクセルということになります。通常、有効解像度は次の式で計算できます。

$$\text{有効解像度} = \text{物理解像度} \times (96/\text{現在の dpi 設定})$$

画面のサイズを示す場合は有効解像度を使用し、レイアウトのサイズと間隔を示す場合は相対ピクセルを使用します。たとえば、Windows の最小有効解像度で画面に適した最大のウィンドウ サイズは、800 × 600 相対ピクセルです。これは、dpi の値に関わらず適用されます。



画面のサイズを示す場合は有効解像度を使用し、ウィンドウやコントロールを示す場合は相対ピクセルを使用します。

開発者向け情報: 詳細については、「[Writing High-DPI Win32 Applications](#)」あるいは「[DPI 対応の Win32 アプリケーションを記述する](#)」を参照してください。

## DLU から相対ピクセル(またはその逆)への変換

レイアウトとサイズを、DLU から相対ピクセル、またはその逆へ変換することが必要になる場合があります。しかし、DLU は正方形のユニットではないので、変換は軸に依存します。さらに、DLU は使用するフォントに基づいて決定されるので、変換はフォントにも依存します。

### 9 ポイント Segoe UI の場合の変換

水平方向の 1 DLU = 1.75 相対ピクセル

垂直方向の 1 DLU = 1.875 相対ピクセル

ダイアログ ユニット	水平方向のピクセル	垂直方向のピクセル
1	2 (1.75)	2 (1.875)
2	4 (3.5)	4 (3.75)
3	5 (5.25)	6 (5.625)

### 8 ポイント Tahoma の場合の変換

水平方向の 1 DLU = 1.50 相対ピクセル

垂直方向の 1 DLU = 1.625 相対ピクセル

ダイアログ ユニット	水平方向のピクセル	垂直方向のピクセル
1	1 (1.5)	2 (1.625)
2	3	3 (3.25)
3	4 (4.5)	5 (4.875)

4	7	7 (7.5)
5	9 (8.75)	9 (9.375)
6	10 (10.5)	11 (11.25)
7	12 (12.25)	13 (13.125)
8	14	15
9	16 (15.75)	17 (16.875)
10	17 (17.5)	19 (18.75)
11	19 (19.25)	21 (20.625)
12	21	22 (22.5)
13	23 (22.75)	24 (24.375)
14	24 (24.5)	26 (26.25)
15	26 (26.25)	28 (28.125)
16	28	30
17	30 (29.75)	32 (31.875)
18	31 (31.5)	34 (33.75)
19	33 (33.25)	36 (35.625)
20	35	37 (37.5)

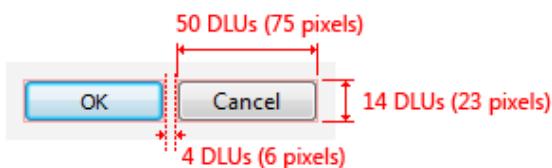
4	6	6 (6.5)
5	7 (7.5)	8 (8.125)
6	9	10 (9.75)
7	10 (10.5)	11 (11.375)
8	12	13
9	13 (13.5)	15 (14.625)
10	15	16 (16.25)
11	16 (16.5)	18 (17.875)
12	18	19 (19.5)
13	19 (19.5)	21 (21.125)
14	21	23 (22.75)
15	22 (22.5)	24 (24.375)
16	24	26
17	25 (25.5)	28 (27.625)
18	27	29 (29.25)
19	28 (28.5)	30 (30.875)
20	30	32 (32.5)

かっこ内の数字は正確な変換のための数値です。他のコンテキストとは異なり、.5は1に切り上げず、0に切り捨てられていることに注意してください。

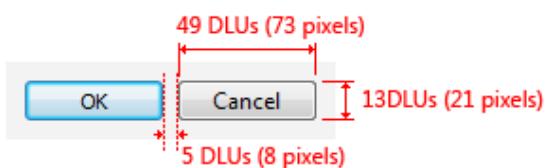
## コントロールとテキストの測定

グラフィック プログラムを使用してコントロールのサイズや間隔を測定するときに、数字が予測とは異なり混乱することがあります。たとえば、コマンド ボタンの標準サイズは 75 × 23 ピクセルですが、測定すると 73 × 21 ピクセルになります。この不一致は、一部のコントロールに、1 ピクセル幅の非表示の枠線があるためです。この枠線があることで、DLU を使用してレイアウトしたときに、コントロールどうしを密着させて配置することができます。コントロールのサイズと間隔を測定する場合は、非表示の枠線に注意します。

実際のコントロール サイズ：



表示サイズ：



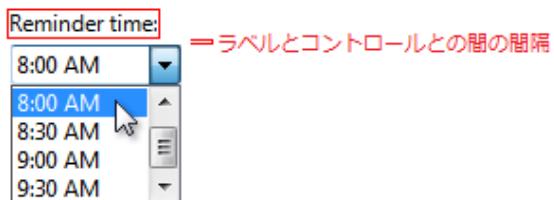
コントロールの外周に透明の1ピクセル  
ボーダーがあるので、表示サイズはコント  
ロールのサイズより小さい。

一部のコントロールには非表示の枠線があります。

グラフィック プログラムでテキストのサイズと間隔を測定する場合にも、同様の問題が発生します。テキスト コントロールの高さは、アセンダー、ディセンダー、分音記号、その間隔(レディングと呼ばれます)を含むテキストの高さで構成されます。そのため、表示されるテキストのサイズと間隔が、実際のサイズと間隔とは異なる場合があります。



フォントのアセンダー、ディセンダー、分音記号、レディング



警告: テキストは表示よりも大きなサイズに設定されています。

## ウィンドウ枠

デザインコンセプト

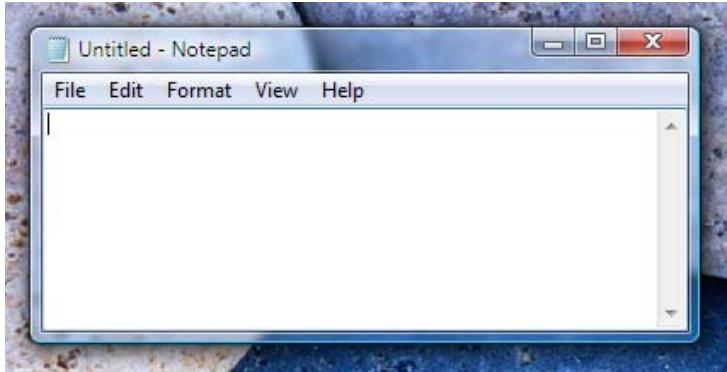
ガイドライン

ウィンドウ枠

全画面表示モード

グラス

ウィンドウ枠を使用すると、ユーザーはウィンドウを操作でき、タイトルとアイコンからコンテンツを特定できます。



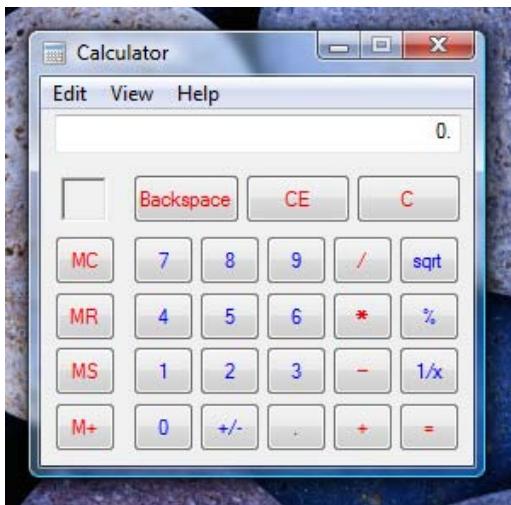
典型的なウィンドウ枠

注: [「ウィンドウの管理」および「ブランド化」に関するガイドライン](#)は、それぞれ別の項目として記載しています。

## デザイン コンセプト

グラス ウィンドウ枠

グラス ウィンドウ枠は Windows Vista® の特長の 1 つである新しい外観を表すものであり、美しさと軽快さの両方を実現しています。半透明のフレームがウィンドウに開放感を与え、圧迫感を少なくしているので、ユーザーは周囲のインターフェイスに気を取られることなくコンテンツと機能に集中できます。



グラス ウィンドウ枠

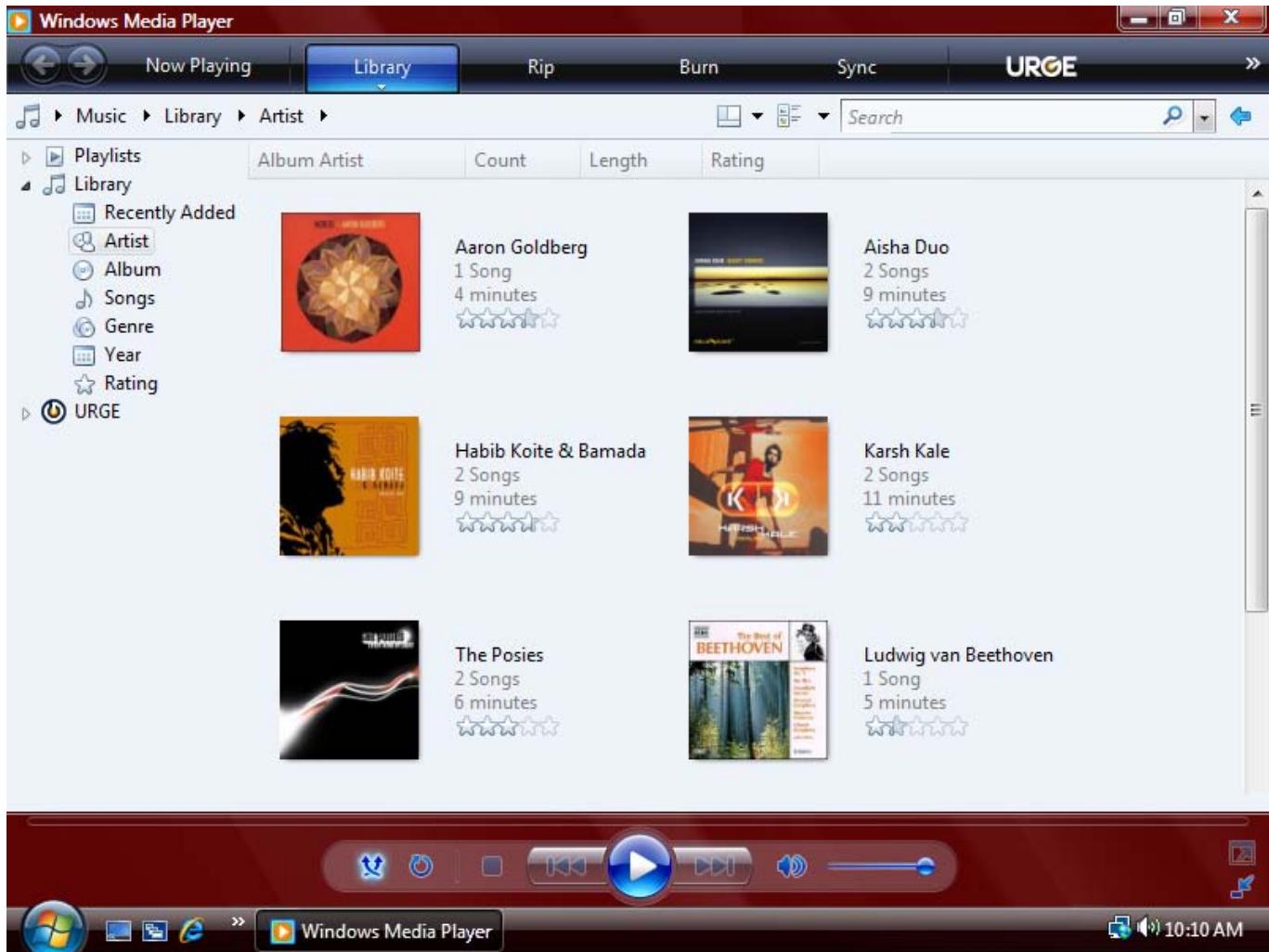
ウィンドウ内の、ウィンドウ枠に接する小さな領域で、グラスを戦略的に使用できます。この領域は、実際にはウィンドウのクライアント領域の一部であるものの、ウィンドウ枠の一部であるかのように見えます。



この例では、グラスが使用されたクライアント領域が、フレームの一部のようになっています。

#### 最大化されたフレーム

最大化されたフレームの場合は、外観が異なります。このようなフレームでは、グラスが使用される代わりに、白いテキストと不透明な背景(既定では黒)が使用されます。異なる外観が使用される理由は、ウインドウが画面全体に表示される場合には、グラスは必要ないためです。さらに、最大化されたウインドウは通常のウインドウとは違って移動やサイズ変更ができません。したがって、異なる外観を使用することで、ユーザーはウインドウが最大化されていると認識できます。

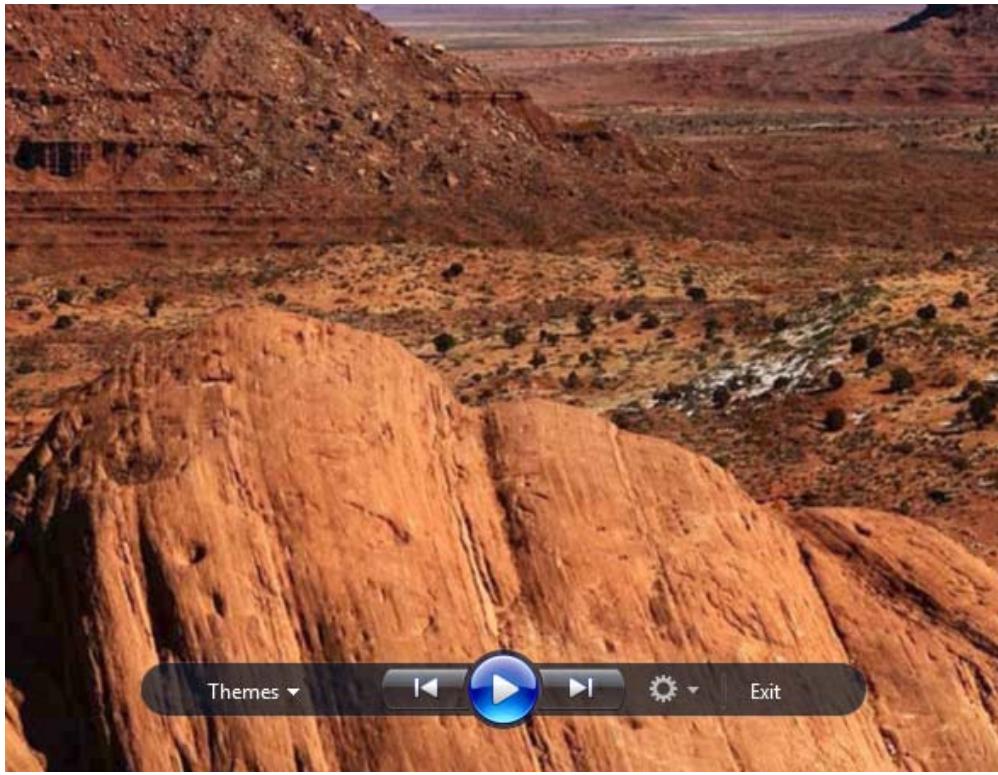


この例では、最大化された不透明なウィンドウ枠が示されています。

#### 非表示のフレーム

場合によっては、枠がまったくないのが最適なウィンドウ枠であることもあります。このようなフレームは、メディアプレーヤー、ゲーム、キオスクアプリケーションなど、別のプログラムとは一緒に使用されない没入型の全画面表示アプリケーションの[メイン ウィンドウ](#)で多く使用されます。

コンテンツビューアーでは、コンテンツの全画面表示を選択できると便利なことがあります。例としては、Windows® Internet Explorer®、Windows フォトギャラリー、Windows ムービーメーカー HD、Microsoft PowerPoint®、Microsoft Word などが挙げられます。



この例では、Windows フォト ギャラリーでコンテンツが全画面表示されています。

#### カスタム枠

ほとんどの Windows アプリケーションでは標準のウィンドウ枠を使用することが推奨されますが、ゲームやキオスク アプリケーションのような全画面表示の没入型スタンダロン アプリケーションでは、全画面表示されないウィンドウにカスタム枠を使用することが適している場合もあります。カスタム枠を使用する理由としては、[ブランド化](#)以外に、エクスペリエンス全体に独自性を与えることがあります。



カスタム枠は、ゲームのような全画面表示の没入型スタンダロン アプリケーションに適しています。

#### ガイドライン

##### ウィンドウ枠

- 標準のウィンドウ枠を使用します。
  - 例外: 全画面の没入型スタンダロン アプリケーションに独自性を与えるには、以下のことを検討します。
    - [メイン ウィンドウ](#)のウィンドウ枠を非表示にします。
    - [サブ ウィンドウ](#)にはカスタム枠を使用します。
    - カスタム枠の使用が適している場合は、フレーム自体が目立ちすぎない、軽快なデザインを選択します。

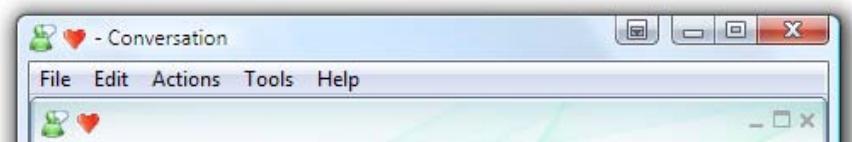
間違った例:



この例では、カスタム枠自体が目立ちすぎています。

- ウィンドウ枠上にコントロールを配置しないようにします。コントロールはウィンドウ枠ではなくウィンドウ内に配置します。

間違った例:



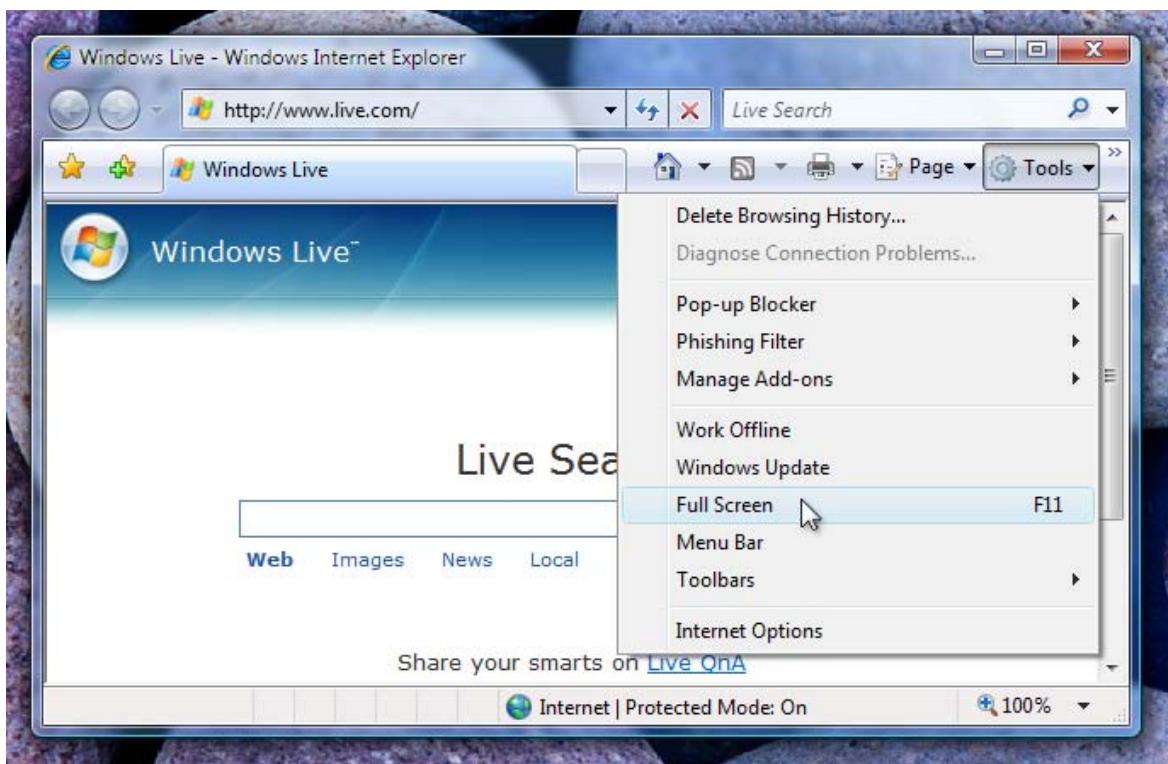
正しい例:



正しい例では、コントロールはウィンドウ枠ではなくクライアント領域内に配置されています。

全画面表示モード

- 全画面表示モードを選択できるプログラムで全画面表示モードを有効にするには、次のようにします。
  - メニューバーまたはツールバーに、モーダルな全画面表示コマンドを用意します。ユーザーがコマンドをクリックしたら、コマンドを選択状態にします。



この例では、全画面表示コマンドが標準のショートカットキーと共に表示されています。

- 全画面表示のショートカットキーには F11 キーを使用します。

- ツールバーがあり、全画面表示モードが共通して使用される場合は、"全画面表示"というツールヒント付きのグラフィックツールバー ボタンも用意します。



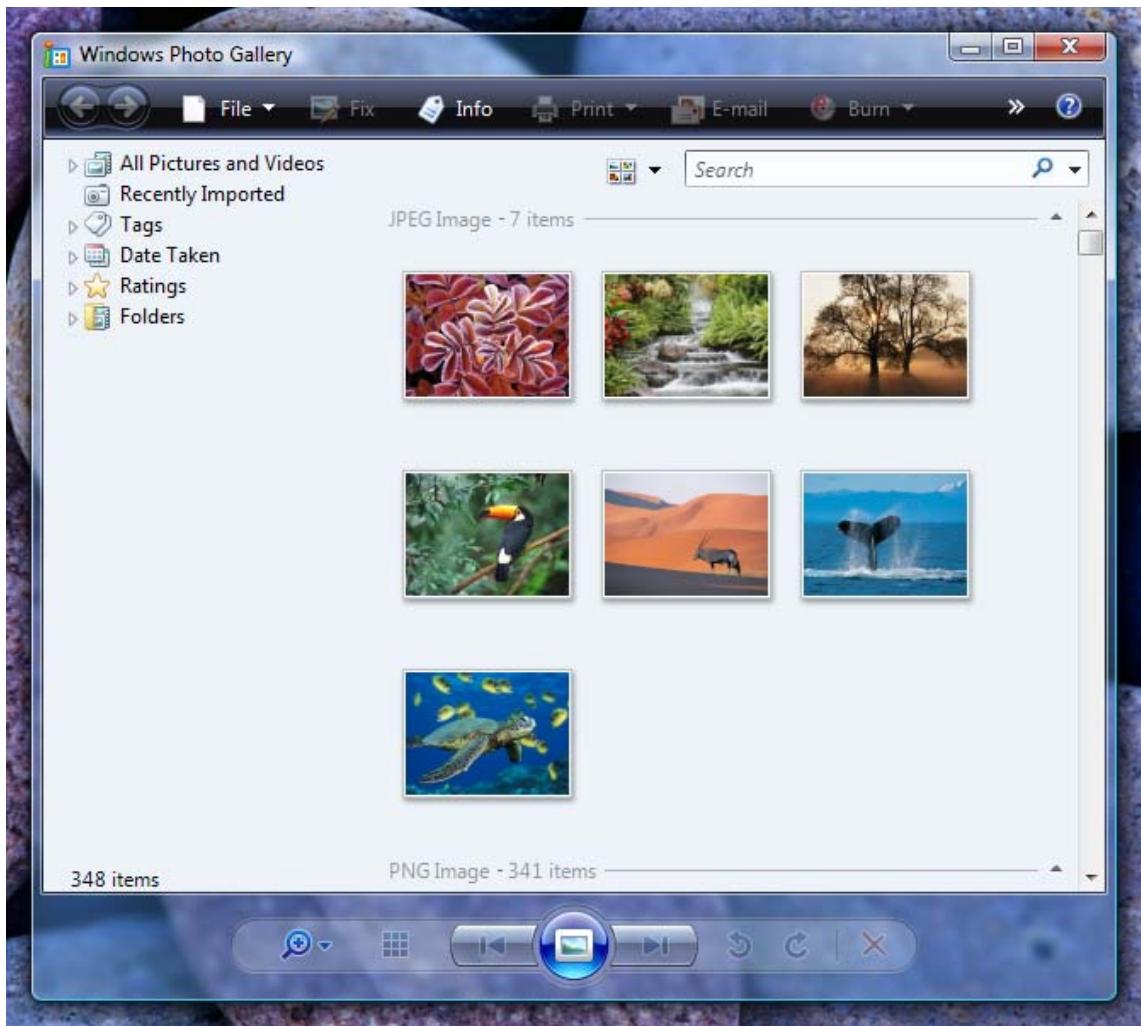
全画面表示ツールバー ボタンの例。

- 全画面表示モードから戻すには、次のようにします。
  - メニュー バーまたはツール バーに、モーダルな全画面表示コマンドを用意します。ユーザーがコマンドをクリックしたら、コマンドを選択解除の状態にします。
  - 全画面表示のショートカット キーには F11 キーを使用します。他に割り当てられていないければ、Esc キーと同じ目的で使用することもできます。

## グラス

Windows の標準のウィンドウ枠では自動的にグラスが使用されますが、ウィンドウ枠に接する領域でもグラスを使用することができます。

- ウィンドウ枠に接する、テキストを含まない小さい領域で、その効果を十分検討した上でグラスを使用するようにします。こうすることで、その領域が枠と一体化し、プログラムに簡素で軽快、かつまとまりのある外観を与えることができます。



この例では、グラスを使用して、コントロールではなくコンテンツにユーザーの注意を引き付けています。

- 通常のウィンドウ背景の方が魅力的であったり、使いやすかったりする状況では、グラスは使用しないでください。

正しい例:



この例では、*Alt* キーを押しながら *Tab* キーを押したときのウィンドウの外観が、グラスの使用によって軽快になっています。このウィンドウは、グラフィックと強調された 1 つのテキストラベルで構成されているため、グラスが効果を発揮しています。

間違った例:



この間違った例では、グラスを使用することでウィンドウが煩雑になっています。このような場合は、通常のウィンドウの背景を使用する方が適しています。

## フォント

デザインコンセプト  
使用パターン  
ガイドライン  
フォントと色  
属性

### メイリオと Meiryo UI

Microsoft Windows 上でユーザーがもっとも利用するユーザーインターフェイス (UI) 要素はテキストです。Segoe UI は欧文用の Windows のシステムフォントですが、日本語のテキスト表示にはメイリオが標準で使用されます。欧文用、和文用ともに既定のフォントサイズは9 ポイントです。

abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ  
あいうえおかきくけこさしすせそアイウエオカキクケコサシスセソ  
亜啞娃阿哀愛挨始達葵西龜惡握渥旭葦芦鰯梓庄斡扱宛姐虹飴綾鮎  
0123456789 () 「」！？、．。

#### メイリオ フォント

Windows 7 では新しいユーザーインターフェイスとしてリボンコントロールを採用しました。リボンコントロールはユーザーの直観的な操作を可能にするためにアイコンを多用しますが、それらのアイコンとともに表示されるテキスト領域は、通常のUIで確保されるテキスト領域に比べて必ずしも十分な広さがあるとは限りません。そうしたリボン上に表示するUIテキストに最適なのが Meiryo UI フォントです。

abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ  
あいうえおかきくけこさしすせそアイウエオカキクケコサシスセソ  
亜啞娃阿哀愛挨始達葵西龜惡握渥旭葦芦鰯梓庄斡扱宛姐虹飴綾鮎  
0123456789 () 「」！？、．。

#### Meiryo UI フォント

メイリオと Meiryo UI とはそれぞれ独立したフォントですが、いずれもClearTypeに最適化された可読性の高い日本語フォントです。メイリオの和文文字はすべて日本語組版で標準的ないわゆる全角幅でデザインされており、本文用としてもUI用としてもあらゆる世代に読みやすく設計されています。一方 Meiryo UI はその名に示すように、UIとして多用されるカタカナやひらがなに関して、狭い領域でも可読性を失うことがないように字面および字間、行間を調整し、Windows 7 の新しいUIプラットフォームであるリボンに最適化しています。

### Segoe UI

Segoe UI (発音は "シーゴ") は、Windows のシステム フォントです。標準のフォント サイズは、9 ポイントと大きくなりました。

abcdefghijklmnopqrstuvwxyz  
ABCDEFGHIJKLMNOPQRSTUVWXYZ

#### Segoe UI フォント

"Segoe UI" と "Segoe" は異なるフォントです。Segoe UI は、ユーザー インターフェイスの文字列に使用する Windows フォントで、Segoe は、印刷や広告用の資料作成に Microsoft とパートナーが使用するブランド フォントです。

Segoe UI はオープンな親しみやすい書体であり、Tahoma、Microsoft Sans Serif、Arial よりさらに読みやすい文字を提供します。Segoe UI には人間味のあるサンセリフの特長があります。たとえば、大文字はどれも広めの同じような幅になっている Helvetica と比べて、Segoe UI の E や S は狭く、幅に変化があります。また、小文字のアクセントや字形にも特徴があるほか、多くの工業的なサンセリフの斜体 "oblique" がローマン体を傾けたものであるのに対し、Segoe UI では真のイタリック体が使用されています。この書体は、画面でも印刷物でも同じ視覚効果が得られるように意図されたものです。個性を強調せず、奇をてらわないことで、人間味のあるサンセリフになるようにデザインされています。

Segoe UI は、Windows で既定で有効になっている ClearType に合わせて最適化されています。ClearType

が有効になっていると、Segoe UI は洗練された読みやすいフォントになります。ClearType が有効になつていないと、Segoe UI は辛うじて受け入れられるほどの表示にしかなりません。この点が、Segoe UI を使用するかどうかの見極めになります。

Segoe UI には、ラテン文字、ギリシャ文字、キリル文字、およびアラビア文字が含まれています。それ以外の文字セットや用途には、同じく ClearType に合わせて最適化された新しいフォントがあります。たとえば、日本語にはメイリオ、韓国語には Malgun Gothic、中国語(繁体字)には Microsoft JhengHei、中国語(簡体字)には Microsoft YaHei、ヘブライ語には Gisha、タイ語には Leelawadee があり、ドキュメント向けには ClearType コレクションのフォントがあります。

メイリオには、Verdana を基にしたラテン文字が含まれています。Malgun Gothic、Microsoft JhengHei および Microsoft YaHei には、Segoe UI のカスタマイズ版が使用されています。これらのフォントのイタリック体の使用はお勧めしません。Malgun Gothic、Microsoft JhengHei および Microsoft YaHei は、標準スタイルおよび太字スタイルのみ提供され、イタリック文字は標準スタイルを傾斜させて合成します。メイリオには真のイタリックとボールドイタリックが含まれていますが、これらのスタイルはラテン文字にのみ適用されます。日本語文字は、イタリックスタイルが適用されても標準スタイルのままになります。

以上の文字セットを使用するロケールをサポートするために、Segoe UI は[ローカライズ](#)のプロセス中に各ロケールに応じた正しいフォントに置き換えられます。

Windows ベースのプログラムに必要な Segoe UI およびその他の Microsoft フォントの配布用ライセンスについては、[Ascender](#) にお問い合わせください。

注: [スタイルとトーンおよびユーザーインターフェイスのテキスト](#)に関するガイドラインは、それぞれ別の項目として記載しています。

## デザイン コンセプト

### フォント、書体、ポイントサイズ、属性

従来の文字体裁の "フォント" とは、書体、ポイントサイズ、およびその他の属性の組み合わせを表すものです。"書体" とはフォントの外観のことであり、Segoe UI、Tahoma、Verdana、Arial などはすべて書体です。"ポイントサイズ" とはフォントのサイズであり、アセンダー上端からディセンダーワークまで長さから内部の間隔(内部レディング)を引いたものです。1 ポイントは 72 分の 1 インチ(約 0.35 mm)のサイズになります。この 2 つに加えて、フォントには太字や斜体などの "属性" を指定できます。

このトピックでもそうですが、"フォント" という言葉を "書体" の代わりに使用することがよくあります。しかし、厳密には Segoe UI はフォントではなく書体です。9 ポイント Segoe UI 標準、10 ポイント Segoe UI 太字のように、属性の組み合わせが固有のフォントになります。

### セリフとサンセリフ

書体はセリフまたはサンセリフのいずれかに属します。"セリフ" とは、フォントの文字ストロークの先端によく付いている小さな飾りのことです。"サンセリフ" はセリフのない書体です。

一般的に、ドキュメント内の本文にはセリフ フォントが好まれます。セリフは、ドキュメントに格式のある上品な感じを与えます。UI テキストには、すっきりした外観が必要であることと、コンピューターのモニターがより低解像度であることから、サンセリフの書体の方が適しています。

### コントラスト

テキストは、背景との明度の差が大きい場合に最も読みやすくなります。白い背景に黒いテキストを使用すると、コントラストが最も大きくなります。明るい背景に濃い色のテキストを使用する場合も、コントラストを大きくできます。この組み合わせは、プライマリ UI 画面に最適です。

暗い背景に薄い色のテキストを使用すると、適度なコントラストがとれます。明るい背景に濃い色のテキストを使用した場合ほどではありません。この組み合わせは、エクスプローラーの作業ウィンドウなど、プライマリ UI 画面ほど強調しないセカンダリ UI 画面に適しています。

ユーザーにテキストが必ず読まれるようにする場合は、明るい背景に濃い色のテキストを使用します。

### アフォーダンス

テキストには、用途を示すために次のような[アフォーダンス](#)を使用できます。

- ポインター。I字型のバー ("テキスト選択") ポインターは、テキストが選択可能なことを示します。一方、左向きの矢印 ("通常選択") ポインターは、テキストが選択可能でないことを示します。
- キャレット。テキストに入力フォーカスがあるときの、点滅する垂直のバーです。選択または編集可能なテキストの挿入または選択ポイントを示します。
- ボックス。テキストを囲むボックスは、テキストが編集可能であることを示します。表示を軽くするため、編集可能なテキストが選択されたときのみ動的にボックスを表示することもあります。
- 前景色。薄い灰色はテキストが無効であることを示します。灰色以外の色、特に青や紫は、テキストがリンクであることを示します。
- 背景色。薄い灰色の背景はテキストが読み取り専用である印象を与えますが、実際は、読み取り専用のテキストにはどのような背景色も使用できます。

以上のアフォーダンスを組み合わせることで、次の意味を表すことができます。

- 編集可能。テキストをボックスで囲み、テキスト選択のポインターと (入力フォーカス時に) キャレットを表示し、通常は背景を白にします。
- 読み取り専用、選択可能。テキストに、選択ポインターと (入力フォーカス時に) キャレットを表示します。
- 読み取り専用、選択不可。テキストに矢印ポインターを表示します。
- 無効。薄い灰色のテキストに矢印ポインターを表示します。背景を灰色にすることもあります。

従来的に、読み取り専用のテキストには灰色の背景が使用されていますが、必ずしも灰色の背景は必要ではありません。実際のところ、テキストブロックが大きい場合などは灰色の背景が望ましくないことがあります。灰色の背景は、テキストが無効である印象を与え、読む気を損なわせます。

#### アクセシビリティとシステム フォント、サイズ、色

障碍のある方がテキストを読めるようにするためのガイドラインは、常にシステム フォント、サイズ、色を使用してユーザーの状況に配慮するという、1つの単純なルールを守ることだけです。

##### 最も重要な点

常にシステム フォント、サイズ、色を使用して、ユーザーの状況に配慮します。

**開発者向け情報:** サイズを含むシステム フォントのプロパティは、GetThemeFont API 関数を使用したコードで確認できます。システム カラーは、GetThemeSysColor API 関数を使用して確認できます。

ユーザーのシステムのテーマがどのような設定になるかわからないため、次のことを守る必要があります。

- フォントの色と背景の色は、常にシステムのテーマ カラーの中で対照を成すようにします。RGB (赤、緑、青) の固定値に基づいて独自の色を作成することはしません。
- システムのテキストの色は、常に対応する背景色と一緒に使用します。たとえば、テキストの色に COLOR\_STATICTEXT を選択する場合は、その背景色にも COLOR\_STATIC を選択する必要があります。
- 新しいフォントを作成する場合は、常にシステム フォントのプロポーショナル サイズ フォントを基準にします。システム フォントのメトリクスから、太字にしたり、斜体にしたり、大きめまたは小さめのフォントを作成できます。

ユーザーの設定に配慮したプログラムになっていることを確認する簡単な方法は、異なるフォント サイズやコントラストの大きい配色でテストすることです。すべてのテキストは、正しくサイズ変更され、選択した配色で正しく表示される必要があります。

#### 使用パターン

テキストにはいくつかの使用パターンがあります。

タイトルバーのテキスト  
Title bar text  
ウィンドウを識別する  
タイトルバー上のテキスト。

メイン指示テキスト  
Main instructions  
ページ、ウィンドウ、  
ダイアログ ボックス  
などでの操作を説明する  
テキスト。

サブ指示テキスト  
ページ、ウィンドウ、  
ダイアログ ボックス  
などでの操作を説明す  
る補足テキスト。

Secondary instructions

標準テキスト  
ユーザー インター  
フェイスに表示される  
通常(読み取り専用)の  
テキスト。

Normal text

強調テキスト  
太字のテキストは、テ  
キストの理解を助けた  
り必読のテキストに注  
目させるために使用し  
ます。斜体のテキスト  
は、テキストを(引用  
符なしで)文字どおり  
に引用したり特定語句  
を強調するために使用  
します。

Emphasized text

編集可能なテキスト  
ユーザーが編集でき  
る、ボックス囲みのテ  
キスト。表示を軽くす  
るために、編集可能なテ  
キストが選択されたと  
きのみボックスを表示  
することもあります。

Editable text

淡色表示のテキスト  
無効にされたコント  
ロールのラベルなど、  
現在のコンテキストに  
該当しないテキスト。  
淡色表示のテキスト  
は、ユーザーが(通常)  
テキストを読まなくて  
もよいことを示しま  
す。

Disabled text

リンク  
別のページ、ウィンド  
ウ、ヘルプ トピック  
への移動、またはコマ  
ンドの開始のためのテ  
キスト。

Link

Links (Hover)

ドキュメント テキ  
スト  
(UI テキストに対して)  
ドキュメント内で使用  
するテキスト。

Document text

ドキュメント見出し  
ドキュメント内で見出  
しとして使用するテキ  
スト。

Document headings

## ガイドライン

### フォントと色

- Windows Vista と Windows 7 の既定のフォントと色は次のとおりです。

パターン	テーマ シンボ	フォント、色
------	---------	--------

	ル	
Title bar text	CaptionFont	9 ポイント黒 (#000000) Segoe UI
Main instructions	MainInstruction	12 ポイント青 (#003399) Segoe UI
Secondary instructions	Instruction	9 ポイント黒 (#000000) Segoe UI
Normal text	BodyText	9 ポイント黒 (#000000) Segoe UI
Emphasized text	BodyText	9 ポイント黒 (#000000) Segoe UI、太字または斜体
Editable text	BodyText	9 ポイント黒 (#000000) Segoe UI、ボックス囲み
Disabled text	Disabled	9 ポイント濃い灰色 (#323232) Segoe UI
Link	HyperLinkText	9 ポイント青 (#0066CC) Segoe UI
Links (Hover)	Hot	9 ポイント薄い青 (#3399FF) Segoe UI
Document text	(なし)	9 ポイント黒 (#000000) Calibri
Document headings	(なし)	17 ポイント黒 (#000000) Calibri

- UI テクノロジーとターゲットとする Windows のバージョンに基づいて、フォントを選択し、最適なウィンドウ レイアウトにします。

UI テクノロジー	ターゲットとする Windows のバージョン	使用するフォントと最適化
Windows Presentation Foundation	すべて	WPF のテーマ パーツを使用します。
Win32 または WinForms	Windows Vista 以降	適切な Segoe UI フォントを使用します。
	拡張可能なコンポーネントまたは Vista 以前の Windows	Windows® XP および Windows 2000 をターゲットとする場合は、8 ポイント MS Shell Dlg 2 擬似フォントを使用します。このフォントは Tahoma にマッピングされます。 Windows の以前のバージョンをターゲットとする場合は、8 ポイント MS Shell Dlg 擬似フォントを使用します。このフォントは、Windows 2000 および Windows XP では Tahoma に、Windows 95、Windows 98、Windows Millennium Edition、Windows NT 4.0 では MS Sans Serif にそれぞれマッピングされます。

#### 開発者向け情報:

- 固定レイアウトを使用する要素 (Windows ダイアログ テンプレート、WinForms など) には、前に挙げた表から適切なフォントを固定値で使用します。
- 動的なレイアウトを使用する要素 (Windows Presentation Foundation など) には、テーマ フォントを使用します。DrawThemeText のようなテーマ API を使用して、テーマ シンボルに基づいてテキストを描画します。テーマ サービスが実行されない場合に備え、システムのメトリクスに基づく代替方法も用意しておいてください。
- Segoe UI には、9 ポイント以上のフォント サイズを使用します。Segoe UI フォントは 9 ポイント以上で適切に表示されるように設計されているので、9 ポイント未満のサイズは使用しないようにします。
- システムのテキストの色は、常に対応する背景色と一緒に使用します。たとえば、テキストの色に COLOR\_STATICTEXT を選択する場合は、その背景色にも COLOR\_STATIC を選択する必要があります。
- 新しいフォントを作成する場合は、常にシステム フォントのプロポーショナル サイズ フォントを基準にします。システム フォントのメトリクスから、太字にしたり、斜体にしたり、大きめまたは小さめのフォントを作成できます。
- 大きなブロックの読み取り専用テキスト (ライセンス条項など) の場合、背景は灰色ではなく明るい色にします。灰色の背景は、テキストが無効である印象を与え、読む気を損なわせます。

- 1行の文字数は半角換算で 65 文字以内になるようにします。これより長いとテキストが読みにくくなります(句読点や空白文字も含めてカウントします)。

## 属性

- 大部分の UI テキストは、属性のないプレーンテキストにします。属性を使用する場合は、次のように使用します。
  - 太字。コントロールのラベルにおいて、テキストの理解を助けるために使用します。また、必読のテキストに注目させるために、控えめに使用します。乱用すると太字の効果が減少します。
  - 斜体。テキストを (引用符なしで) 文字どおりに引用するために使用します。また、特定語句を強調するために、控えめに使用します。テキストボックスや編集可能なドロップダウンリストの中のプロンプトにも使用します。
  - 太字斜体。使用しません。
  - 下線。リンク以外には使用しません。強調するには代わりに斜体を使用します。

すべてのフォントで太字と斜体がサポートされるわけではないため、太字や斜体がなくてもテキストを理解できるようにします。

# 色

デザイン コンセプト

ガイドライン

全般

テーマ カラーとシステム カラーの使用

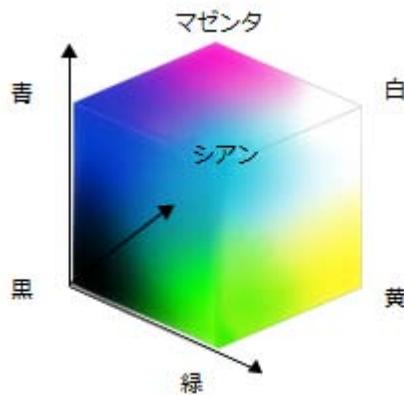
色の意味

データへの色の使用

ドキュメント

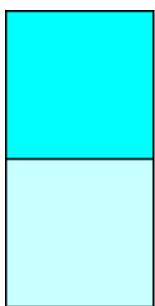
色は、大部分のユーザー インターフェイスにおいて重要な視覚的な要素です。単なる外観だけではなく、さまざまな意味に関連付けられ、心理的効果をもたらします。意味の解釈で混乱が生じないようにするには、一貫性のある方法で色を使用する必要があります。また、望ましい心理的効果を得るには、適切に色を使用する必要があります。

色はよく、色空間という観点で考えられます。最も一般的に使用される色空間は、RGB (赤、緑、青)、HSL (色相、彩度、輝度)、およびHSV (色相、彩度、明度) です。



RGB の色空間は、立方体として視覚化できます。

ディスプレイには RGB が採用されているため、開発者はよく色を RGB で考えますが、RGB の色空間は人間の知覚には対応しません。たとえば、濃いシアンに赤を追加した場合、知覚的には赤みが強くなるわけではなくシアンが薄くなります。



暗いシアン (red 0, green 255, blue 255)

赤を加えると...

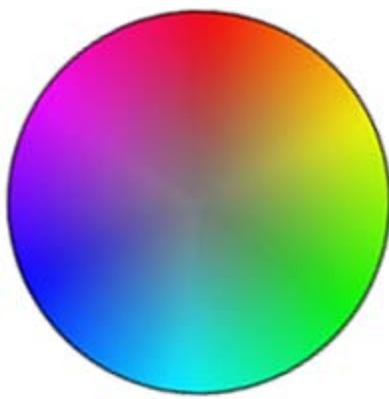
明るいシアン (red 200, green 255, blue 255)

この例では、濃いシアンに赤を追加した結果、赤みが強くなるのではなくシアンが薄くなっています。RGB の色空間は、人間の知覚には対応していません。

HSL と HSV の色空間は、色相、彩度、輝度または明度の 3 つの要素で構成されています。この 2 つの色空間はより人間の知覚に近いので、RGB の代わりによく使用されます。

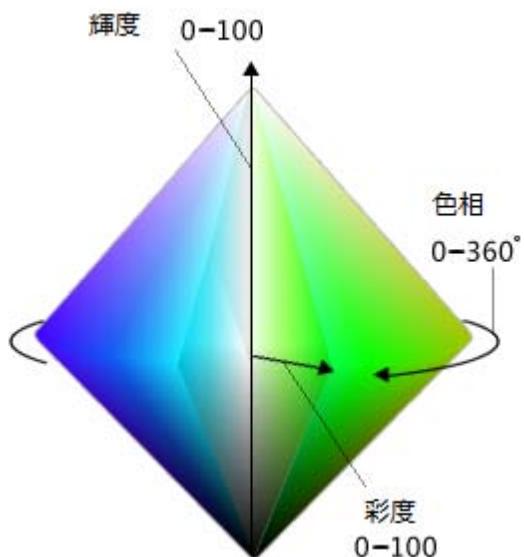
HSI の色空間は、双円すいで示され、最上部が白、最下部が黒、中央部が中間色になります。

- 色相: カラー ホイールの基本色です。0 ~ 360 度の範囲で、0 度と 360 度はいずれも赤です。



赤が 0 度、黄が 60 度、緑が 120 度、シアンが 180 度、青が 240 度、マゼンタが 300 度のカラー ホイール

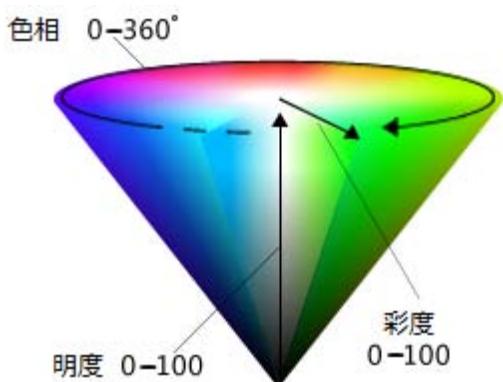
- 彩度: 色の鮮やかさを表します。0 ~ 100 の範囲で、100 が純色、0 が無彩色になります。
- 輝度: 色の明るさを表します。0 ~ 100 の範囲で、100 が最も明るい色(色相や彩度に関係なく白)、0 が最も暗い色(黒)になります。



HSL の色空間は、双円すいとして視覚化できます。

HSV の色空間も、単一の円すいであるという点を除いて HSL と同様です。

- 色相: カラー ホイールの基本色です。0 ~ 360 度の範囲で、0 度と 360 度はいずれも赤です。
- 彩度: 色の鮮やかさを表します。0 ~ 100 の範囲で、100 が純色、0 が無彩色になります。
- 明度: 色の明るさを表します。0 ~ 100 の範囲で、100 が最も明るい色(HSL 色空間の輝度の半分)、0 が最も暗い色(黒)になります。



HSV の色空間は、単一の円すいとして視覚化できます。

HSL と HSV のどちらの色空間でも、彩度が 0 の場合、輝度は灰色の度合いになります。Windows では、HSL と HSV の色空間は、通常 0 ~ 240 の等級に再マッピングされ、色が 32 ビットの値で表現されるようになっています。

注: [フォント](#) および [アクセシビリティ](#) に関するガイドラインは、それぞれ別の項目として記載しています。

## デザイン コンセプト

色を効果的に使用すると、プログラムのユーザー インターフェイス (UI) の効率化を図ることができます。色によって、ユーザーは特定の意味をひとめでつかむことができ、製品の外観もより魅力的で洗練されたものになります。

残念なことに、ビジュアル デザインの訓練を受けていない場合は特に、色を不適切に使用しがちです。色が不適切に使用されているデザインは、素人のような印象や古い印象を与えたり、他と紛らわしかったり、または単に見苦しいだけのものになります。このようなデザインは、色をまったく使用しないデザインよりも悪くなる可能性があります。

ここでは、色を効果的に使用するために必要な知識を説明します。

### 色の用途

通常、色は UI で以下のことを伝えるために使用されます。

- 意味。メッセージの意味を色に集約できます。たとえば、赤で問題またはエラー、黄で注意または警告、緑で正常が示されることがよくあります。
- 状態。オブジェクトの状態を色で示すことができます。たとえば、Windows® では選択状態やポイントしたときの状態を示すために色が使用されます。Web ページの場合、訪れていないリンクは青、訪れたリンクは紫で示されます。
- 区別。ユーザーは、同じ色のアイテムには何らかの関係があるものと考えます。そのため、色分けはオブジェクトを区別する効果的な方法です。たとえばコントロール パネルのアイテムでは、作業 ウィンドウに緑の背景が使用され、メイン コンテンツと視覚的に区別がつけられています。また Microsoft® Outlook® では、ユーザーがメッセージに異なる色のフラグを割り当てることができます。
- 強調。ユーザーの注意を引くために色を使用できます。たとえば Windows では、[メイン指示テキスト](#) が青で表示され、他のテキストより目立つようになっています。

もちろん、色がグラフィックスにおいて外観的理由だけで使用されることも多くあります。外観は重要ですが、UI 要素の色は外観よりも意味を重視して選択する必要があります。

### 色の解釈

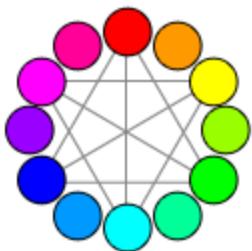
ユーザーがどのように色を解釈するかは、文化によって違うことがあります。たとえば米国では、花嫁の婚礼衣装は主に白で、黒は葬儀の色です。一方以前の日本では象徴するものが逆になり、伝統的に白は葬儀の中心の色で、黒は婚礼に幸運をもたらす色と考えられていました。

赤、黄、緑が示す状態の解釈は、世界中で一貫しています。これは、[UNESCO の道路標識および信号に関するウイーン協定](#)によるものです。この協定では、交通信号に関する世界共通の決まりとして、赤は止まれ、緑は進め、黄は注意を意味するということが定められています。これらの状態の色は、文化による解釈の違いを気にすることなく使用できます。

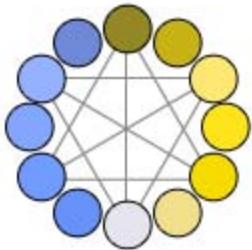
状態の色の他に、Microsoft® Windows® では一定の決まりに基づいて色に意味が割り当てられています。後の「ガイドライン」を参照して、プログラムでの色の使用方法が Windows での色の決まりと矛盾しないようにしてください。

### 色のアクセシビリティ

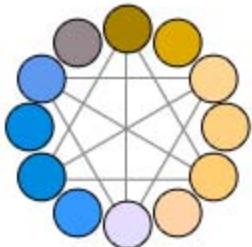
ソフトウェアで幅広いユーザーを得るために、色の使用方法に注意する必要があります。視力障礙のあるユーザーは、色がよく見えない可能性があります。白人の成人男性の約 8% には、何らかの色覚異常があります。その中で最も一般的なのは、赤と緑の区別が付きにくい色覚異常です ("色盲" という言葉もよく使われますが、これは不適切な表現とされています)。



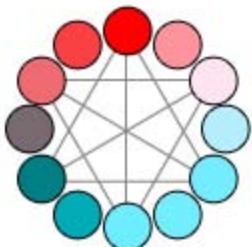
通常の色覚での原色の見え方



第一色覚異常の場合の原色の見え方 (男性人口の 1%)



第二色覚異常の場合の原色の見え方 (男性人口の 6%)



第三色覚異常の場合の原色の見え方 (男性人口の 1%)

詳細については、「[色覚に障碍を持っていたとしたら、あなたのサイトは見えるでしょうか？](#)」を参照してください。

#### 視覚的な補強手段としての色の使用

色の解釈とアクセシビリティの問題を解決する一番の方法は、以下のような一次的伝達手段の意味を視覚的に補強するための手段として色を使用することです。

- テキスト。簡潔なテキストは、UIで直接伝える場合もツールヒントを通して伝える場合も、通常は最も有効な一次的伝達手段です。



この例では、ツールヒントのテキストがアイコンの意味を伝えるために使用されています。

- デザイン。アイコンは、デザイン、特に輪郭の形状によって、簡単に見分けることができます。



この例では、標準のアイコンをデザインによって簡単に見分けることができます。

- 位置構成。相対的位置構成も伝達に使用できますが、この手段は他の伝達手段ほど効果がありません。効果を得るには、信号のように、標準的なよく知られた位置構成になっている必要があります。

色は多くのデザインの最も明白な属性ですが、常に代替手段と共に使用する必要があります。

### 色を使用した設計

逆説的ですが、色を使用した設計を行う最適な方法は、まずは色のないワイヤーフレームまたはモノクロのいずれかで設計し、その後で色を追加することです。こうすると、情報が色だけで伝えられる状況を避けることができます。また、モノクロ プリンターでも見栄え良く出力されるようになります。

### テーマ カラーまたはシステム カラーの使用

色を効果的に使用するには多くの複雑な要因を考慮する必要がありますが、Windows の UI では、いくつかの簡単なルールに従って適切なテーマ カラーまたはシステム カラーを選択するだけで済むこともあります。さらにユーザーの側でも、これらの配色を選択してカスタマイズできます。

テーマ カラーまたはシステム カラーを使用すると、あらゆるユーザーに対してそれぞれに適した色を表現できるだけでなく、あらゆる好み、スタイル、文化に対応できる配色を選択しなければならないという負担から解放されます。もちろんこれは、他の方法では実現できません。

#### 最も重要な点

適切なテーマ カラーまたはシステム カラーで色を選択します。色は、一次的伝達手段としては使用しません。意味を視覚的に補強する二次的手段として使用します。色が二次的なものになるように、設計はワイヤーフレームまたはモノクロで行います。

### テーマ カラーまたはシステム カラーの正しい使用

ユーザーが個人的なニーズに基づいてテーマ カラーまたはシステム カラーを選択するということと、テーマ カラーまたはシステム カラーが適切に作成されたものであるということを前提とすると、常に目的に合わせたテーマ カラーまたはシステム カラー、および前景と背景の正しい組み合わせを選択すれば、確実に見やすい色になり、ハイコントラスト モードを含むすべてのビデオ モードでユーザーの希望に沿えるようになります。たとえば、ウィンドウの文字用のシステム カラーとウィンドウの背景用のシステム カラーを組み合わせると、確実に見やすくなります。

具体的には、常に以下の点に留意します。

- 目的に基づいて色を選択します。現在の外観に基づいて色を選択しないようにします。ユーザーによって、または今後のバージョンの Windows で、外観が変更される可能性があります。
- 前景色と背景色の正しい組み合わせを選択します。正しい背景色と組み合わせることで初めて、前景色は確実に見やすい色になります。他の背景色と組み合わせたりはしません。前景色どうしを組み合わせることは論外です。
- 色の種類が混在しないようにします。常に、テーマ カラーは関連するテーマ カラー、システム カラーは関連するシステム カラー、固定色は他の固定色と組み合わせます。たとえば、テーマ カラーの文字を固定色の背景と組み合わせた場合、見やすいかどうかは保証されません。
- 固定色を使用する必要がある場合は、ハイコントラスト モードを特殊なケースとして扱います。

#### 最も重要な点

常に目的に合わせたテーマ カラーまたはシステム カラー、および前景と背景の正しい組み合わせを選択します。

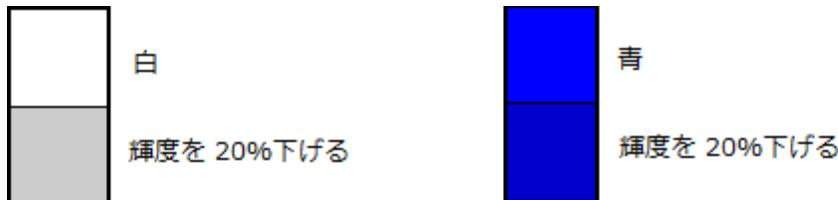
### その他の色の使用

Windows テーマでは包括的なセットとして各部分が定義されていますが、テーマ ファイルに定義されていない色がプログラムに必要な可能性もあります。このような色は、固定色にすることもできますが、それより良いのはテーマ カラーやシステム カラーから取得する方法です。この方法を戦略的に取り入れると、テーマ カラーやシステム カラーを使用するメリットをすべて得られると同時に、テーマ カラーやシステム カラーだけでは得られない柔軟性も得ることができます。

たとえば、ウィンドウの背景を、テーマのウィンドウの背景色より暗くする必要があるとします。HSL の色空間では、暗い色とは輝度の低い色を意味します。したがって、次の手順でより暗いウィンドウの背景色を取得できます。

1. ウィンドウの背景のテーマカラーをRGBで取得します。
2. RGBをHSLの値に換算します。
3. 輝度の値を下げる(約20%など)。
4. RGBの値に換算し直します。

この方法により、元の色がとても暗い色でない限りは、取得した色は元の色より暗い色に感じられます。



この例では、テーマカラーからより暗いウィンドウの背景色を取得しています。

## 色のテスト

プログラムの色の使用方法が、アクセシビリティを妨げず、一次的伝達手段になっていないことを確認するには、[富士通 ColorDoctor](#) や [Vischeck](#)などのユーティリティを使用して、次の点をチェックすることをお勧めします。

- ・ 全体的な色への依存度(グレースケールフィルターを使用)
- ・ 特定の色覚異常の問題(第一色覚異常、第二色覚異常、および第三色覚異常のフィルターを使用)

プログラムの色の使用方法が正しく組み込まれていることを確認するには、次のモードでプログラムをテストします。

- ・ 既定の Windows テーマを使用してテーマを有効にしたモード
- ・ 既定以外のテーマを使用してテーマを有効にしたモード
- ・ テーマを無効にしたモード([コントロールパネル]の[個人設定]で[テーマ]を選択し、[テーマの設定]の[Windows クラシック]を選択)
- ・ [ハイコントラスト 黒]のモード(黒い背景に白いテキスト)
- ・ [ハイコントラスト 白]のモード(白い背景に黒いテキスト)

すべての画面要素は、モードを変更した直後でも、見やすく想定どおりに表示される必要があります。

## ガイドライン

### 全般

- ・ 色は、一次的伝達手段としては使用しません。意味を視覚的に補強する二次的手段として使用します。

### テーマカラーとシステムカラーの使用

- ・ 可能であれば常に、適切なテーマカラーまたはシステムカラーで色を選択します。こうすることで、ユーザーそれぞれに適した色を表現できます。
- ・ 目的に基づいてテーマカラーとシステムカラーを選択します。現在の外観に基づいて色を選択しないようにします。ユーザーによって、または今後のバージョンの Windows で、外観が変更される可能性があります。
- ・ 前景色と背景色の正しい組み合わせを選択します。正しい背景色と組み合わせることで初めて、前景色は確実に見やすい色になります。他の背景色と組み合わせたりはしません。前景色どうしを組み合わせることは論外です。
- ・ 色の種類が混在しないようにします。常に、テーマカラーは関連するテーマカラー、システムカラーは関連するシステムカラー、固定色は他の固定色と組み合わせます。たとえば、テーマカラーの文字を固定色の背景と組み合わせた場合、見やすいかどうかは保証されません。

- ・ テーマ カラーまたはシステム カラー以外の色を使用する必要がある場合は、次の点に留意します。
  - ・ 値を固定した色を使用するよりも、テーマ カラーまたはシステム カラーから色を取得するようにします。前の「その他の色の使用」で説明した手順に従います。
  - ・ ハイコントラスト モードを特殊なケースとして扱います。
- ・ テーマの変更に対応します。テーマの変更は、標準のウィンドウ枠やコモン コントロールを含むウィンドウには自動的に反映されます。カスタム ウィンドウ枠や、カスタム コントロール、オーナー描画コントロールなどの色の使用を含むウィンドウの場合、テーマの変更は明示的に処理する必要があります。
  - ・ 開発者向け情報: テーマの変更イベントには、WM\_THEMECHANGED メッセージを処理することで対応できます。

## 色の意味

- ・ 可能であれば常にテーマ カラーとシステム カラー(またはこれらの色から取得した色)を使用し、他の色を使用する場合は、色の使用方法が次に示す Windows での色の使用方法と矛盾しないようにします。

色相	意味	Windows での使用方法
青と緑	Windows ブランド	背景: Windows のブランド化
グラス、黒、灰色、白	中立	背景: 標準のウィンドウ枠、スタート メニュー、タスクバー、サイドバー 前景: 標準のテキスト
青	開始、コミット	背景: 既定のコマンド ボタン、検索、ログオン アイコン: 情報、ヘルプ 前景: メイン指示テキスト、リンク
赤	エラー、停止、脆弱、重大、緊急、制限	背景: 状態、進行の停止(進行状況バー) アイコン: エラー、停止、ウィンドウを閉じる、削除、必須の入力、不足、使用不可
黄	警告、注意、疑わしい	背景: 状態、進行の一時停止(進行状況バー) アイコン: 警告
緑	進む、続行、進行、安全	背景: 状態、正常な進行(進行状況バー) アイコン: 進む、完了、更新 前景: パスおよび URL(検索結果内)
紫	訪問済み	前景: 訪れたリンク(Windows Internet Explorer® およびドキュメント内のリンク)

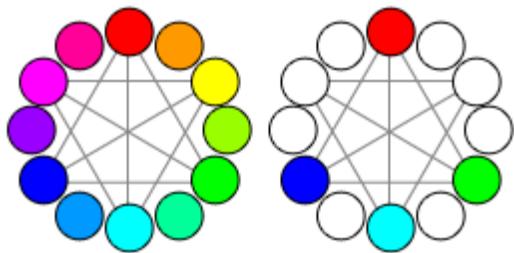
- ・ ここに挙げる意味を避けるには、中の上から低い彩度、かつ高いか低い輝度の色を選択します。ユーザーは、純色または高い彩度かつ中くらいの輝度の色で、これらの意味を想起します。そのため、彩度と輝度を変えることで、これらの意味を回避できます。



この例では、彩度と輝度が異なる 3 種類の黄色のうち、純色に近い彩度かつ中くらいの輝度のものが、警告を想起させます。黄色のフォルダーアイコンは、警告のように見えません。

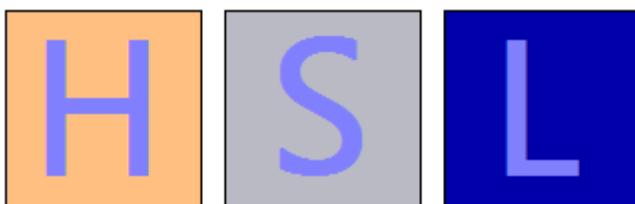
## データへの色の使用

- ・ 有用であれば、データに色を割り当てて、ユーザーが区別できるようにします。ユーザーは色が似ているデータを類似の意味のデータととらえることに注意してください。
- ・ 区別しやすい色を既定で割り当てます。一般的に、HSL や HSV の色空間で遠くの位置にある色どうしは区別しやすく、背景とも大きなコントラストを維持できます。
  - ・ 色を選択する際には、隣接する色相ではなく、三角形の頂点にくる調和色や補色を優先します。



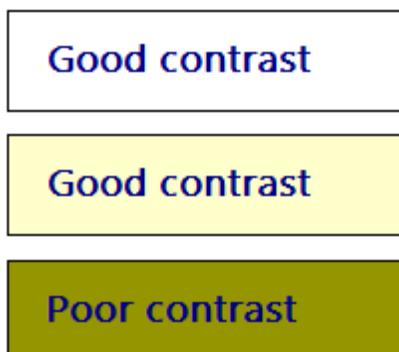
この例では、最初の色が赤の場合、次の色はマゼンタ、紫、オレンジ、黄ではなく、青、緑、またはシアンがふさわしいことを示しています。

- 色相、彩度、または輝度に大きな差がある色どうしは、大きなコントラストを成します。



この例では、ライトブルーの基本色が、色相、彩度、または輝度の差が大きい背景とコントラストを成すことを示しています。

- 白やとても明るい背景でコントラストをつけると、前景色を区別しやすくなります。



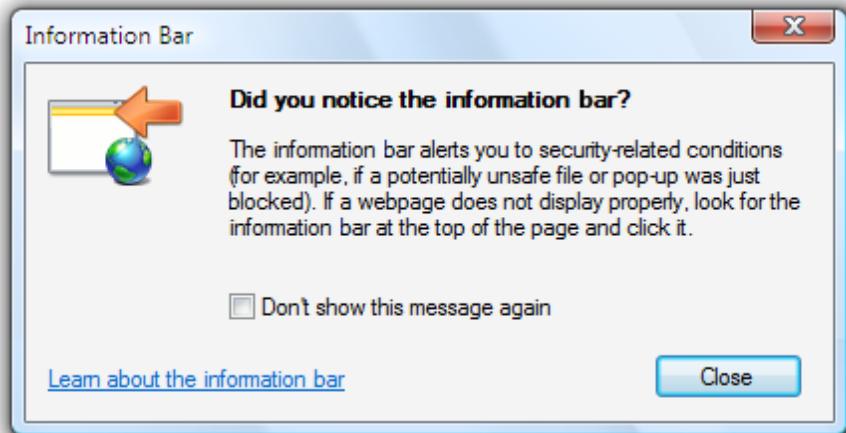
この例では、白や明るい背景色により、前景色が区別しやすくなっています。

- 色の割り当ては、ユーザーがカスタマイズできるようにします。個人が好みに応じて主観的に色を選択できる必要があります。セットになった色が多くある場合は、配色を提供することで、ユーザーがまとめて色を変更できるようにします。
- ユーザーが色の割り当てにラベルを付けられるようにします。こうすると、色の特定や検索がしやすくなります。
- UI の色とは異なり、データの色はシステム カラーを変更しても変更されないようにします。

## ドキュメント

- UI 要素は色ではなく名前で示します。色で要素を示すことはアクセシビリティを妨げます。また、システム カラーは変更される可能性があります。UI 要素の名前がよく知られていないか、わかりにくい場合は、スクリーンショットで明示します。

正しい例:



間違った例:



間違った例のメッセージでは、Windows Internet Explorer の情報バーが名前ではなく色で示されています。

## アイコン

デザイン コンセプト  
ガイドライン  
遠近法  
光源  
影  
色と彩度  
サイズ要件  
細部描写のレベル  
アイコンの開発  
リストビュー、ツールバー、ツリービューなどのコンテキスト内のアイコン

"アイコン" は、オブジェクトを画像で表現したものです。プログラムの視覚的なアイデンティティとしての外観面だけでなく、ユーザーがほぼ一瞬で把握できる形で意味を伝える手段という実用面でも重要です。Windows Vista® には新しいスタイルのアイコン表現が導入され、Windows にさらなるディテールと洗練性を加えています。

注: [標準アイコン](#)に関するガイドラインは、別の項目として記載しています。

## デザイン コンセプト

Aero とは、Windows Vista のユーザー エクスペリエンスの呼称です。美しいデザインとしての価値と、ユーザー インターフェイス (UI) を背後から支えているビジョンの両方を表現しています。Aero は、"Authentic、Energetic、Reflective、Open" の略です。Aero での目標は、プロフェッショナルかつ美しいデザインを確立することです。Aero の美しさは、高品質でエレガントなエクスペリエンスを生み出します。ユーザーの生産性を向上し、さらには感情的な反応さえも生じさせます。

Windows Vista のアイコンは、Windows® XP スタイルのアイコンと次の点が異なります。

- イラスト風スタイルから抜け出してより写実的スタイルになっていますが、写真ほどリアルではありません。アイコンは象徴化されたイメージであり、写真のようにリアルなものより見栄えは良いといえます。
- アイコンの最大サイズは 256 × 256 ピクセルで、高dpi 表示に適しています。これらの高解像度アイコンを使用することで、リストビューで大きなアイコンを高画質で表示できます。
- 通常は決められたドキュメントアイコンが表示されますが、これらは実用的な場合は常にコンテンツのサムネイルで置き換えられるので、ドキュメントを識別したり見つけ出しが容易になります。
- ツールバーのアイコンは、小さいサイズおよび視覚的な区別のために最適化されるので、描画は詳細ではなく遠近法も適用されていません。

優れたデザインのアイコンとは、次のようなものです。

- プログラムの視覚的な情報伝達性を高める。
- ユーザーの、プログラムのデザインへの全体的な印象と操作性の評価に大きな影響を与える。
- プログラム、オブジェクト、アクションなどの識別、習得、探索が容易になり、ユーザビリティを向上する。

次のイメージは、Windows Vista のアイコン表現である Aero style と、Windows XP で使用されるアイコン表現スタイルとの違いを示しています。



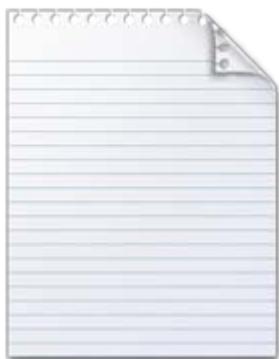
Windows Vista のアイコン (左側の錠と鍵) は美しく明確、かつ細部が描写されています。これらのアイコンは、描画されたというよりもレンダリングに近いものですが、写真のような完全なリアルさはありません。



Windows Vista のアイコン (左側の 2 つ) は、プロフェッショナルで美しく、アイコンの品質を向上させるため細部まで注意が行き届いています。



Windows Vista のアイコンでは、光のバランスが表現され、遠近表現とディテールを正確に認知できるようになっています。そのため、アイコンの大小にかかわらず、間近から見ても遠くから見ても見栄えの良いものになっています。さらに、このアイコン表現スタイルは高解像度の画面に有効です。

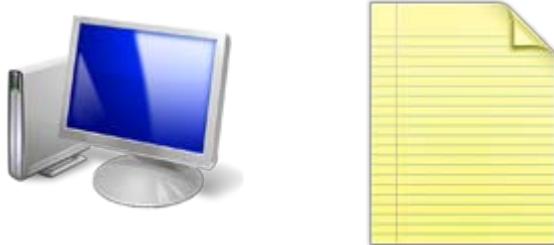


これらの例では、各種アイコン (遠近法で表現された 3D オブジェクト、正面からの (フラットな) アイコン、およびツールバー アイコン) を示しています。

## ガイドライン

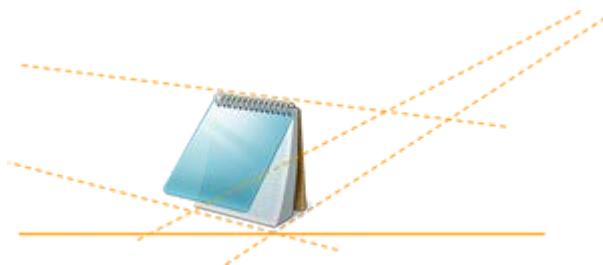
## 遠近法

- Windows Vista のアイコンは、3D で立体的なオブジェクトとして遠近法で表現されるか、または真正面から見た 2D オブジェクトとして表現されます。ドキュメントや紙など、実際に平面的なファイルやオブジェクトには、フラット アイコンを使用します。



典型的な 3D アイコンとフラット アイコン

- 3D オブジェクトは、低い俯瞰から見た、2つの消失点を持つ立体オブジェクトとして遠近法で表現します。



この例では、3D アイコンで一般的な遠近法と消失点を示しています。

- 同じアイコンでも、小さいサイズの場合は、遠近法表示から真正面の表示に変更できます。大きさが 16 × 16 ピクセル以下の場合は、アイコンを真正面からレンダリングします。大きいアイコンの場合は、遠近法表示を使用します。
  - 例外: ツールバーのアイコンは、大きいサイズであっても常に正面向きで表示します。



この例では、同じアイコンのサイズ別の扱いを示しています。

## 光源

- 遠近法グリッド内のオブジェクトに対する光源は、オブジェクトからわずかに手前、かつわずかに左側の上方にします。
- この光源による影を、オブジェクトの底部から後方および右側にわずかに伸ばします。
- すべての光線は平行にし、太陽光線のようにオブジェクトに同じ角度で当てます。すべてのアイコンおよびスポットライト効果で、同じライティング条件にすることを目標にします。平行な光線を当てることで、均一な長さと濃さで影が入り、複数のアイコン間の一貫性が向上します。

## 影

## 全般

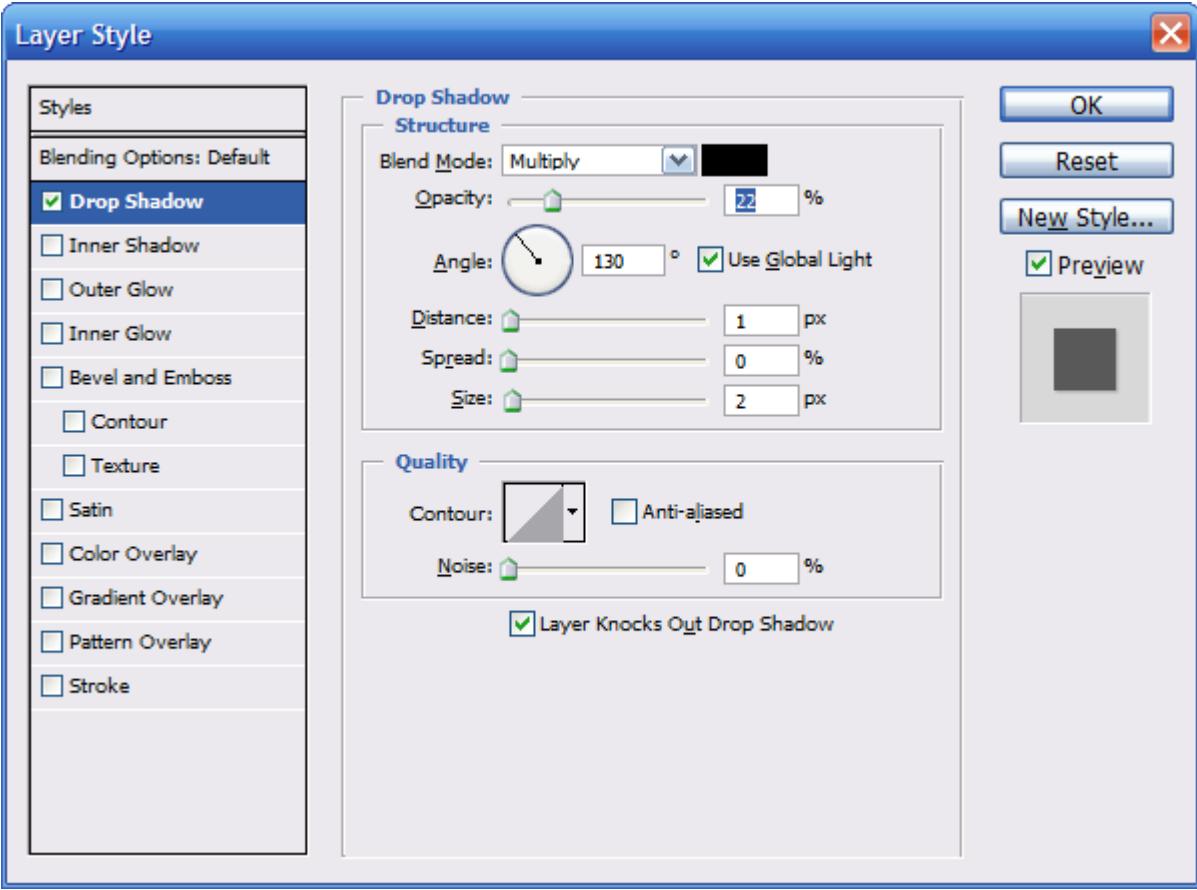
- 影を使用して、背景から視覚的にオブジェクトを浮き出させ、3D オブジェクトを宙に浮いているのではなく接地しているように表示します。
- 影の不透明度は、30 ~ 50% の範囲にします。アイコンの形状や色に応じて、異なる不透明度の影を使用することが必要な場合もあります。
- アイコン ボックスのサイズで影が切り詰められないように、必要に応じて影をぼかしたり、短くします。
- 24 × 24 以下の小さなサイズのアイコンには、影を使用しないでください。



典型的なアイコンの影

## フラット アイコン

- フラット アイコンは、通常、ファイル アイコンおよび平らな現実世界のオブジェクトを示すアイコンに使用します。これらにはドキュメントや紙などが含まれます。
- フラット アイコンに対しては光源を左上 130 度にします。
- 小さなアイコン (たとえば 16 × 16 や 32 × 32) は、見やすいように単純化されます。ただし、アイコン内に反射がある場合に、小さな影を使用することもあります (反射も多くの場合は単純化されます)。影には、不透明度 30 ~ 50% を使用します。
- フラット アイコンにレイヤー効果を使用できますが、他のフラット アイコンとの比較で使用する必要があります。最適な外観、および同じサイズのアイコンや Windows Vista の他のアイコンとの一貫性を考慮して、オブジェクトの影は多少変化します。影の修正が必要となる場合もあります。特に、オブジェクトが他のオブジェクトの上に重なっている場合です。
- 目的の効果を得るために、ごく少量の色を使用します。影を使用すると、オブジェクトの座りが良くなります。色によって影は重くも軽くも見え、重すぎる場合はイメージをゆがめることもあります。



[レイヤースタイル] ダイアログ ボックスでの [ドロップシャドウ] (影) オプション、およびフラット アイコンでの典型的な影

基本的なフラット アイコンでの影の範囲

特性	範囲
色	黒
描画モード	乗算
不透明度	アイテムの色に応じて 22 ~ 50%
角度	120 ~ 130 (グローバル光源使用の場合)
距離	3 (256 × 256 の場合) ~ 1 (32 × 32 の場合)
スプレッド	0

### 3D アイコン

- 状況に応じて 3D アイコンの影を作成します。投影距離の範囲内に収まるように、完全な透明になるまでぼかします。影を付ける必要があるサイズの場合は、アイコン全体のサイズより少し小さめな画像を作成し、影を付ける領域を確保します。アイコンの端で影が途切れないようにします。



これらの例では、オブジェクト自体の形状や配置に基づいて作成された各種アイコンを示しています。アイコンボックスのサイズで切り詰められないように、影をぼかしたり、短くする必要がある場合があります。

### 色と彩度

- 通常、Windows Vista の色の彩度は Windows XP より低くなっています。
- グラデーションを使用して、よりリアルな外観のイメージを作成します。
- 標準アイコン用のカラー・パレットはありませんが、多くのコンテキストやテーマと調和して表示されるようにする必要があります。色の標準セットを優先し、警告アイコンなどの標準アイコンの配色は変更しないでください。ユーザーが意味を解釈する際に混乱が生じます。その他のガイドラインについては、「[色](#)」を参照してください。

- アイコンファイルには、8ビットおよび4ビットのパレットを使用したバージョンが必要です。これは、リモートデスクトップでの既定の設定をサポートするためです。これらのファイルはバッチ処理で作成できますが、より見やすくするために修整が必要な場合があるため、確認が必要です。



カラー パレットに厳密な制限はありませんが、最も高い彩度となる純色(右上)の使用は避けます。

- ビットレベル: ICO デザイン、32 ビット(アルファを含む)+8 ビット+4 ビット(自動ディザリング、ピクセル拡大用途のみで最重要)。256×256 ピクセルイメージの 32 ビットコピーのみ含めます。また、256×256 ピクセルイメージのみファイルサイズを小さく保つために圧縮します。いくつかのアイコンツールでは、Windows Vista 向けの圧縮が提供されています。
- ビットレベル: ツールバー用、24 ビット+アルファ(1 ビットマスク)、8 ビットおよび4 ビット。
- ツールバーまたは AVI ファイル: 透明色の背景として、マゼンタ(R255 G0 B255)を使用します。

## サイズ要件

### 全般

- 目につく場所のアイコンには特に注意を払う必要があります。これらには、メインアプリケーションのアイコン、Windows エクスプローラーで表示されるファイルアイコン、スタートメニュー/デスクトップに表示されるアイコンなどが含まれます。
  - アプリケーションアイコンおよびコントロールパネルのアイテム: 16×16、32×32、48×48、256×256 でフルセットとなります。(コードで 32 ~ 256 に拡大縮小します)。.ico ファイル形式である必要があります。クラシック モードの場合、フルセットは 16×16、24×24、32×32、48×48 および 64×64 になります。
  - リストアイテムのアイコンのオプション: ファイルの種類(.doc など)のライブ サムネイルまたはファイルアイコンをフルセットで使用します。
  - ツールバーのアイコン: 16×16、24×24、32×32。ツールバーのアイコンは、サイズが 32×32 であっても、3D ではなく常にフラットにすることに注意してください。
  - ダイアログボックス アイコンおよびウィザード アイコン: 32×32 および 48×48。
  - オーバーレイ: 主要なシェル コード(たとえばショートカットを表すもの)は 10×10(16×16 用)、16×16(32×32 用)、24×24(48×48 用)、128×128(256×256 用)。形状や視覚的なバランスにより、少し小さめにする場合も、このサイズに近くなるようにします。
  - クリック起動領域: Alt + Tab キーを押したときに動的なオーバーレイでアイコンが表示されますが、これらは 48×48 から縮小されたものです。より鮮やかに表示するには .ico ファイルに 40×40 を追加します。
  - バルーン アイコン: 32×32 および 40×40。
  - 追加のサイズ: 128×128、96×96、64×64、40×40、24×24、22×22、14×14、10×10、および 8×8。注釈、ツールバーストリップ、オーバーレイ、高 dpi、特別な用途向けなどのファイルを作成するリソースとして、これらのサイズを用意しておくと便利です。その領域内のコードによ

り、.ico、.png、.bmp、またはその他のファイル形式を使用できます。

## 高dpi用

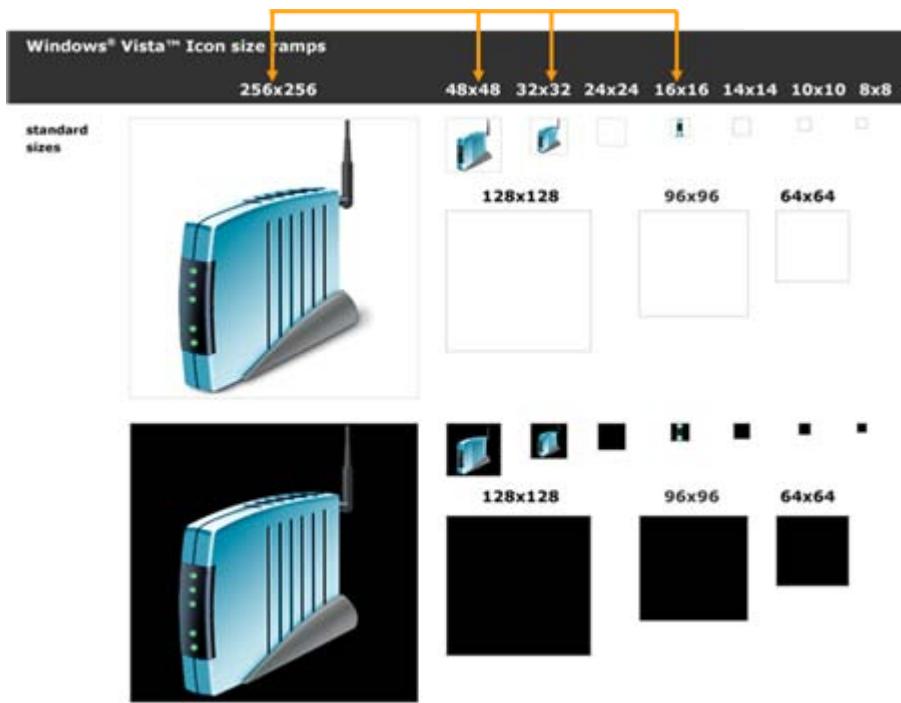
- Windows Vistaでは、96 dpiと120 dpiが対象とされています。

次の表に、2つの一般的なサイズのアイコンに適用された拡大縮小比率の例を示します。これらのサイズのすべてが.icoファイルに含まれる必要はないことに注意してください。大きなアイコンは、コードによってスケールダウンされます。

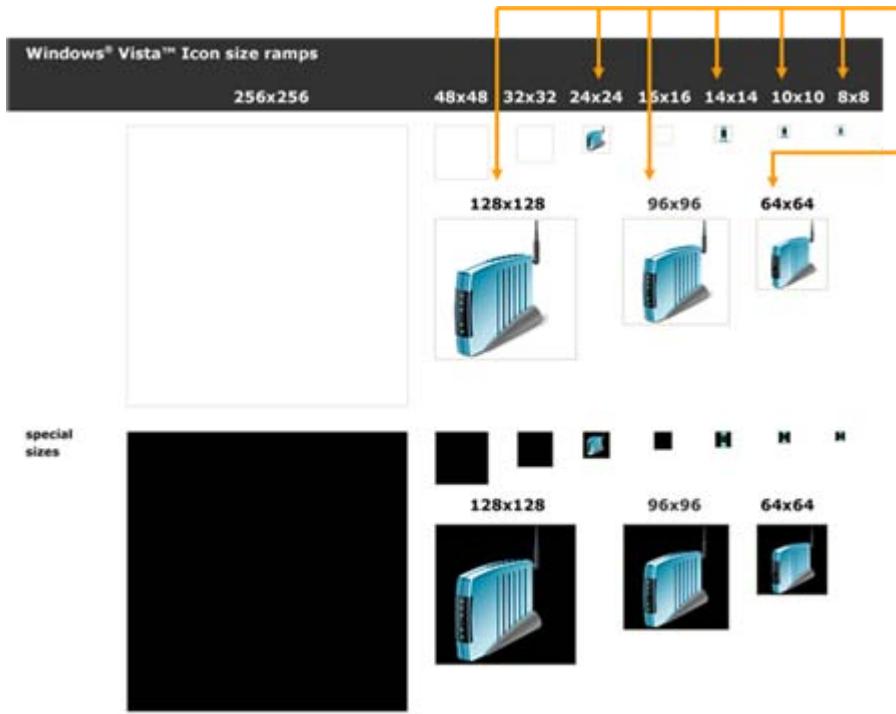
dpi	アイコンのサイズ	拡大縮小の係数
96	16 × 16	1.0 (100%)
120	20 × 20	1.25 (125%)
144	24 × 24	1.5 (150%)
192	32 × 32	2.0 (200%)

dpi	アイコンのサイズ	拡大縮小の係数
96	32 × 32	1.0 (100%)
120	40 × 40	1.25 (125%)
144	48 × 48	1.5 (150%)
192	64 × 64	2.0 (200%)

## .icoファイルのサイズ(標準)



## .icoファイルのサイズ(特別な場合)

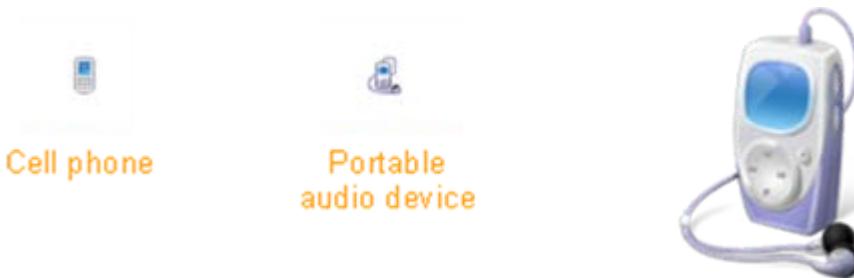


## 注釈とオーバーレイ

- 注釈はアイコンの右下隅に表示し、アイコン領域の 25% を占める必要があります。
  - 例外: 16 × 16 のアイコンには 10 × 10 の注釈が必要です。
- アイコン上には複数の注釈を使用しないでください。
- オーバーレイはアイコンの左下隅に表示し、アイコン領域の 25% を占める必要があります。
  - 例外: 16 × 16 のアイコンには 10 × 10 のオーバーレイが必要です。

## 細部描写のレベル

- これらのアイコンの多くでは、16 × 16 サイズが依然として広く使用されています。したがってこのサイズは重要なサイズとなっています。
- このサイズのアイコンの細部描写では、アイコンの要点が明確に示される必要があります。
- アイコンが小さくなるにつれ、大きいサイズに見られる透明度や一部の特別な細部描写は、単純化のためと要点を伝えるために省く必要があります。
- 属性や色を誇張し、重要な形を強調する必要があります。



16 × 16 では、ポータブルオーディオデバイス用のアイコンが携帯電話用のアイコンと間違われやすくなります。そのため、イヤホンは重要な細部描写であり表示させる必要があります。

- 256 × 256 サイズから単にスケールダウンするだけではうまくいきません。
- すべてのサイズには、サイズに応じたレベルの細部描写が必要となります。アイコンが小さくなればなるほど、定義している細部描写を誇張する必要があります。

### Good Level of Detail



### Poor Level of Detail



## アイコンの開発

### アイコンのデザインと作成

- 経験豊富なグラフィックデザイナーに依頼します。優れたグラフィックス、イメージ、アイコンを作成するには、専門家と協力してください。ベクター アートや 3D プログラムを使用したイラストの作成経験がある専門家との協力をお勧めします。
- 繰り返しの処理で一連のセットを作成するための計画を立てます。最初のコンセプト スケッチから、コンテキスト内の模擬表示、最終段階でのレビュー、および運用製品での表示と仕上がりまでを想定します。
- アイコンの作成はコストがかかる可能性があるため、前もって検討します。既存の詳細情報および要件を収集します。これには、必要とされるアイコン セット、主要機能とそれぞれの意味、セット内で明確に表現するファミリ(群)、ブランド要件、正確なファイル名、コード内で使用するイメージ形式、サイズ要件などが含まれます。デザイナーとの作業時間を最大限に活用できるように、事前に準備します。
- デザイナーが製品をよく知らない場合もあるので、必要に応じて、機能情報、スクリーンショット、仕様書などを提供します。
- 必要に応じて、地政的検討および法的検討を計画します。
- スケジュールを綿密に計画し、定期的に連絡を取り合います。

### コンセプト スケッチから作成完了まで



- コンセプト スケッチを作成します。
- 異なるサイズでコンセプトを試します。
- 必要に応じて 3D でレンダリングします。
- 異なる背景色で各サイズをテストします。
- 実際の UI のコンテキスト内でアイコンを評価します。
- 最終の .ico ファイルまたはその他のグラフィック リソース形式を作成します。

## ツール

- 鉛筆と紙: 最初のコンセプト アイデアを一覧にし、スケッチを作成します。

- **3D Studio Max:** 3D オブジェクトをレンダリングし、遠近感を持たせます。
- **Adobe Photoshop:** スケッチと繰り返しの処理を行い、コンテキスト内で模擬表示をして、細部を完成させます。
- **Adobe Illustrator/Macromedia Freehand:** スケッチと繰り返しの処理を行い、詳細描画を完成させます。
- **Gamani Gif Movie Gear:** .ico ファイルを作成します (必要に応じて圧縮します)。
- **Axialis IconWorkshop:** .ico ファイルを作成します (必要に応じて圧縮します)。
- Microsoft Visual Studio® では、Windows Vista のアイコンはサポートされていません (アルファ チャネルや 256 色以上はサポートされていません)。

## 作成

### ステップ 1: コンセプト化

- 可能であれば確立したコンセプトを使用し、アイコンの意味および他の用途との関連性が一貫性を持つようにします。
- UI のコンテキスト内でアイコンがどのように表示されるか、および一連のアイコンの中でそれがどのように機能するかを検討します。
- 既存のアイコンを修正する場合は、複雑さを軽減できるかどうかを検討します。
- グラフィックの文化的な影響を検討します。アイコン内の文字、語句、手、顔などの使用は避けます。人やユーザーを表現することが必要な場合は、できる限り一般化して描写します。
- 複数のオブジェクトをアイコンの単一イメージにまとめる場合は、イメージが小さなサイズにどのように縮小されるかを検討します。1つのアイコン内に使用するオブジェクトは 3 つ以内とします (2 つ以内を推奨)。16 × 16 サイズについては、認識しやすくするためにオブジェクトの削除やイメージの単純化を検討します。
- アイコン内では Windows フラグを使用しないでください。

### ステップ 2: イラスト化

- Windows Aero style のアイコンのイラスト化には、Macromedia Freehand、Adobe Illustrator などのベクターツールを使用します。このトピックで既に説明したように、パレットやスタイルの特性を使用します。
- Freehand や Illustrator を使用してイメージをイラスト化します。ベクターイメージをコピーして Adobe Photoshop に貼り付けます。
- Photoshop でテンプレート レイヤーを作成して使用し、規定されたサイズの正方形の領域内で作業するようにします。
- 影を付ける必要があるサイズの場合は、アイコン全体のサイズより少し小さめの画像を作成し、影を付ける領域を確保します。
- ディレクトリ内のすべてのアイコンの位置が同じになるように、正方形の底辺に合わせてイメージを配置します。影が途切れないようにします。
- 1 つのイメージや一連のイメージに別のオブジェクトを追加する場合は、メインオブジェクトを固定位置に保ち、小さめのサイズのフラットイメージを状況に応じて左下や右上などの固定位置に配置します。

### ステップ 3: 24 ビット イメージの作成

- Photoshop に各サイズを貼り付けたら、特に 16 × 16 以下のサイズのイメージの見やすさを確認します。色の比率を使用したピクセル拡大が必要な場合があります。透明度を下げることが必要な場合もあります。小さめのサイズの場合、キー ポイントに焦点を当てるために、ある特徴を誇張したり、ある特徴を削除することが一般的に行われています。
- 8 ビットアイコンは、32 ビット未満のカラー モードで表示され、8 ビットアルファ チャネルを持ちません。アンチエイリアスは行われないので、アイコンの端がギザギザになったり、イメージが見にくことがあります。そのため、アイコンの端などの部分をきれいに整える必要があります。
- Photoshop で、24 ビットイメージ レイヤーを複製し、4 ビットイメージ用にそのレイヤーの名前を変更します。4 ビットイメージを Windows の 16 色パレットでインデックス化します。
- このパレットの 16 色だけを使用してイメージを整えます。オブジェクトの色と明るさだけが違う色が輪郭を構成している場合は、輪郭線に灰色や黒を使用することをお勧めします。

- ビットマップの場合は、背景色が透過色になるので、イメージ自体に背景色と同じ色が使用されていないことを確認します。多くの場合、マゼンタ (R255 G0 B255) が背景の透過色に使用されます。

#### ステップ 4: 8 ビットイメージおよび 4 ビットイメージの作成

- 24 ビットイメージを 32 ビットアイコンにする準備が整ったので、次は 8 ビット版を作成する必要があります。
- ここで各アイコンを並べてコンテキスト内でのスクリーンショットを確認することをお勧めします。コンテキスト内の他のアイコンやアイコンファミリを確認することで思わぬ発見をすることがあります。この工程により、時間とコストを節約できます。ファイルが作成されて引き渡される前に問題を把握できます。
- 影を必要とするサイズのイメージに、影を追加します。
- 影と 24 ビットイメージを結合します。
- 各サイズ用の新しい Photoshop ファイルを作成します。適切なイメージをコピーして貼り付け、各ファイルを .psd ファイルとして保存します。
- イメージレイヤーを背景レイヤーと結合しないでください。ファイル名にサイズや色深度を含めると作業中は便利ですが、最終的にはファイル名の変更が必要になる場合があります。

#### ステップ 5: .ico ファイルの作成

- 作成者のニーズやスキルに最も適したアプリケーションを選択します。出荷製品に使用するアイコンは、正規購入したツールまたはライセンスされたツールで作成する必要があることに注意してください。試用版は使用できません。
- 以下の製品はいずれも、Windows Vista のアイコンを作成したデザイナーが使用しています。また、両製品共に、Adobe Photoshop CS へのエクスポート機能を備えています。
  - Gamani Gif Movie Gear: .ico ファイルの作成
  - Axialis IconWorkshop: .ico ファイルの作成
- Visual Studio では、Windows Vista のアイコンはサポートされていません (アルファチャネルや 256 色以上はサポートされていません)。そのため、使用は推奨しません。
- アイコン (.ico 形式) ファイルには、24 ビット + アルファだけでなく、4 ビット版や 8 ビット版を含める必要があります。
- 使用するアイコン作成プログラムにかかわらず、"Windows アイコン (.ico)" としてファイルを保存します。
- アイコン資産の中には、実際にビットマップストリップ (ツールバー用などにアルファチャネルが必要) で構成されるものや、透明度付きで保存された .png ファイルを使用するものがあります。すべての場合に .ico 形式が必要とは限りません。コードでサポートされている形式を確認してください。

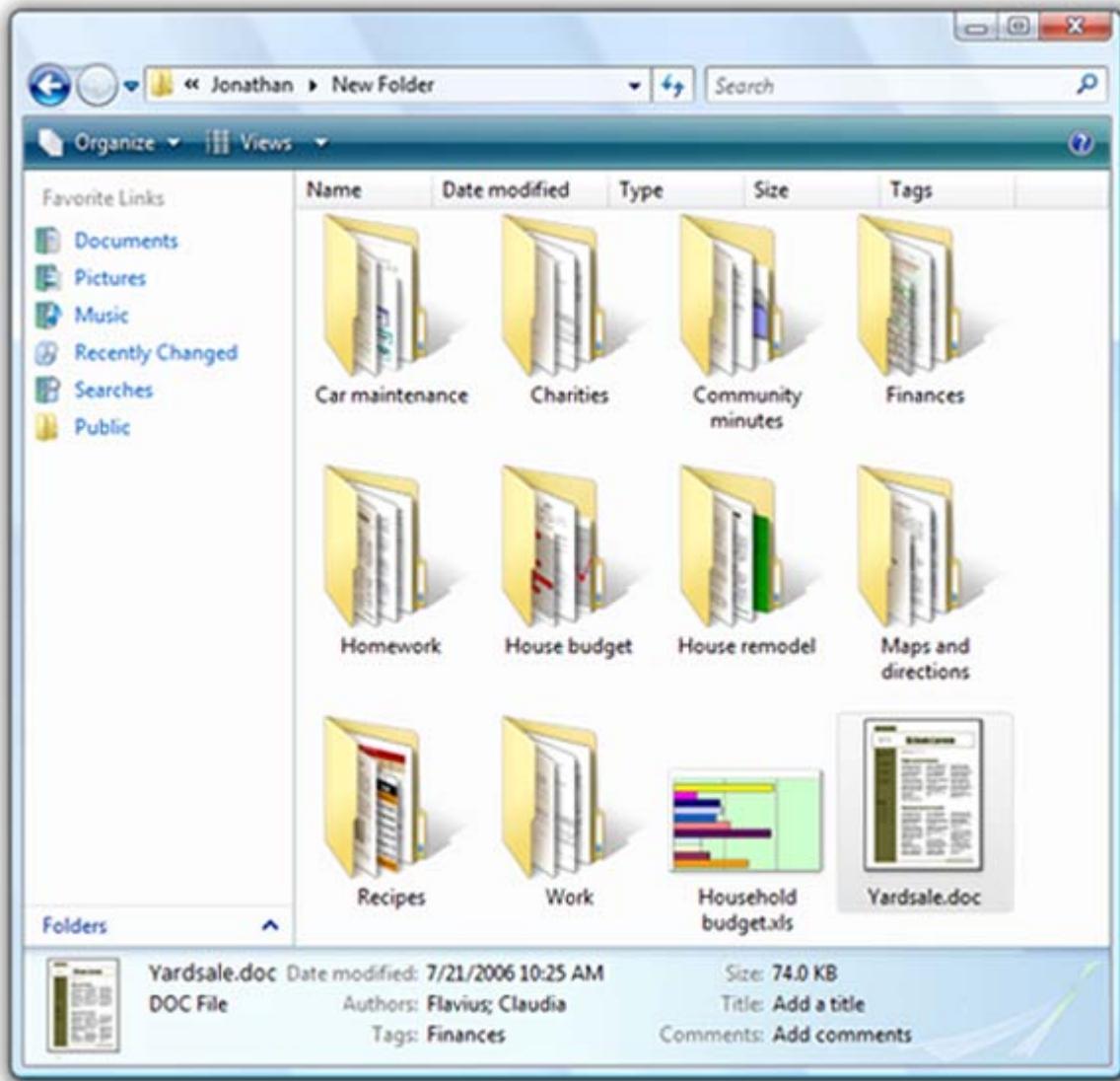
#### ステップ 6: 評価

- すべてのサイズを確認します。
- ファミリをまとめて確認し、ファミリの類似性、視覚的なバランス、識別性などを評価します。
- コンテキスト内で確認し、相対的な重みと視認性を評価します (あるアイコンだけがひどく目立つたりしていないことを確認します)。
- 現在は使用されていても、近い将来に使用される可能性があるかどうかを検討します。このアイコンに注釈が付けられたり、オーバーレイ表示されたりする可能性を検討します。
- コードを確認します。

#### リスト ビュー、ツール バー、ツリー ビューなどのコンテキスト内のアイコン

##### リスト ビュー

- Windows Vista では、小さいサイズのコンテンツ表示でもユーザーが目的のファイルを直接認識できるようなファイルに対しては、サムネイルを使用します (このためには、Windows サムネイル アプリケーション プログラミング インターフェイスを使用します)。



- サムネイル上のアプリケーションアイコンオーバーレイ(ここでは表示されていません)があると、ファイルのプレビュー表示に加え、ファイルの種類に関連付けられているアプリケーションがわかりやすくなります。

注: 見分けることができるコンテンツのないファイルには、サムネイルを使用しないでください。代わりに、オブジェクトを表現し、関連付けられているアプリケーションや種類を示す、従来のシンボリックなファイルアイコンを使用します。

#### ツールバー

- ツールバーに表示されるアイコンでは、サイズ、色、複雑さなどにおいて、視覚的なバランスをとる必要があります。
- コンテキスト内でアイコンの候補をスクリーンショットで確認し、必要以上に目立つものや、バランスが悪いものの使用を避けます。
- スクリーンショットでテストすると、繰り返しのコードを作成するためにかかるコストを回避できます。
- アイコンはコード内でも確認します。アニメーションやその他の要因がアイコンの成功の鍵となることがあります。場合によってはさらに繰り返しの処理が必要となる場合があります。



上の例では、視覚的なバランスがとれていません。



コンテキストにおける繰り返しの処理を試します。

### ツリー ビュー

- ツリー ビュー コントロール内の階層を保つには、視覚的なバランスが必要となります。
- したがって、このコンテキストで一般的に使用されるアイコンは、ツリー ビュー内で評価する必要があります。形状によって他のアイコンよりも目立つ特定の  $16 \times 16$  アイコンは、小さめに作成することが要求される場合があります。
- 視覚的なアンバランスを補正することは、高品質なアイコンを作成する上で重要です。



## 標準アイコン

デザイン コンセプト

ガイドライン

全般

アイコンのサイズ

エラー アイコン

警告 アイコン

情報 アイコン

疑問符 アイコン

"標準アイコン" とは、Windows® に不可欠な要素である、エラー アイコン、警告 アイコン、情報 アイコン、疑問符 アイコンのことです。



標準のエラー アイコン、警告 アイコン、情報 アイコン、疑問符 アイコン

標準アイコンには次の意味があります。

- エラー アイコン。このユーザー インターフェイス (UI) は、発生したエラーまたは問題を示します。
- 警告 アイコン。この UI は、今後問題を引き起こす可能性のある状態を示します。
- 情報 アイコン。この UI は、有益な情報を示します。
- 疑問符 アイコン。この UI は、ヘルプのエントリ ポイントを示します。

標準アイコンが注目に値する理由は、タスク ダイアログ ボックス、メッセージ ボックス、バルーン、通知など、多くの Windows アプリケーション プログラミング インターフェイス (API) にこのアイコンが組み込まれているためです。標準のアイコンは、インプレース メッセージ やステータス バーでもよく使用されます。

注: [アイコン](#)に関するガイドラインは、別の項目として記載しています。

## デザイン コンセプト

適切な標準アイコンを選択するにはいくつかの要因を考慮する必要があります。アイコンの不適切な使用があまりにも多いのは、これらの要因が考慮されていないためでもあります。最もよくある間違いとしては次のものがあります。

- 軽度のエラーに警告 アイコンを使用する。警告はエラーを "やわらかく伝えるもの" ではありません。
- アイコンをまったく使用しない方が良い場合にも標準アイコンを使用する。すべてのメッセージにアイコンが必要なわけではありません。
- 軽度の問題に対して警告を表示したり、日常的な操作に対する質問を警告として表示することでユーザーを不安にさせる。このように警告を使用すると、プログラムが危険にさらされやすいもののように思われ、本当に重要な問題から注意がそらされてしまいます。

このセクションの残りの部分では、このようなよくある間違いを避けるために、標準アイコンをどのように考えれば良いかについて説明します。

## メッセージの種類と重要度

標準アイコンを選択する際には、基になる問題の重要度ではなく、メッセージの種類を基準にします。メッセージの種類は次のとおりです。

- エラー。発生したエラーまたは問題を示します。
- 警告。今後問題を引き起こす可能性のある状態を示します。
- 情報。有益な情報を示します。

したがって、エラー メッセージにはエラー アイコンを使用できますが、警告 アイコンは使用できません。

ん。軽度のエラーを "やわらかく伝える" ための方法として、警告アイコンを使用しないでください。重要度に関わらず、"不適切なフォントサイズ" はエラーであり、"この操作を続行すると家が火事になる" は警告です。

## 適切なメッセージの種類の決定

問題によっては、強調するポイントや表現の仕方により、エラー、警告、または情報のいずれでも表現できる場合があります。たとえば、現在の Windows® Internet Explorer® の設定により、ある Web ページにおいて署名なしの ActiveX コントロールを読み込むことができない状態は、次のどのメッセージでも表現できます。

- エラー。"このページに署名なしの ActiveX コントロールを読み込むことができません。" (既存の問題として表現されている)
- 警告。"Windows Internet Explorer が、署名なしの ActiveX コントロールを読み込むように設定されていないため、このページは予期したとおりに動作しない可能性があります。" または "このページで署名なしの ActiveX コントロールをインストールできるようにしますか? 信頼できない発行元からのコントロールをインストールすると、コンピューターに危害を与える可能性があります。" (どちらも、将来問題を引き起こす可能性のある状況として表現されている)
- 情報。"Windows Internet Explorer は、署名なしの ActiveX コントロールをブロックするように設定されています。" (単なる事実として表現されている)

適切なメッセージの種類を決定するには、ユーザーが認識または対処する必要のある、問題の最も重要な側面に焦点を当てます。通常、その問題があるためにユーザーが先に進めない場合はエラーとして示し、ユーザーが先へ進める場合は警告として示します。この点に基づいて、[メイン指示テキスト](#)またはその他の対応するテキストを作成し、そのテキストに一致するアイコン(標準またはその他のアイコン)を選択します。メイン指示テキストとアイコンは、常に調和している必要があります。

## 重要度

重要度は、エラー、警告、および情報アイコンのどれを選択するかを決定する際の考慮事項ではなく、標準アイコンの使用自体を決定する際の 1 つの要因となります。

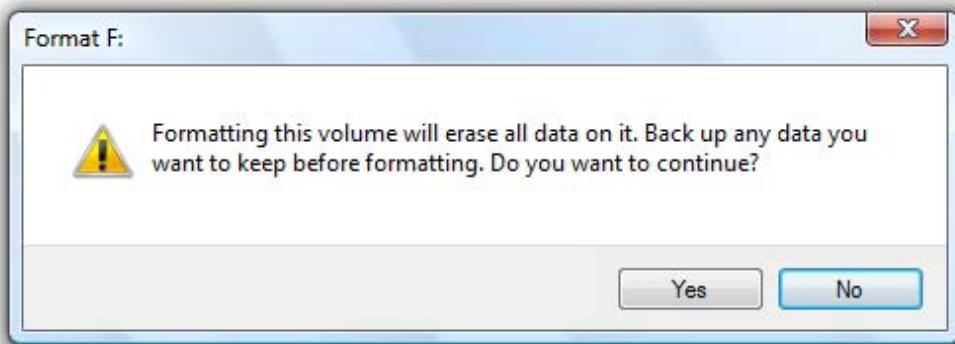
アイコンは視覚的な伝達に使用したときに最も効果的になります(アクセシビリティの理由により、この視覚的な伝達は常にテキストやサウンドといったその他の形式と共に使用する必要があります)。アイコンは、ユーザーが一見しただけで、情報の性質と応答したときの結果を理解できるものである必要があります。したがって、重大なエラーおよび警告と、正常時の状態の区別を付ける必要があります。重大なエラーおよび警告には、次の特徴があります。

- 次のうち 1 つ以上が失われる可能性がある
  - 値の高い資産(データ損失、金銭的な損失など)
  - システムへのアクセスまたは整合性
  - プライバシーまたは機密情報の制御
  - ユーザーの時間(30 秒かそれ以上の長時間)
- 予期しない、または意図しない結果をもたらす
- 今すぐ適切な対処が必要(間違いが簡単には修正できず、元に戻せない可能性もある)

重大なエラーおよび警告とそれ以外を区別するには、通常、重大ではないメッセージをアイコンなしで表示します。こうすると、重大なメッセージが注目され、それ以外のメッセージとの間に視覚的な区別が付けられます。これは、[Windows のトーン](#)とも整合性が取れています。

すべてのメッセージにアイコンが必要なわけではありません。アイコンはメッセージを修飾するための手段ではありません。

次に示すのは、前に挙げた特徴を満たした、重大な警告の良い例です。



この例では、重大な警告により、元に戻せないデータ損失の可能性があることをユーザーに伝えています。

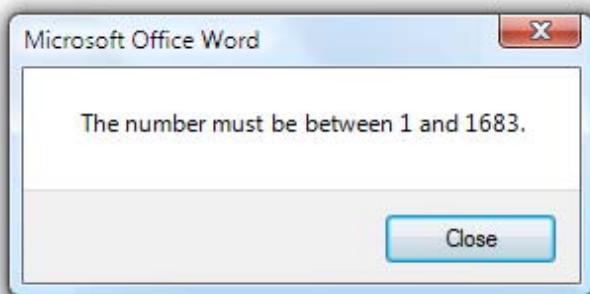
一方、次の例は重大な警告ではありません。意図的な操作である可能性が高く、結果は容易に元に戻すことができます。

間違った例:



この例の確認は重大ではありません。意図的な操作である可能性が高く、容易に元に戻すことができます。

一般的な UI では、ほとんどのエラーはユーザー入力のエラーと関係しています。ほとんどのユーザー入力のエラーは容易に修正できるため重大ではなく、ユーザーは続行する前に修正する必要があります。また、軽度のユーザーの間違いについて過剰に注意を引くことは、Windows のトーンに反します。したがって、軽度のユーザー入力のエラーは、通常、エラーアイコンなしで表示されます。重大ではないことを強調するために、このような問題をユーザー入力の問題と呼びます。



この例では、この軽度のユーザー入力の問題は重大ではないため、ダイアログ ボックスでの表示にアイコンは必要ありません。

#### 過剰な警告を避ける

Windows プログラムでは過剰な警告が見られます。一般的な Windows プログラムでは、外見上いたるところに警告アイコンが置かれ、ほとんど重要とは言えないことについて警告しています。プログラムによっては、ほとんどすべての質問が警告として表示されているものもあります。過剰に警告を表示すると、プログラムを使用することが危険な行為のように感じられ、本当に重要な問題から注意がそらされてしまいます。

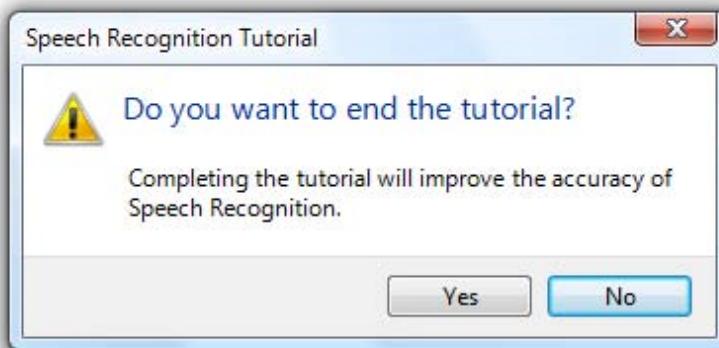
単にデータ損失の可能性があるだけでは、警告アイコンを使用する理由として不十分です。この可能性

に加えて、望ましくない結果が起こった場合、それが予期または意図しない結果で、容易に修正されない結果であるということも条件になります。以上のことと条件にしないと、不適切に回答された質問はほとんどすべて、何らかのデータ損失を招き、警告アイコンに値するという意味につながります。

本当に重大な問題のみに警告アイコンを表示するには、以下のことを検討します。

- 問題にユーザーの注意を高める正当な理由があるか。[日常的な操作に対する確認](#)および質問には、警告アイコンを使用しません。
- 警告アイコンの結果としてユーザーが行動を変える可能性が高いか。ユーザーがより慎重に答えを検討する可能性が高いかどうかを考えます。

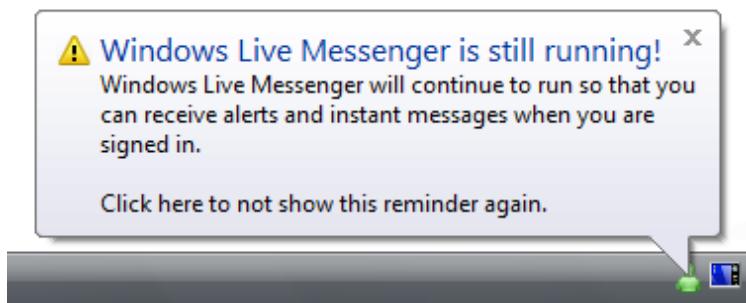
間違った例:



この例では、警告アイコンがあっても、ユーザーが質問への回答を変える可能性はありません。

- 重要な対応または意思決定が発生するかどうか。対処方法のない警告は、ユーザーの混乱を招くだけです。

間違った例:



この通知を警告にする必要はありません。ユーザーは何を要求されているのか(さらには何に注意すればよいのか)わかりません。

## コンテキスト

コンテキストも標準アイコンを使用する際の考慮事項です。これは、コンテキスト自体が情報の伝達になるためです。以下はその具体例です。

- ダイアログ ボックス(タスク ダイアログ ボックスおよびメッセージ ボックスを含む)および通知において重大ではないエラーを示す場合、アイコンは必要ありませんが、インプレース エラーには常にエラー アイコンが必要です。アイコンを表示しない場合、このようなモードレス フィードバックは容易に見過ごされてしまいます。
- インプレース警告には、通常のテキストと区別するために常に警告アイコンが必要です。
- ダイアログ ボックス、通知、およびバブルでは明確に情報が表示されるため、情報アイコンは必要ありません。一方、バナーのようなモードレス フィードバックは見過ごしやすいので、バナーには 16 × 16 ピクセルの情報アイコンまたはその他のアイコンが必要です。

コンテキストはアイコンの使用における重要な要因です。したがって、ここでは標準アイコンのガイドラインをコンテキストの観点から説明します。

## 標準アイコンの妥当性の評価

UI テキストを評価するときは、標準アイコンも含めて読みます。つまり、エラー アイコンは "エラー"、警告アイコンは "警告、十分な注意が必要"、情報アイコンは "注目" のように読み、続いて、残りのコンテキストであるメイン指示テキスト、コンテンツ エリア、コミット ボタンなどを読んでいきます。それぞれの標準アイコンの意味やトーンが、そのコンテキストの意味やトーンと一致していることを確認します。一致していない場合は問題です。

### 最も重要な点

それぞれの標準アイコンの意味やトーンが、そのコンテキストの意味やトーンと一致していることを確認します。一致していない場合は、アイコンを変更するか削除します。

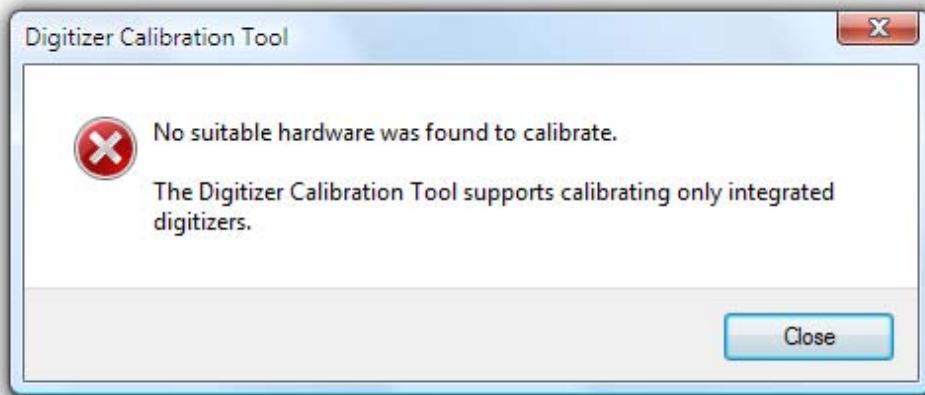
## ガイドライン

注: 以下のガイドラインでは、"インプレース" とはウィザードのコンテンツ エリア内、プロパティ シート内、コントロール パネル アイテムのページ内などの、通常のウィンドウ画面上にあることを指します。

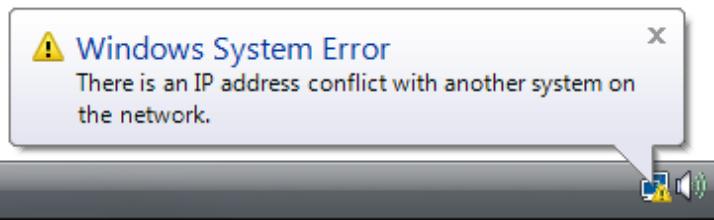
### 全般

- 標準アイコンを選択する際には、その問題の重要度ではなく、メッセージの種類を基準にします。
  - エラー。発生したエラーまたは問題を示します。
  - 警告。今後問題を引き起こす可能性のある状態を示します。
  - 情報。有益な情報を示します。
- 問題が複数のメッセージの種類に該当する場合は、ユーザーが対処する必要のある最も重要な側面に焦点を当てます。
- アイコンは、メイン指示テキストまたはその他の対応するテキストと常に一致する必要があります。

正しい例:



間違った例:



間違った例では、標準の警告アイコンと、エラーを示すメイン指示テキストとが一致していません。

### アイコンのサイズ

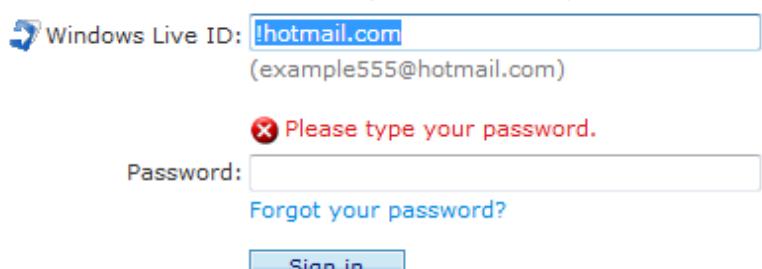
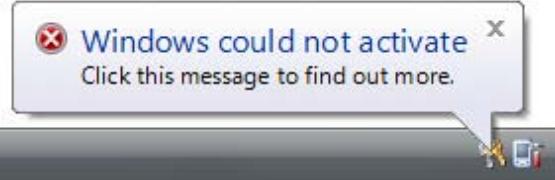
- コンテキストに応じて、標準アイコンのサイズを選択します。

コンテキスト	使用する場合
ダイアログ ボックス	コンテンツ エリアのアイコンには $32 \times 32$ ピクセル、脚注エリアのアイコンには $16 \times 16$ ピクセルを使用します。
インプレース	エラー ページには $32 \times 32$ ピクセル、それ以外には $16 \times 16$ ピクセルのアイコンを使

	用します。
通知	16 × 16 ピクセルのアイコンを使用します。
バルーン	16 × 16 ピクセルのアイコンを使用します。
バナー	16 × 16 ピクセルのアイコンを使用します。

## エラー アイコン

- エラー アイコンはエラーまたは問題が発生したときにだけ使用します。

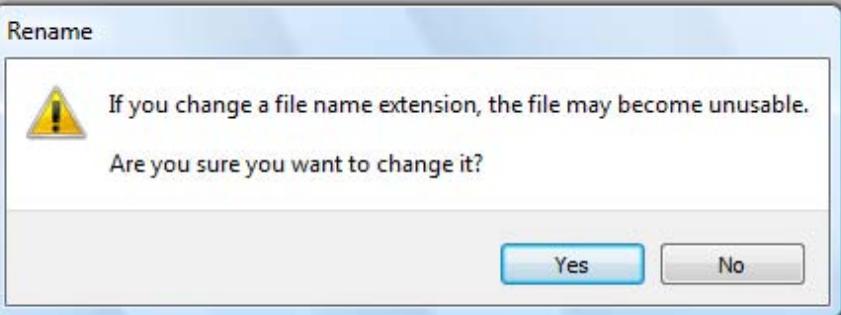
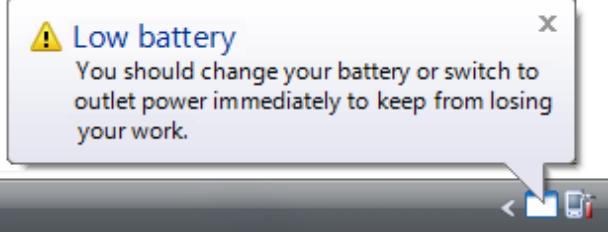
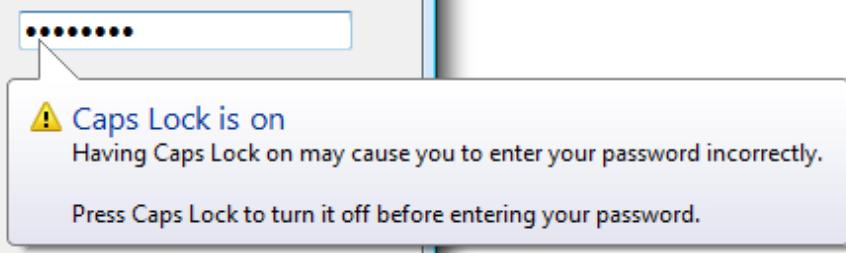
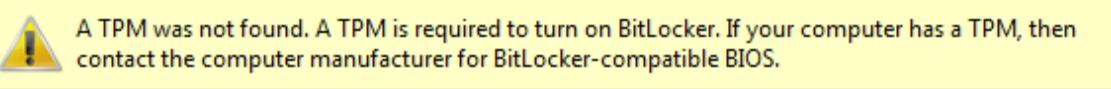
コンテキスト	使用する場合
ダイアログ ボックス	重大なエラーのみに使用します(重大ではないエラーには標準アイコンを使用しません)。
	
インプレースエラー	すべてのエラーに使用します。
	
通知	重大なエラーのみに使用します(操作の失敗の場合)。
	
バルーン	使用しません。バルーンは重大な問題には使用しません。重大ではないエラーの場合も、バルーンにエラー アイコンは必要ありません。
バナー	使用しません。バナーはエラーに使用しません。

一般に、重大ではないユーザー入力の問題にエラー アイコンは必要ありません。ただし、インプレースエラーにはアイコンが必要です。このようなコンテキストでのフィードバックは、アイコンがなければ容易に見過ごされてしまいます。

- タスク ダイアログ ボックスでは、脚注のエラー アイコンは使用しません。エラー アイコンが使用できるのはコンテンツ エリアのみです。

## 警告アイコン

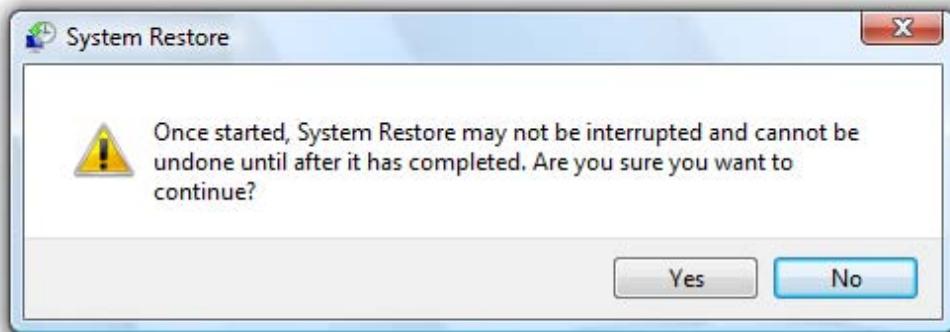
- 警告アイコンは、今後問題を引き起こす可能性のある状態のときにだけ使用します。

コンテキスト	使用する場合
ダイアログボックス	すべての警告に使用します。 
インプレース警告	テキストを警告として示すために使用します。 
通知	すべての警告に使用します (重大ではないシステムイベントの場合)。 
バーゲン	特殊な状態に使用します。 
バナー	バナーに注目させるために使用します。 

- 重大ではないエラーを "やわらかく伝える" 目的では、警告アイコンは使用しません。エラーは警告ではありません。エラー アイコンのガイドラインに従ってください。

質問のダイアログ ボックスでは、重大な結果を伴う質問に対してのみ警告アイコンを使用します。日常的な操作に対する質問には警告アイコンを使用しません。

正しい例:

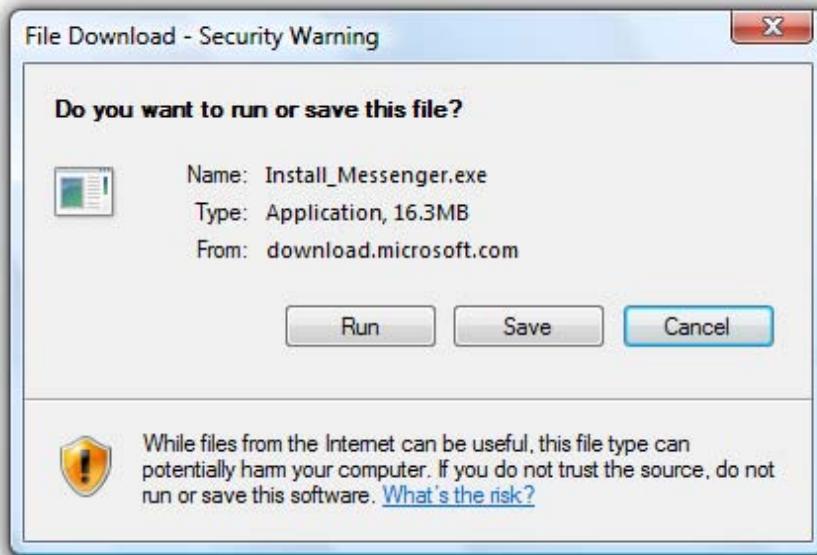


間違った例:



間違った例では、日常的な操作に対する質問に、警告アイコンが不適切に使用されています。

- タスク ダイアログ ボックスでは、危険な結果をユーザーに警告するために、脚注の警告アイコンを使用できます。ただし、警告アイコンはコンテンツ エリアと脚注エリアのどちらかでのみ使用し、両方で使用しないようにします。



この例では、黄色のセキュリティ シールドが脚注で使用されています。

#### 情報アイコン

- 情報アイコンは、コンテキストで明確に情報が示されないときにだけ使用します。

コン テキ スト	使用する場合
----------------	--------

ダイアログ ボックス	使用しません。
インプレース	使用しません。代わりに、プレーンな静的テキストまたはバナーを使用します。
通知	使用しません。
バルーン	使用しません。
バナー	バナーに注目させるために使用します。

 For your security, some settings are controlled by Group Policy

ダイアログ ボックス、通知、およびバルーンでは、ユーザーに情報を提供していることがコンテキストにより十分に伝わるため、情報アイコンは必要ありません。

- タスク ダイアログ ボックスでは、脚注の情報アイコンは使用しません。脚注は容易に見つけることができ、情報であることは明白です。

#### 疑問符アイコン

- 疑問符アイコンは、ヘルプのエントリ ポイントとしてのみ使用します。詳細については、[ヘルプのエントリ ポイント](#)のガイドラインを参照してください。
- 質問に疑問符アイコンは使用しません。疑問符アイコンは、ヘルプのエントリ ポイントとしてのみ使用します。質問に疑問符アイコンは一切必要なく、メイン指示テキストを質問として提示すれば十分です。
- 疑問符アイコンを無条件に警告アイコンに置き換えることはしません。疑問符アイコンを警告アイコンに置き換えるのは、質問が重大な結果をもたらす場合に限ります。それ以外の場合は、アイコンを使用しないようにします。

## グラフィック要素

適切なユーザーインターフェイスかどうかの判断基準

使用パターン

ガイドライン

全般

グラフィックデザイン

背景とバナー

グラス

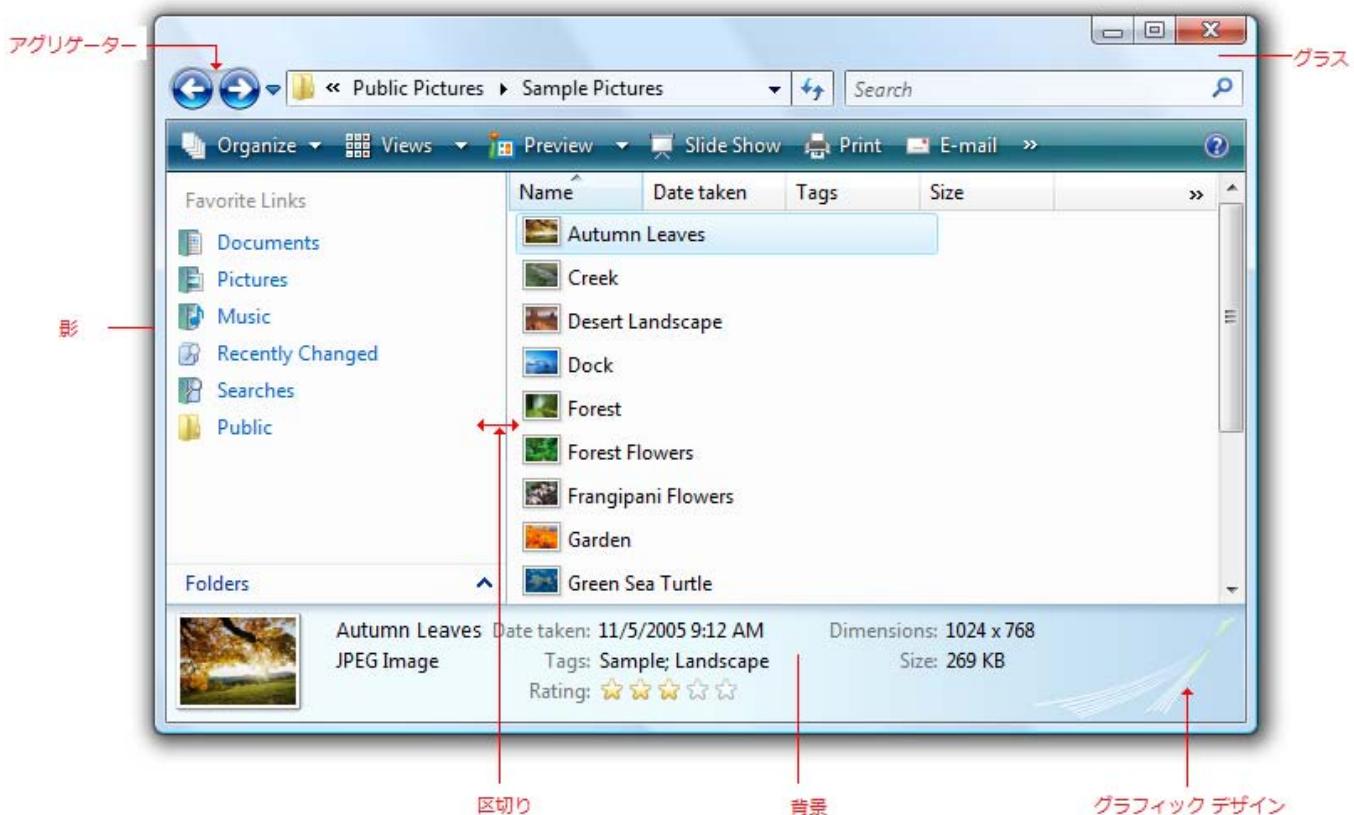
区切り記号

影

高dpiサポート

テキスト

"グラフィック要素"を使って、関連性、階層、強調を視覚的に表現できます。グラフィック要素には、背景、バナー、グラス、アグリゲーター、区切り記号、影、ハンドルなどがあります。



### グラフィック要素の種類の例

グラフィック要素は、通常、対話型ではありません。ただし、区切り記号は、サイズ変更可能なコンテナの場合は対話的に操作可能です。対話的に操作可能であることはハンドルのグラフィックで示されます。

注: グループボックス、アニメーション、アイコン、ブランド化に関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

グラフィック要素は関連性を示す有効な視覚的手段ですが、使いすぎると見た目が乱雑になり、利用できるスペースが減ります。そのため、控えめに使用する必要があります。

Microsoft® Windows® の設計は、不要なグラフィックや線を排除して、外観を簡素にすっきりさせる方向に向かっています。

グラフィック要素が必要かどうかは、以下の点に基づいて判断します。

- グラフィック要素なしでも、その設計の提示方法と情報伝達方法が明確で効果的か。該当する場合は、グラフィック要素を削除します。
- レイアウトだけで効果的に関連性を伝えることができるか。該当する場合は、代わりにレイアウトを使用します。コントロールが互いに関連する場合は並べて配置し、関連しない場合は余分に間隔を空けます。インデントによって階層関係を示すこともできます。



## System and Maintenance

Get started with Windows

Backup your computer



## Security

Check for updates

Allow a program through Windows Firewall



## User Accounts

Change account type



## Appearance and Personalization

Change desktop background

Customize colors

Adjust screen resolution

この例では、レイアウトだけを使用してコントロールの関連性を示しています。

- テキストなしで情報が効果的に伝わるか。該当しない場合は、**グループボックス**、ラベル付き区切り記号、その他の**ラベル**などを使用します。

## 使用パターン

グラフィック要素にはいくつかの使用パターンがあります。

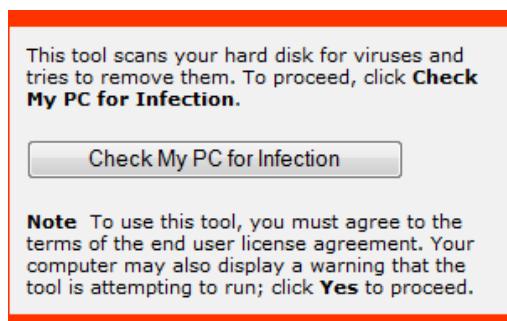
**イラスト**　任意のサイズにできること、また通常は対話型でないことを除けば、イラストはアイコンに似ています。  
概念を視覚的に伝えます。



この例では、イラストで機能が示唆されています。

**背景**　背景を使用して、重要なコンテンツを強調できます。

異なる種類のコンテンツを強調したり、逆に目立たなくしたりします。



この例では、背景を使用して重要なタスクを強調しています。

背景を使用して、2次的なコンテンツを目立たないようにすることもできます。



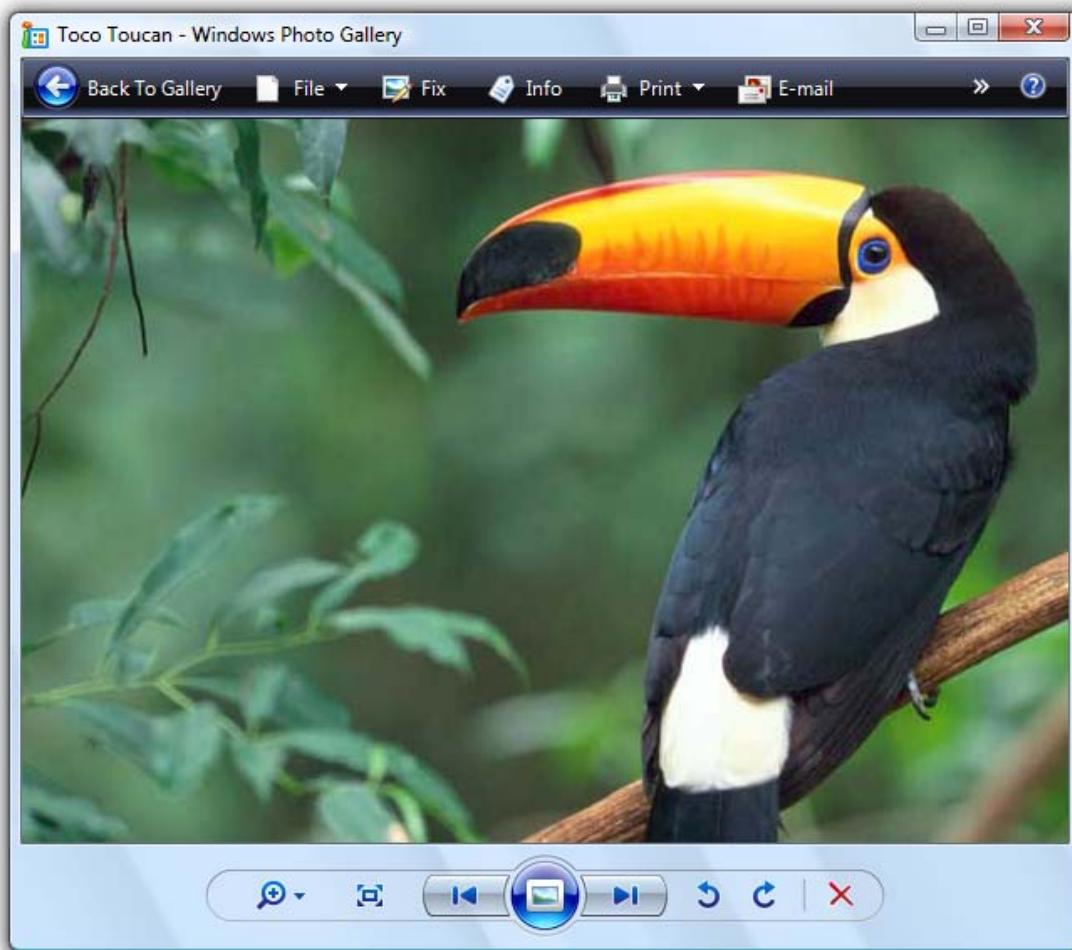
バナー  
重要な状態を示すために使用します。

この例では、2次的なタスクを作業ウィンドウ内に配置して目立たなくしています。

背景とは対照的に、バナーでは主に単一のテキスト文字列が強調されます。

グラス  
ウィンドウを軽快な外観にする手段として使用します。

グラスを使用してウィンドウ自体ではなくコンテンツに重点を置くことで、軽快な外観にできます。

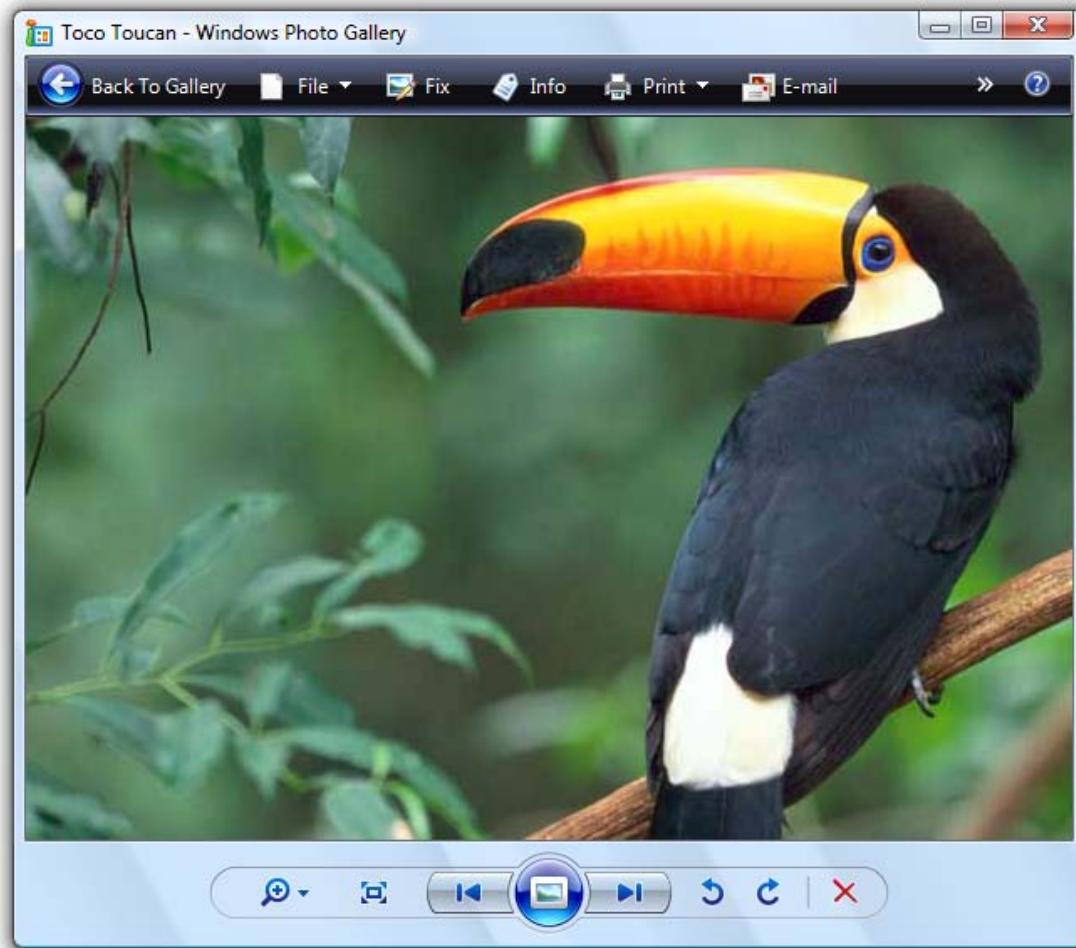


この例では、グラスを使用して、コントロールではなくコンテンツにユーザーの注意を引き付けています。

アグリゲーター  
関連性の高いコントロール間で、関連性を視覚的に表現します。



この例では、アグリゲーターの背景を使用して、エクスプローラーの[戻る]ボタンと[進む]ボタンの間の関連性を強調しています。



この例では、境界線アグリゲーターを使用して、コントロール間の関連性を強調し、8つではなく単一のコントロールであるかのように表示しています。

- 区切り記号 関連の弱いコントロールや関連しないコントロールを分割します。
- 区切り記号は、対話型にも非対話型にもできます。サイズ変更可能なコンテンツ間の対話型区切り記号は"スプリッター"と呼ばれます。



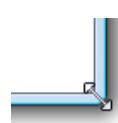
この例では、対話型区切り記号がサイズ変更可能なコンテンツに使用されています。

- 影 背景に対してコンテンツが浮き出でて見えるようにします。



この例では、影を使用して背景に対してアートワークを浮き出させています。

- ハンドル オブジェクトの移動やサイズ変更が可能なことを示します。



UX Guide

これらの例では、オブジェクトのサイズを変更できることがハンドルで示されています。

## ガイドライン

### 全般

- 重要な情報をグラフィック要素だけで伝えようとしないでください。視覚障害のあるユーザーにアクセシビリティの問題が生じます。

### グラフィック デザイン

- グラフィックは、1つのシンプルな概念を補強している場合に最も効果的です。解釈に頭を使う複雑なグラフィックは効果的ではありません。象形文字は、洞窟の壁画にしか向いていません。

間違った例:



この Windows XP からの例では、信頼に関する複雑な決定を複雑なグラフィックで説明しようとしていますが、うまく伝わっていません。

- 対話型コントロールを連想させる矢印、シェブロン、ボタン枠などのアフォーダンスは使用しないでください。このようなグラフィックを使用すると、ユーザーはグラフィックに対して操作を行おうとします。
- 赤、黄、緑の純色は、デザインに使用しないようにします。混乱を避けるために、これらの色は状態を伝えるためにのみ使用します。状態を伝える目的以外に使用する必要がある場合は、純色ではなく控えめな色調の色を使用します。
- 文化的に中立なデザインを使用します。1つの国、地域、または文化で特定の意味を持っている場合でも、別の国、地域、または文化では別の意味を持つことがあります。
- 宗教、政治、国家のシンボルはもちろんのこと、人、顔、性別の表現、身体部分などもグラフィック要素での使用は避けます。このような画像は、簡単に解釈できなかつたり、不快感を与えることがあります。
- 人やユーザーを表現する必要がある場合は、一般化して描写します。リアルな描写は避けてください。

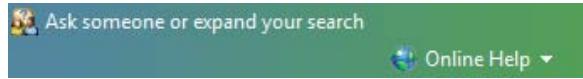
### 背景とバナー

- コンテンツを強調するには、明るい背景に濃い色のテキストを使用します。薄い灰色や黄色の背景に黒いテキストを使用すると効果的です。



この例では、リンクが黄色の背景上にあり、ユーザーの注意を引き付けています。

- コンテンツを目立たなくするには、暗い背景に明るい色のテキストを使用します。濃い灰色や青い背景に白いテキストを使用すると効果的です。



この例では、濃い色の背景でコンテンツが目立たなくなっています。

- グラデーションを使用する場合は、テキストの色がグラデーション範囲全体と程良いコントラストになるようにします。
- バナーに注意を引き付けるには、常に 16 × 16 ピクセルのアイコンを使用します。これ以外のサイズのアイコンでは、バナーが見落とされやすくなります。その他のガイドラインと例については、「[標準アイコン](#)」を参照してください。
- 背景とバナーの使用は慎重にします。背景やバナーをコンテンツの強調に使用すると、かなり頻繁に逆の結果(「バナー ブラインドネス」として知られる現象)が起きます。

### グラス

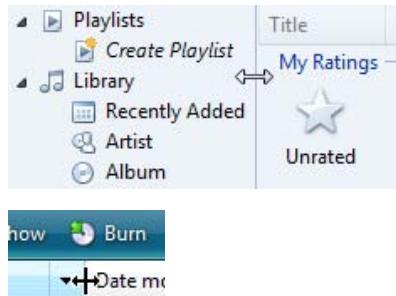
- ウィンドウ枠に接する、テキストを含まない小さい領域で、その効果を十分検討した上でグラスを使用するようにします。こうす

ることで、その領域が枠と一体化し、プログラムに簡素で軽快、かつまとまりのある外観を与えることができます。

- 通常のウィンドウ背景の方が魅力的であったり、使いやすかったりする状況では、グラスは使用しないでください。

## 区切り記号

- 区切り記号には、縦線や横線を使用します。区切り記号と分割されるコンテンツの間には、十分なスペースを確保します。
- サイズ調整可能なコンテンツ間の区切り記号(スプリッター)には、ポイント時にサイズ変更のポインターを表示します。



これらの例では、ポイント時にサイズ変更のポインターが表示されています。

## 影

- ドラッグの対象となる、プログラムの最も重要なコンテンツやオブジェクトを、背景から浮き出して見えるようにします。これ以外の状況で影を使用すると、乱雑さの元になり得るので注意します。

## 高dpi サポート

- 96 および 120 dpi のビデオ モードをサポートします。起動時に dpi モードを検出し、dpi 変更イベントを処理します。Windows は 96 dpi および 120 dpi に最適化されており、既定で 96 dpi が使用されます。
- グラフィックを拡大縮小するのではなく、96 dpi 用と 120 dpi 用に別々のビットマップを用意することをお勧めします。少なくとも、最も重要な可視ビットマップについては 96 dpi と 120 dpi の両バージョンを用意し、他のサイズについては中央配置や拡大縮小で対応します。このようなアプリケーションは "高 dpi 対応" と見なされ、Windows によって自動的に拡大縮小されるプログラムよりも、全体的に優れた視覚効果が提供されます。
  - 開発者向け情報: 高dpi 対応フラグをプログラムのマニフェストに設定するか、またはプログラムの初期化中に SetProcessDPIAware() API を呼び出して、プログラムを高dpi 対応として宣言すれば、自動の拡大縮小を回避できます。また、マクロを使用して、正しいグラフィックの選択を単純化できます。Win32 ビットマップの場合は、SS\_CENTERIMAGE を使用して中央に表示したり、SS\_REALSIZECONTROL を使用して拡大縮小を行うことができます。
- プログラムを 96 dpi および 120 dpi の両方でチェックし、以下の問題がないか確認します。
  - 小さすぎたり、大きすぎるグラフィック
  - 切り詰められたり、重複したり、またはそれ以外の適切に収まっていないグラフィック
  - 画素が粗く見えるまで不適切に引き伸ばされたグラフィック
  - グラフィックの背景で切り詰められたテキスト、またはグラフィックの背景に収まっているテキスト

## テキスト

- アクセシビリティとローカライズ性を確保するため、グラフィック内にはテキストを使用しません。例外は、[ブランド化](#)の表現および抽象的概念としてのテキストのみとします。

## サウンド

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

使用方法

再生

サウンドの選択

Windows のシステム サウンド

サウンド デザイン

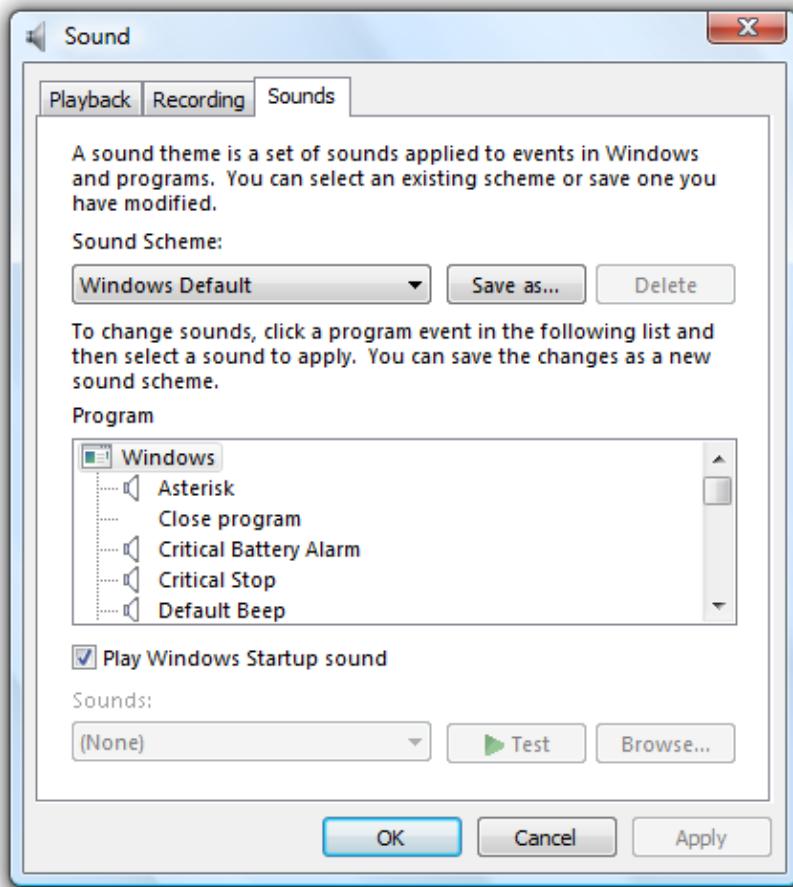
ミキシング

Windows との統合

DirectSound プログラミングに関する問題

テキスト

"サウンド" とは、ユーザー エクスペリエンスのオーディオ要素のことです。サウンドは、適切に使用すればユーザーとの言葉によらない関係を確立できる効果的な伝達手段となります。場合によっては心理的な共感を得られることもあります。サウンドは単独でも、視覚的な UI の補足としても使用できます。たとえば、通知にサウンド効果を追加すると、イベントの発生時にユーザーが画面を見ていない場合なども、ユーザーが気付く可能性が高まります。



コントロールパネルの [サウンド] を選択して表示されるダイアログ ボックスの [サウンド] タブで、ユーザーはシステム サウンドに変更を加えることができます。

ここでは、プログラム内でイベントやユーザー操作への応答(効果音)としてサウンドを使用する方法と、プログラムのサウンド制御を Windows と統合する方法について説明します。音楽や音声の使用については説明しません。

注: 通知およびブランド化に関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

サウンドを使用するかどうかは、以下の点に基づいて判断します。

- サウンドを使用することでユーザーに明確なメリットがあるか。サウンドの使用によるデメリットがメリットを

上回ることも多々あるため、明確なメリットがあるときだけサウンドを使用します。

- サウンドの使用が適切かどうか。サウンドを使用して注意を引く対象が注目に値するものかということや、サウンドがなければユーザーは物足りない気がするかということを考えます。ユーザーに情報が常に届くようにするためのサウンド、ユーザーがそれを耳にして行動を変える可能性が高いサウンド、または有益なフィードバックを提供するサウンドとなるように気をつけます。
- サウンドの使用がユーザーの邪魔になるか。頻繁に再生されたり、音量が大きかったり、耳障りにならないかを考えます。サウンドを使用した結果、ユーザーがシステムの音量やプログラムの音量を下げる事になる場合、そのサウンドは不適切です。
- サウンドを一次的伝達手段として使用しているか。聴覚障害のあるユーザーに対してなど、サウンドを一次的伝達手段として使用することが不適切なケースは多くあります。サウンドは、テキストや視覚的要素といった他の伝達手段の補助として、より大きな効果を発揮します。
- 主な対象ユーザーが IT プロフェッショナルであるかどうか。サウンドは、通常、IT プロフェッショナルを対象とするタスクには効果的ではありません。そのようなタスクの多くは、無人で実行されます。また、サウンドは IT プロフェッショナルの作業規模に対して適切ではありません。一度に何百ものタスクを実行し、完了または失敗したときにサウンドが再生されるのを想像してみるとわかります。

## デザイン コンセプト

一般的に、サウンドは次の一部またはすべての目的を達成できます。

- 通知。サウンドは特定のイベントと関連付けることができます。たとえば "新しいメール" サウンドは、現在のタスクを中断させることなくメールの到着をユーザーに知らせます。
- フィードバック。サウンドを使用すると、特定のユーザー操作に対しフィードバックを提供できます。たとえば、ボリュームコントロールのつまみを放したときに再生される短いサウンドは、現在の設定レベルについてのフィードバックになります。
- ブランド化。サウンドを特定のコンテンツと関連付けて、製品、アプリケーション、またはサービスをブランド化できます。Windows® では、オペレーティングシステムのスタートアップにブランド化のサウンドが使用されています。
- エンターテインメント。サウンドは、エンターテインメント製品を強化するためや、あらゆる製品をより魅力あるものにするためによく使用されます。たとえば、ほとんどのゲーム、教育アプリケーション、一般消費者向け製品では、ユーザーを楽しませ、ユーザー エクスペリエンスを高めるためにサウンドが使用されています。

ある種のサウンドでは、これらの目的のいくつかを一度に達成できます。たとえば Windows スタートアップのサウンドは、スタートアッププロセスが完了しデスクトップが使用できるようになったことを伝えると同時に、製品のブランド化を効果的に行い、わずかの間でもユーザーの注意を引きます。

これらの目的をどれも満たさないサウンドは、おそらく削除するのが適切です。

## サウンドの不適切な使用

サウンドにはメリットがあるものの、適切に使用するにはかなり控えめにする必要があります。そうしないと、ユーザーに不快感を与えるプログラムになる可能性があります。不適切に設計された耳障りなサウンドが頻繁に繰り返し再生され、ユーザーが不快に思うようになると、ユーザーはサウンドを完全に切ってしまいます。この原因の 1 つは、注目を要求し、無視するのは難しいというサウンドそのものの特性にあります。適度なバランスを見つけるためのヒントについては、[サウンド デザイン ガイドライン](#) を参照してください。

サウンドの使用によるデメリットがメリットを上回ることも多々あるため、明確なメリットがあるときだけサウンドを使用します。疑わしい場合は、サウンドは使用しません。

## サウンドの補助的な使用

サウンドを適切に使用したとしても、次のように、サウンドがすべてのユーザーに効果的とは限らない場合も多くあります。

- ユーザーが、サウンドが聞こえないような騒がしい環境で操作している場合
- ユーザーが、サウンドを切るか音量を下げなければならないような静かな環境で操作している場合
- ユーザーが聴覚に障害を持っている場合
- コンピューターにスピーカーがない場合

このような理由から、通知およびフィードバックに使用されるサウンドは、唯一の伝達手段とならない

ようになります。これらのサウンドは、視覚的要素またはテキストによる合図を補足するものとして使用します。

## 望ましいサウンドの特性

基本的に、次の特性を満たすサウンドが好まれます。

- 中～高周波数 (600 ヘルツ [Hz] ~ 2 キロヘルツ [kHz])
- 短い (1 秒未満)
- 中程度かそれよりやや控えめな音量
- 明確な意図がある
- 心地良く、不安な音や耳障りな音に聞こえない
- 言葉を使わない
- 繰り返さない

サウンドに関して言えば、少ない方がより効果的です。理想的なサウンド効果とは、ユーザーがかろうじて気付く程度ではあるものの、なければ物足りない気がするようなサウンド効果です。

よくある誤解として、重要なイベントに対するサウンドは、ユーザーの注意を引くために大音量で目立つようにしなければならないという考えがあります。サウンドは補助的な伝達手段である必要があるため、これは正しくありません。重要なエラー メッセージの場合は、提示方法 (モーダル ダイアログ ボックスでの提示など)、アイコン (エラー アイコン)、テキスト、トーンのすべてが組み合わさって、どのようなエラーであるかを伝えます。効果的なエラー サウンドにするには標準的な Windows のサウンドよりもやや音量が大きくてもかまいませんが、著しく大きくする必要はありません。

## Windows のサウンドの特性

Windows のサウンドは、そのシンプルさの基本要求を満たす以上のものになっています。Windows のサウンドは、軽やかで澄んだトーンを使った、透明感のある吹き抜けていくようなサウンドです。ゆるやかにフェードイン/フェードアウトする (開始と終了をぼかす) ことで、唐突で耳障りで衝撃的になるのを防いでいます。Windows のサウンドはどれも、控えめで、穏やかで、協和的な印象を与えるようにデザインされており、エコー、リバーブ、イコライザーを使用して、自然で溶け込むような効果を出しています。

Windows の既定のサウンド設定は、全体的に、楽器や聞き覚えのある日常的な音を使用せず、特徴的、音楽的すぎないようになっています。たとえば、ピアノや打楽器などの楽器、動物の鳴き声、環境にある騒音、音声、声、映画のような効果音、その他人間の発する音は避けられています。また、音楽として (長い、音を組み合わせたメロディとして) 感じられないように考慮されているので、他の種類のサウンドとは機能的に区別が付けられるようになっています。

Windows のサウンドは、サウンドの望ましい特性を備え、幅広いユーザーにアピールするように専門的にデザインされています。したがって、適切なときはいつでも Windows 内蔵のサウンドを使用することを検討してください。

## 独自のサウンド デザイン

独自のサウンドを作成する必要がある場合は、前で説明した特性を備え、関連するタスクまたはイベントを補足するものとなるようにデザインします。

独自のサウンドをうまく作成するのは、幅広いユーザー向けのサウンドの場合は特に、難しいということを理解してください。サウンドは好き嫌いの分かれやすいデザイン要素であり、1つのサウンドを好きだというユーザーもいれば、それを嫌いだというユーザーも多くいると考えられます。

プログラムに使用するサウンドは、グループとして、1つのテーマに沿った関連バリエーションであると感じられるようにデザインします。プログラムの聴覚的なエクスペリエンスは視覚的なエクスペリエンスと調和させる必要があります。また、サウンドの "トーン" は [テキストのトーン](#) と調和させる必要があります。心地良い自然なトーンのテキストに、耳障りで不安にさせるサウンドが添えられると、テキストの効果が損なわれます。

### 4 つの重要な点

1. サウンドは控えめに使用します。全体としてユーザーに明確なメリットがある場合にだけ使用するようにします。疑わしい場合は、サウンドは使用しません。

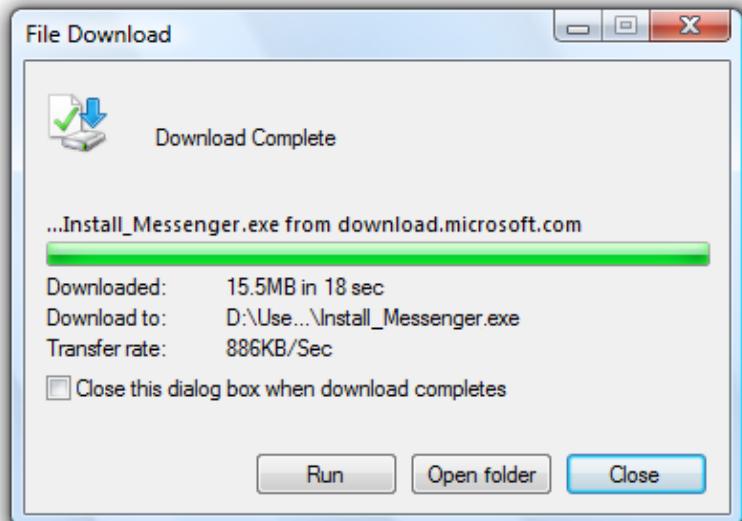
- 適切なときはいつでも Windows 内蔵のサウンドを使用します。
- 独自のサウンドをデザインする場合は、望ましいサウンドの特性を備え、全体として 1 つのテーマに沿ったバリエーションと感じられるようなサウンドにします。
- サウンドはユーザーの注意を引くために大音量で目立つようにしなければならないという考えは捨てます。

## 使用パターン

サウンドにはいくつかの使用パターンがあります。

### 操作の完了

ユーザーが開始した時間がかかる操作が正常に完了したときに、ユーザーに音で通知します。



この例では、ユーザーにダウンロードが完了したことを通知するために、ダイアログ ボックスでサウンドが再生されます。

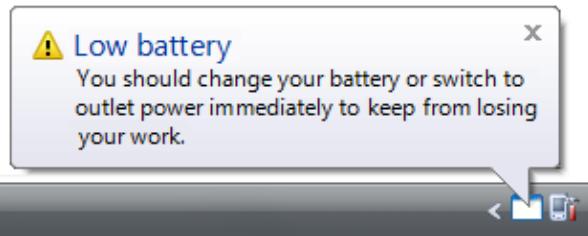
### 操作の失敗

ユーザーが開始した時間がかかる操作が失敗したときに、ユーザーに音で通知します。



この例では、ユーザーにバックアップ操作が失敗したことを通知するために、Windows でサウンドが再生されます。

**重要なシステムイベント**  
重要なシステムイベントまたは緊急の注意を必要とする状態を、ユーザーに音で警告します。



この例では、バッテリが少なくなり緊急に対処する必要があることを、ユーザーに警告しています。

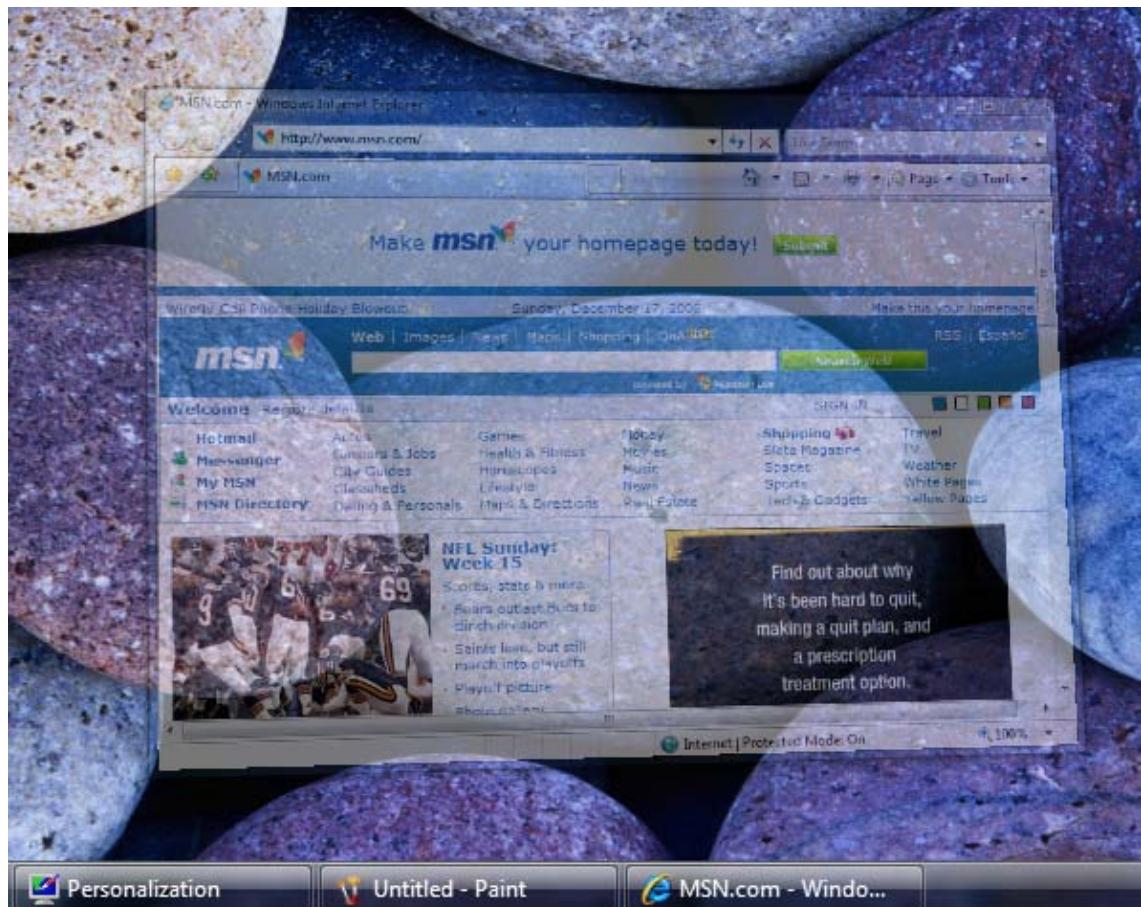
**FYI (参考)**  
有用で関連性が高いと考えられる情報を、ユーザーに音で通知します。

この情報は、通常、緊急の注意を必要としません。したがって FYI サウンドは、ユーザーの作業を中断しない控えめなフィードバック サウンドとなっています。



この例では、連絡先一覧に含まれるユーザーがインスタントメッセージングサービスへサインインしたときに、サウンドが再生されます。

**サウンド効果** ディレクト操作に対して適切なサウンド(リアルなサウンドまたは形式化されたサウンド)によるフィードバックを行います。よくあるサウンド効果としては、ユーザーが現実世界のオブジェクトを実際に操作しているかのような効果が挙げられます。これは、フォーリーとも呼ばれるサウンド効果です。  
ユーザー操作に  
対し、音による  
フィードバック  
を行います。



この例では、ウィンドウの最小化のサウンド効果として、現実世界のオブジェクトがサイズ縮小されているかのような音が再生されます。

**ブランド化サウンド** ブランド化サウンドは、視覚的なイベント、特にプログラム ウィンドウの表示のような UI の切り替え効果などと同期して再生するのが最適です。本物のサウンド ブランドは、商標登録された語やロゴのように製品の提供元を表し、比較的の独自性があります。  
心理的なインパクトを通じて  
ユーザー エクスペリエンスを強化し、副次的に製品のブランドをアピールするために提供されるサウンドです。



この例では、Windows スタートアップに、ブランド化された切り替え効果のエクスペリエンスが使用されています。

## ガイドライン

### 使用方法

- サウンドは控えめに使用します。全体としてユーザーに明確なメリットがある場合にだけ使用するようにします。ユーザーに情報が常に届くようにするためのサウンド、ユーザーがそれを耳にして行動を変える可能性が高いサウンド、または有益なフィードバックを提供するサウンドとなるように気をつけます。疑わしい場合は、サウンドは使用しません。
- 使用方法に応じて、サウンドとその特性を選択します。各使用パターンの説明については、前のセクションを参照してください。
- サウンドを、通知およびフィードバックのための唯一の伝達手段としては使用しません。サウンドは、視覚的要素またはテキストによる合図を強調するための補足手段として使用します。そうすることで、ユーザーはサウンドを聞くことができない場合でも情報を確実に得ることができます。
- 大きいサウンドや耳障りなサウンドを頻繁に再生することはしません。このような効果は不要であり、質の悪いユーザー エクスペリエンスをもたらします。再生頻度の高いサウンドほど、目立たなくする必要があります。注意を引くためにサウンドを大音量で目立つようにする必要はありません。
- ビープ音は使用しません。ビープ音は今日のプログラムには適しません。ビープ音には特定の意味を割り当てることができず、ユーザーはビープ音を制御できません。
  - 例外: 重要なシステム機能によって、緊急の対応を必要とする状況をユーザーに警告するためにビープ音が使用されることがあります。バッテリ残量の危機的な低下などがその例です。

### 再生

- サウンドを 3 回以上連續して繰り返すことはしません。
- 関連するサウンドイベントが連続する場合は、最初のイベントでのみサウンドを再生します。複数のサウンドの使用は同時に再生される可能性があるため避けます。複数サウンドの同時再生は、好ましくないユーザー エクスペリエンスにつながります。

### サウンドの選択

- 心地良いサウンドを選択します。ブンブン鳴る音、衝突音、破壊音などの、不快で不安にさせるようなサウンドは使用しません。
- 短いサウンドを使用します (1 秒未満)。
- 標準的な Windows のサウンドとほぼ同じ音量のサウンドを使用します。特に会議やプレゼンテーションなどの公的な環境で、コンピューターまたはプログラムの起動時に、音量を下げなければならないほどの音が再生されるのをユーザーは嫌がります。Microsoft® Windows® サウンドファイルは、Windows フォルダーの Media フォルダーの中にあります。
- ローカライズを必要とするようなサウンドは避けます。言葉を使用せず、文化に依存する意味や含蓄を含まないサウンドにすれば、ローカライズは必要ありません。

## Windows のシステム サウンド

- 適切なときはいつでも Windows 内蔵のサウンドを使用します。
- システム サウンドは、単にサウンドそのものではなく、関連付けられている意味に基づいて使用するようにします。システム サウンドは一貫して使用される必要があります。

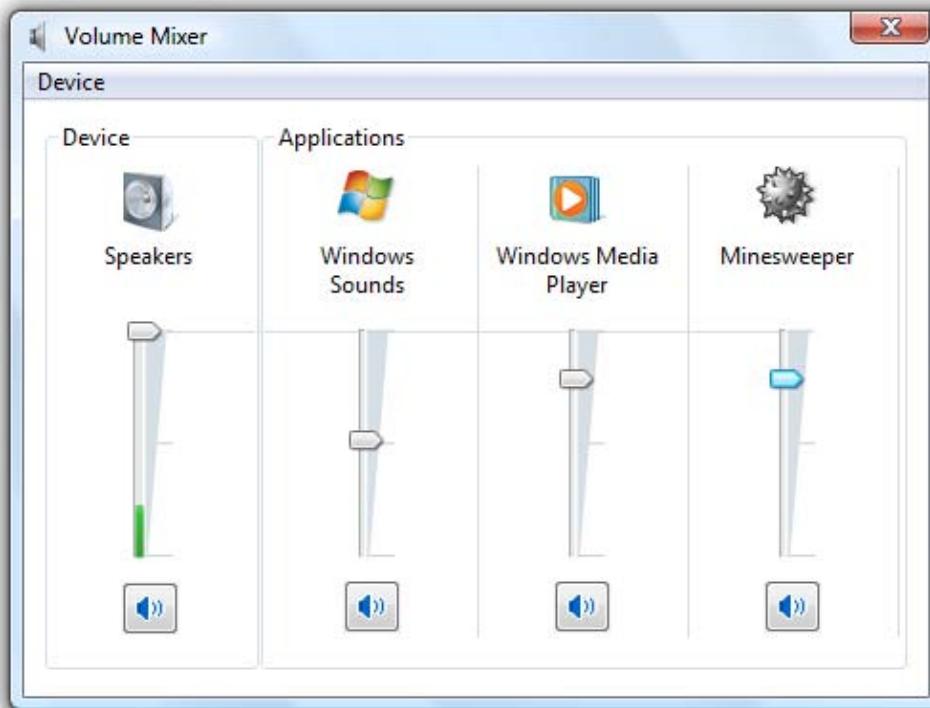
## サウンド デザイン

独自のサウンドを作成するときは、以下の点に留意します。

- 望ましいサウンドの特性を備えたサウンドを作成します。
- 主に中～高周波数のサウンドを作成します (600 Hz ~ 2 kHz)。低い周波数は、より遠くまで伝わり、聞こえにくく、不安にさせる効果を持つため、使用は避けます。
- 標準のサウンドの相対音量は、標準的な Windows のサウンドのレベルに設定します。Windows のサウンドは家庭や職場での環境に適したレベルになっています。作成したサウンドに別のレベルを使用すると、ユーザーは音量を調整しなければならなくなります。
  - 重要なサウンドはやや大きめの音量に設定します。これには、操作の完了、操作の失敗、重要なシステムイベントなどのサウンドが含まれます。
  - 頻繁に再生されるサウンドはやや控えめな音量に設定します。これには、FYI、ブランド化サウンド、サウンド効果などが含まれます。
- Windows のサウンドの意味と矛盾しないサウンドを選択します。Windows のサウンドのカスタム バージョンを作成するには、同じ音程と間隔を保持したまま、オーケストレーションまたは音色を変更します。Windows のサウンドと同等の音程と間隔のサウンドに、別の意味を割り当てるとはしません。
- プログラムに使用するサウンドは、1 つのテーマに沿った関連バリエーションであると感じられるように設計します。プログラムの聴覚的なエクスペリエンスは視覚的なエクスペリエンスと調和させる必要があります。
- サウンドは .wav ファイル形式にする必要があります。16 ビット、44.1 kHz ステレオ非圧縮 PCM (pulse code modulation) 形式が推奨されます。ファイルサイズが重視される場合は、圧縮またはモノラル(モノ)形式を使用します。ただしこの場合、容易に認識できる音質の低下があり、アプリケーションの品質評価に悪影響を与える可能性があることに注意してください。

## ミキシング

- プログラムに、ボリューム コントロールやミュート コントロールは含めません。代わりに、ユーザーが Windows 音量ミキサーを使用して、アプリケーション間の相対的な音量を調整するようにします。プログラムにボリューム コントロールがあると、ユーザーが設定を調整する場所が複数存在することになり、混乱を招くおそれがあります。



Windows 音量ミキサーでは、ユーザーはメインの音量と、現在オーディオを再生しているプログラムご

との音量を調整できます。

- 例外: プログラムの主な目的がオーディオやビデオの再生または作成である場合は、プログラムにボリューム コントロールを含めることが有用な可能性もあります。この場合は、スライダー コントロールを使用し、ユーザーがボリュームを変更するとすぐにフィードバックが返されるようにします。

## Windows との統合

- プログラムのサウンドを Windows のサウンド レジストリに登録します。こうすると、Windows 音量ミキサーでプログラム用のスライダーを追加できるようになります。
- プログラムのカスタム サウンドイベントを登録します。こうすると、Windows のコントロールパネルの [サウンド] にカスタム サウンドイベントが表示されるようになります。各カスタム サウンドイベントには、次のキーを作成します: HKEY\_CURRENT\_USER\AppEvents\Event Labels\EventName = <イベント名>
- プログラムのサウンドイベントに、ハードコードしたサウンドは使用しません。代わりに、サウンドがレジストリエントリを使用して再生されるように指定します。こうすると、ユーザーはコントロールパネルの [サウンド] からサウンドイベントをカスタマイズできるようになります。
  - 例外: ブランド化用のサウンドはハードコードすることもできます。
- プログラムで、ユーザーがサウンドを設定できるようなオプションは提供しません。この目的には、Windows のコントロールパネルの [サウンド] を使用します。
- 頻繁に発生するイベントには、既定でサウンドを割り当てないようにします。気に入らない初期エクスペリエンスを変更する作業をユーザーがしなくて済むようにします。

## DirectSound プログラミングに関する問題

- 独自のボリューム コントロールのある DirectSound プログラムに対しては、プログラムの音量を既定で 100% に設定します。こうすると、オーディオのダイナミックレンジが最大化されます。
- プログラムを排他モードで実行することにより、他のサウンドイベントをロックしないようにします。他のサウンドイベントをロックしてしまうと、他のプログラムの正常な動作が妨げられる可能性があります。たとえば、排他モードを使用すると、コンピューターはテレフォニー デバイスとして使用できなくなります。

## テキスト

- "サウンドアダプター" という用語は使用せず、代わりに "サウンドカード" という用語を使用します。
- "デバイス" は、通常、スピーカー、ヘッドフォン、マイクを指すときに使用します。
- "コントローラー" は、サウンドカードやチップセットなど、デバイスを制御するオーディオ ハードウェアを指すときに使用します。
- "サウンド設定" という用語は、ログオンや新しい電子メールの受信など、一般的なプログラム イベントに対するサウンドのコレクションに対して使用します。"デスクトップ テーマ" という用語は、コンピューターのデスクトップに使用される視覚的要素とサウンドのコレクションに対して使用します。
- "オーディオ" という用語は、音声、音楽、サウンドを広く指すときに使用します。"サウンド" という用語は、もっと狭い意味で、ここで説明しているプログラムおよび Windows のサウンドを指すときに使用します。

## エクスペリエンス

以下の記事では、さまざまなユーザー エクスペリエンスを向上させるためのガイドラインについて説明しています。

- [ソフトウェアのブランド化](#)。普遍的な製品ロゴおよび配色の先を見据えて検討していきます。
- [ファースト エクスペリエンス](#)。ユーザーとプログラムの初めての出会いを効果的、的確にデザインするためのガイドラインです。
- [印刷](#)。現在でも、プログラムのエクスペリエンス全体でユーザーが期待を寄せる重要な要素です。

## ソフトウェアのブランド化

### デザインコンセプト

#### ガイドライン

全般

名前とロゴ

コントロール

スプラッシュスクリーン

サウンド

"ブランド化"された製品は、顧客側に感性的な位置付けを生み出します。製品のブランド化は、さまざまな要因が組み合わさって達成されます。この要因に含まれるものとしては、製品名とロゴ、色、テキスト、グラフィックス、サウンドの使用、その他の各種設計要素のスタイル、マーケティング、そして最も重要なものとして、製品のエクスペリエンス自体の特性が挙げられます。

ブランド化を成功させるには、製品イメージを作り出す一定の技術が必要になります。単に製品のロゴや配色を前面に押し出してアピールするのではなく、ユーザーのエクスペリエンス向上につながるような、有意義で高品位なブランド化を行うことによって、より大きな成果を上げることができます。

注: [アイコン](#)、[フォント](#)、[色](#)、[アニメーション](#)、[サウンド](#)、[ウィンドウ枠](#)に関するガイドラインは、それぞれ別の項目として記載しています。

### デザイン コンセプト

競争の激しい市場では、企業は自社の製品をブランド化することにより、競合他社と差を付けることができます。製品のブランド化は概ね間違っている、避けるべきなどという意見は安直ですが、ソフトウェアのブランド化がうまくいっていない例はあまりに多くあると言っても過言ではありません。ソフトウェアのブランド化の目標は、製品のスタイルと品質、およびそのエクスペリエンスをブランドに関連付けることです。開発者がプログラム自体に注目を集めることでこの目標を達成しようとし、ユーザーを楽しませるのではなくユーザーの気を散らせる結果に終わることがよくあります。

### 成功するソフトウェアのブランド化の共通点

ソフトウェアのブランド化が成功するケースには、次のような共通点があります。

- 明確な、独自のスタイルと個性を確立している
- 共感しやすい
- 品質が高い
- 戦略的な位置付けが行われ、一貫した方法で実施されている
- 全社的なブランド戦略に沿っている
- いつ見ても新鮮で楽しく、飽きない

これに対し、ソフトウェアのブランド化がうまくいかないケースには、次のような共通点があります。

- 明白なテーマやポイントがない
- ユーザーの印象が良くない
- わざわざらしい
- いたるところで目にする
- 独自の外観を持つが、ユーザーにメリットを感じられない
- すぐに飽きがくる

### 製品自体の設計

ソフトウェアのブランド化を成功させるには、まず製品自体の設計が重要です。優れた設計のプログラムには、適切な対象ユーザーを想定して慎重に練り上げられた機能が用意されています。独自の機能と細部までの並外れた心配りが、ブランド化の強力な要因になります。詳細については、「[優れたユーザー エクスペリエンスをデザインする方法](#)」を参照してください。

### 製品名の慎重な選択

強力なブランドには優れた製品名が欠かせません。優れたソフトウェア製品名とは、印象に残りやすく、製品のメリットを端的に示し、多様な市場での差別化を図ることができる製品名です。的確な製品名を選択するには、ブランド化のプロフェッショナルに相談することをお勧めします。長期的に、ブランド化に向けた取り組みの中では、的確な製品名の方がロゴ、配色、コントロールのテーマのような細部よりもはるかに重要になります。

### ブランド化する要素

ソフトウェアのブランド化の要素は、以下のように分類されます。

#### 1 次要素

- 製品名

- ・ 製品ロゴ
- ・ 製品の配色
- ・ 製品固有のサウンド



*Windows Vista® ブランド化の 1 次要素の例*

ブランド化の 1 次要素は、多くの注目を集める傾向にあるため、制限して使用する必要があります。ブランド化の 1 次要素の使用は、いくつかの戦略的なエクスペリエンスに限定します。製品固有のサウンドは、大部分のプログラムにはお勧めしません。

## 2 次要素

- ・ 要素の形状
- ・ アイコンやグラフィックのスタイル
- ・ 2 次的なグラフィック要素
- ・ アクセントの色
- ・ アニメーション
- ・ 切り替え効果
- ・ 影
- ・ 背景と透過



*Windows Vista ブランド化の 2 次要素の例*

ブランド化の 2 次要素は、1 次要素ほど目立たない傾向にあるため、より頻繁に使用できます。ブランド化の 2 次要素には個別にそれほどインパクトを与えないものもありますが、要素が集まることで、製品に特性やスタイルを加えることができます。たとえば静的なグラフィックスはユーザーが慣れてくると無視されるようになりますが、切り替え効果は比較的大きいインパクトを与えることができます。ブランド化の 2 次要素は、1 次要素よりも優先的に使用することをお勧めします。

## 3 次要素

ブランド化の要素のカテゴリはもう 1 つあります。

- ・ カスタム ウィンドウ枠
- ・ カスタム コントロール

ゲームなどの特定の種類のプログラムでは、カスタム コントロールやウィンドウを使用してリアルな独自のエクスペリエンスを作り出すことも有効です。しかし大部分のプログラムでは、標準的な要素のバリエーションを使用するにとどめておく必要があります。プログラムの外観や動作を風変わりなもの

にしても、ブランドの独自性を強化することにはなりません。それよりも、受け入れられやすく、同時に他より目立つ、特性のあるプログラムを作成することを目標にします。

#### ブランド化に適する場所

すべてのものをブランド化する必要はありません。少数のブランド化の要素を戦略的に配置する方が、多くのブランド化の要素をいたるところに無造作に配置するよりも、強力な印象を与えることができます。

ブランド化に向けた取り組みでは、プログラムの特別なエクスペリエンスの部分に重点を置きます。最も心理的なインパクトを与えられるのは、次のような部分です。

- [ファースト エクスペリエンス](#) (特にプログラムの初回使用時)
- メイン ウィンドウまたはホーム ページ
- 重要なタスクの開始時と終了時
- タスクやプログラム領域間での重要な切り替え時
- 時間がかかるタスクでの待機時間
- ログインおよびログオフ時

#### ブランド化に適さない場所

プログラム内のどの要素も "潜在的には" ブランド化に使用できますが、Windows デスクトップは使用しないようにします。これには、[作業領域](#)、[スタートメニュー](#)、[クイック起動バー](#)、[通知領域](#)、[ガジェット](#)も含まれます。

デスクトップはユーザーにとって Windowsへのエントリ ポイントなので、制御の主体はユーザーであるようになります。このようなエントリ ポイントは、プログラムやブランドに対する意識を高める方法としてとらえるのではなく、適切に使用する必要があります。詳細については、「[デスクトップ](#)」を参照してください。

#### ブランド化のプロフェッショナルの協力

ブランド化は専門的なスキルであり、経験のあるプロフェッショナルの協力を得るのが一番良い方法です。わざわざしく思われるようような非効率的なブランド化を広範囲に展開するよりも、最小限のブランド化をユーザーに示す方がはるかに有効です。エンドツーエンドの良質なブランド化エクスペリエンスを作り上げるには、ブランド化およびマーケティングの担当チームと連携します。

#### 5つの重要な点

1. まずは製品自体の設計に注力します。最も強力なブランド化の要因になるのは、ユーザーのニーズを確実に満たす製品設計です。
2. 印象に残りやすく、特徴的で、製品のメリットを端的に示す適切な製品名を選択します。
3. 製品ロゴや配色ではなく、エクスペリエンスや共感性を重視してブランド化を検討します。
4. ブランド化の2次要素を優先します。ブランド化の1次要素の使用は、いくつかの戦略的なエクスペリエンスに限定します。
5. ブランド化のプロフェッショナルの力を借ります。

## ガイドライン

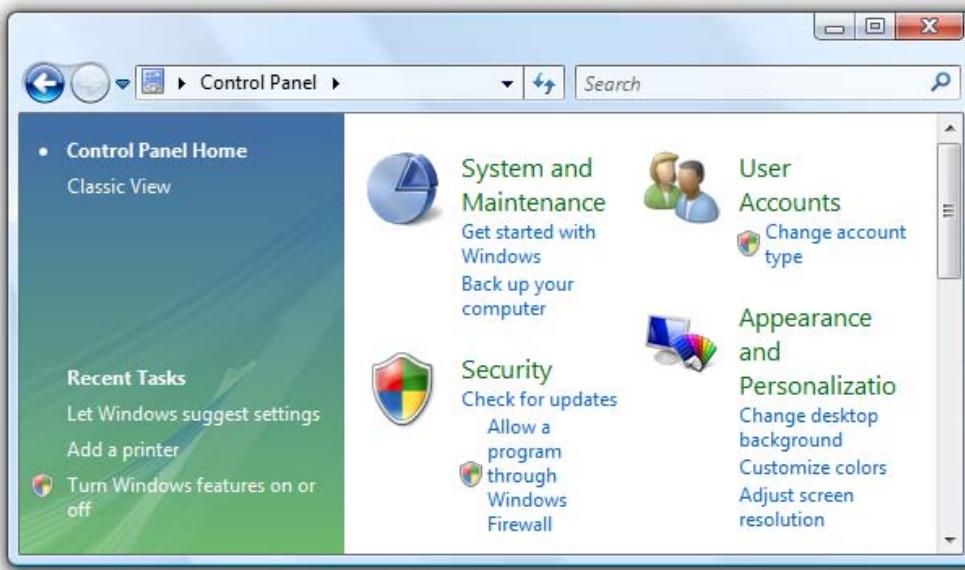
### 全般

- 印象に残りやすく、特徴的で、製品のメリットを端的に示す適切な製品名を選択します。製品名はブランドの基盤となります。
- ブランド化に向けた取り組みでは、プログラムの特別なエクスペリエンスの部分に重点を置きます。たとえば、次のような部分です。
  - [ファースト エクスペリエンス](#) (特にセットアップ中やプログラムの初回使用時)
  - メイン ウィンドウまたはホーム ページ
  - 重要なタスクの開始時と終了時
  - タスクやプログラム領域間での重要な切り替え時
  - ログインおよびログオフ時
- ブランド化の2次要素を優先します。ブランド化の1次要素の使用は、いくつかの戦略的なエクスペリエンスに限定します。たとえば、ロゴではなく、2次要素のグラフィックス、切り替え効果、色などを使用するようにします。ユーザーが多くの時間を費やす場所にブランド化の目立った1次要素を配置しないことも重要です。このような要素があると、ユーザーはわざわざ感じることがあります。

許容される例:



より良い例:



より良い例では、Windows のコントロールパネルアイテム用に、製品ロゴではなくグラフィックの 2 次要素が使用されています。

- ユーザーの気を散らせたり、ユーザビリティやパフォーマンスに悪影響を与えるブランド化は使用しません。
- Windows のデスクトップやスタートメニューはブランド化に使用しません。詳細については、「デスクトップ」および「スタートメニュー」を参照してください。

名前とロゴ

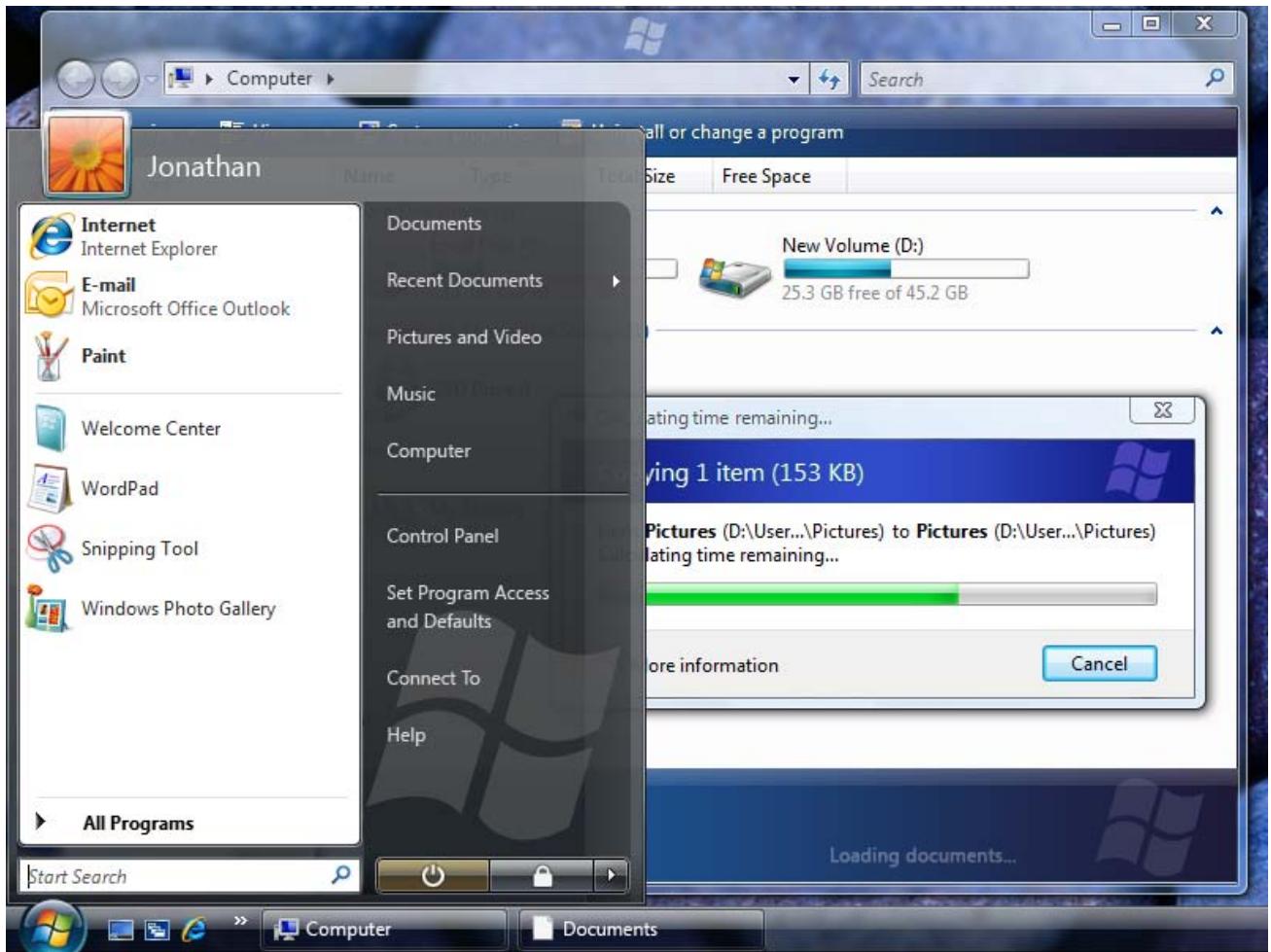
- ユーザーインターフェイスでは、製品や会社のロゴの使用を限定します。UI 画面ごとに会社や製品のロゴを表示することは避けます。
  - 製品や会社のロゴの使用は、メイン ウィンドウまたはホームページと [バージョン情報] ボックスなど、最大 2 か所に限定します。
  - 製品や会社のロゴの使用は、どのような画面においても 2 回までとします。
  - 製品名や会社名の使用は、どのような画面においても 3 回までとします。

間違った例:



この例では、会社名が過度に使用されています。

間違った例:



この例では、個別部分でのロゴの使用は許容されますが、全体の効果としては過剰になっています。

- 製品や会社のロゴは小さいものを使用します。ロゴはユーザーの作業の流れを妨げない場所に配置し、場所に応じたサイズを使用します。
- グラフィック ロゴを使用します。グラフィック ロゴは、フォント、テキスト サイズ、言語パッケージ、テーマなどの違いに影響を受けないため、テキスト ロゴより安定しています。
- アニメーション効果付きのロゴは使用しません。

#### コントロール

- カスタム コントロールはブランド化に使用しません。カスタム コントロールは、特殊な没入型エクスペリエンスを作り出すために必要な場合、または特別な機能を必要とする場合に使用します。

間違った例:



この例では、カスタム コントロールがブランド化に使用されていますが、この使用は不適切です。

#### スプラッシュ スクリーン

- スプラッシュ スクリーンはブランド化に使用しません。スプラッシュ スクリーンは、パフォーマンスの低いプログラムと認識されるおそれがあるため、使用しないようにします。使用するのは、読み込みに通常以上の時間がかかるプログラムで、フィードバックを返したり読み込み時間を短く感じさせることを目的とする場合だけにします。
- アニメーション効果付きのスプラッシュ スクリーンは使用しません。アニメーション効果付きのスプラッシュ スクリーンは、読み込みに時間がかかる原因ととらえられることがよくあり、その仮定が正しいこともあります。

#### サウンド

- 一般的に、ブランド化のみにサウンドを使用することはお勧めしません。どうしてもサウンドをブランド化に使用する場合は、次の点に留意します。
  - サウンドの再生はプログラムの起動時のみとします。さらに、ユーザーがプログラムを起動した場合のみとします。
  - サウンドを視覚的なイベントと同期させます。たとえばプログラム ウィンドウの表示のような UI の切り替え効果と同期させます。

詳細については、「[サウンド](#)」を参照してください。

## ファースト エクスペリエンス

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

ガイドライン

全般

提示方法

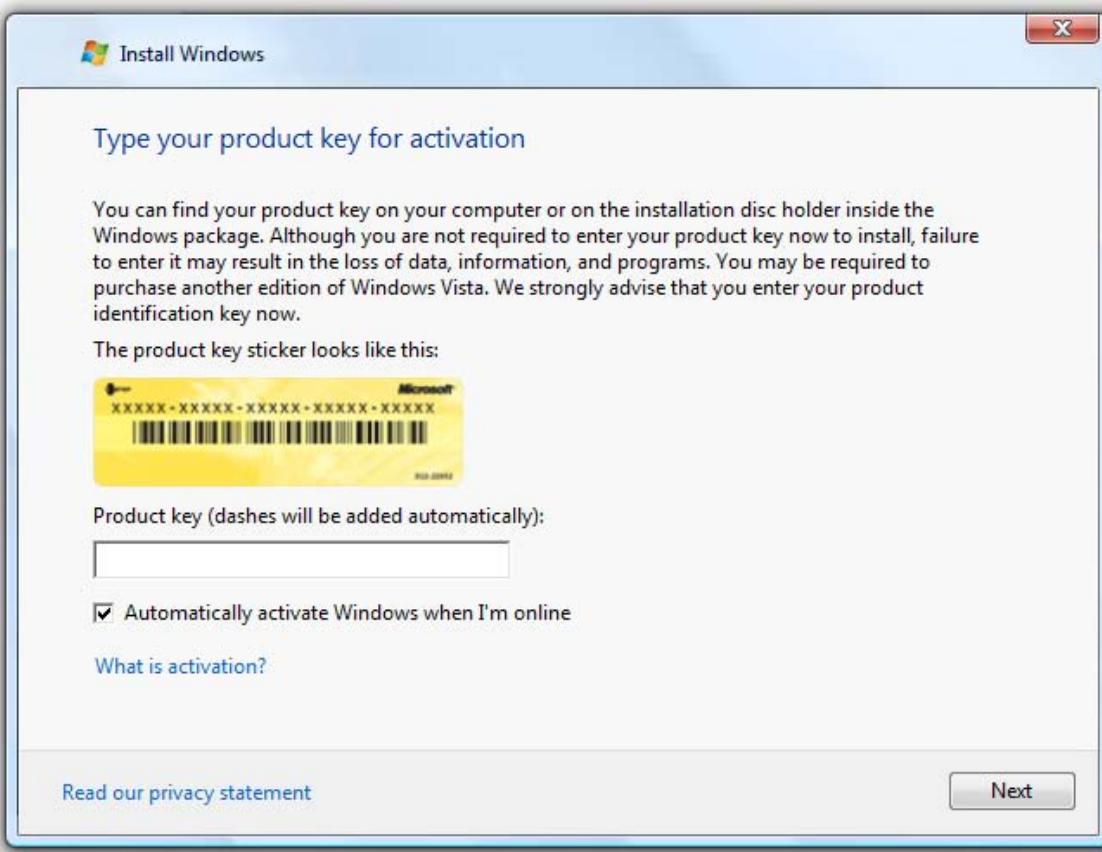
設定

タスク

"ファースト エクスペリエンス" ユーザーインターフェイスを使用すると、ユーザーは新しいプログラムや機能に初めて触れてから日常的に使いこなすまでの移行を簡単に行うことができます。

Windows® プログラムでは、ファースト エクスペリエンスは、ユーザーがセットアップ プログラムを実行するときに初めて実行されます。通常、セットアップ プログラムでは以下の設定が行われます。

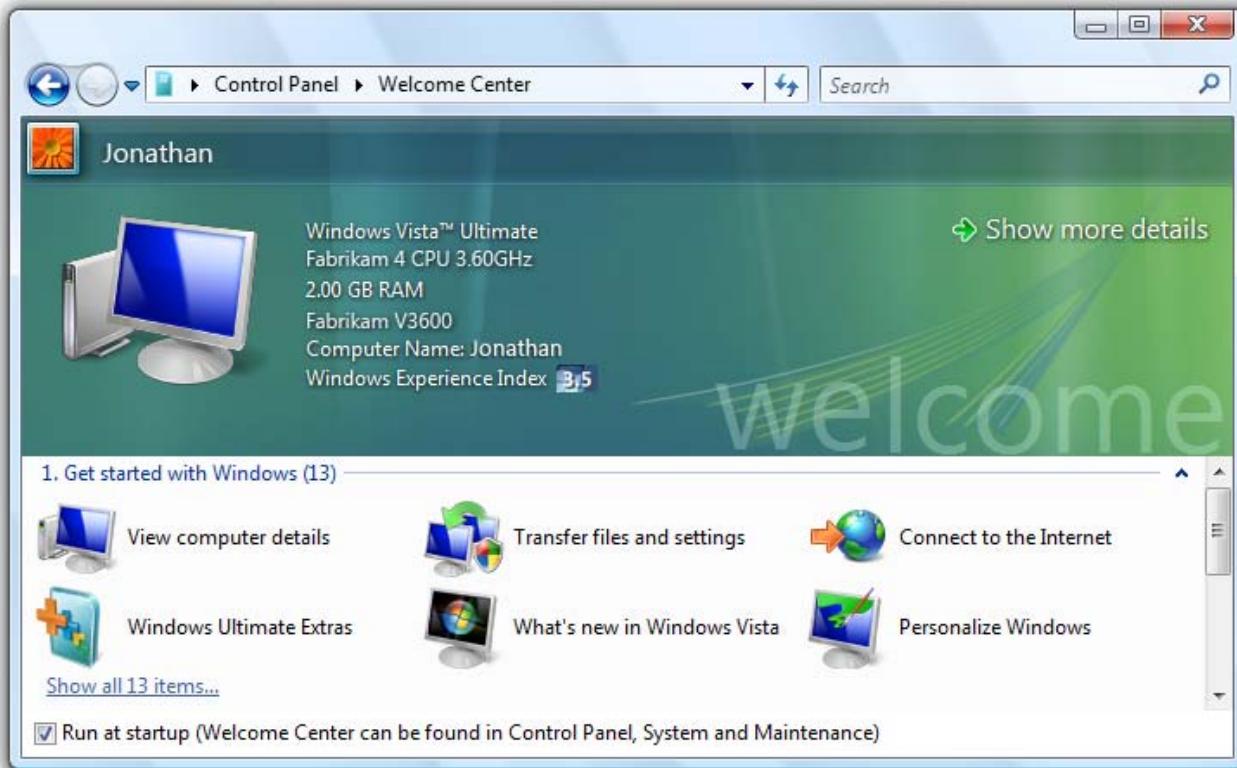
- 使用許諾契約書 (EULA) への同意をユーザーに要求します。
- プロダクトキーの入力を求めます。
- オプションソフトウェアのインストールなど、必要な構成関連のオプションを提示します。
- ユーザーのハードディスクにソフトウェアをコピーします。
- すべてのユーザーに適用されるプログラム オプションを提示します。



一般的な Microsoft® Windows® セットアップ エクスペリエンスの一部

ファースト エクスペリエンスは、それからプログラムや機能の初回使用時まで続きます。この初回使用時のエクスペリエンスでは、以下を行うことができます。

- 現在のユーザーにのみ適用されるプログラム オプションを提示します。
- 製品や機能のチュートリアルを提供します。



## 初回使用時のエクスペリエンス

注: プログラム オプションに関するガイドラインは、別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

#### セットアップ エクスペリエンス

以下の条件に当てはまるかどうかを確認します。

- プログラムを使用するには正しい設定が必要で、設定はすべてのユーザーに適用される。
- その設定によって、コア エクスペリエンスや、プログラムのユーザー情報設定に不可欠なエクスペリエンスがカスタマイズされる。
- 安全な既定がなく、ユーザーは既定以外の設定を選択する可能性が高い。または既定の設定にはユーザーの同意が必要となる。
- ユーザーはセットアップ後、設定を変更する可能性が低い。
- 設定の変更に昇格が必要。

該当する場合は、セットアップ エクスペリエンス中に設定を提示することを検討します。

#### 初回使用時のエクスペリエンス

以下の条件に当てはまるかどうかを確認します。

- プログラムや機能を使用するには正しい設定またはタスクが必要で、設定は個々のユーザーに適用される。
- その設定によって、コア エクスペリエンスや、プログラムのユーザー情報設定に不可欠なエクスペリエンスがカスタマイズされる。
- 安全な既定がなく、ユーザーは既定以外の設定を選択する可能性が高い。または既定の設定にはユーザーの同意が必要となる。
- ユーザーはセットアップ内ではなく、プログラムのコンテキスト内で適切な選択を行う可能性が高い。
- ユーザーはオプションを使用して設定を変更する可能性が低い。

該当する場合は、プログラムや機能の初回使用時のエクスペリエンス中にタスクおよび設定を提示することを検討します。

### デザイン コンセプト

理想的なファースト エクスペリエンスでは、ユーザーが多くの質問に答えたり、たくさんの物事を習得したりしなくとも、プログラムをインストールして(インストールが必要でない場合はプログラムを

起動して)、すぐにプログラムを効率よく使用できます。

この理想はほとんどのプログラムで実現可能なので、できる限り、この理想的なエクスペリエンスに向けて努力します。ただし、大規模なシステム統合が必要なプログラム、多数のオプション機能があるプログラム、またはプライバシーに関連するプログラムでは、この目的は実現できないことがよくあります。たとえば、信頼できない第三者に個人情報が漏れる可能性があるプログラムの機能がある場合、[信頼できるコンピューティング](#)の原則に基づいて、機能を有効にする前にユーザーの同意を得る必要があります。

#### 質問と選択肢との区別

質問には応答が必要です。質問に答えなければユーザーは次に進めません。ファースト エクスペリエンス中の質問は、ユーザーがプログラムを効率よく使えるようになるために乗り越えなければならないハードルです。これに対して、選択肢は任意です。ユーザーは選択肢に対して応答する必要はなく、必要に応じて選択肢を確認することができます。

したがって、セットアップ ウイザードのメイン フローで提示される設定は質問で、セットアップのメイン フロー以外の設定またはプログラムのオプション ダイアログ ボックスの設定は選択肢となります。不必要的質問があるとプログラムのファースト エクスペリエンスは面倒で時間がかかるものになり、ユーザーがプログラムを使い始める前に思い描いていた期待や楽しみが失われます。

#### 適切なファースト エクスペリエンスの使用

必要なタイミングで、ファースト エクスペリエンス中に設定とタスクをユーザーに提示しますが、多くの場合、適切な別の手段があります。

ファースト エクスペリエンス	別の手段
セットアップの質問	適切な既定を選択する。 ユーザーがプログラム オプションから変更できるようにする。 通常のセットアップ方法とカスタム セットアップ方法を提供する。
初回使用時の質問	適切な既定を選択し、ユーザーがプログラム オプションから変更できるようにする。
初回使用時のタスク	代わりに、コンテキストに応じて提示する。
初回使用時の機能の告知	最も一般的で重要なタスクを見つけやすく、コンテキストに即したものにする。
初回使用時のチュートリアル	プログラムの機能をわかりやすくする。
製品の登録	[ヘルプ] メニューおよび[バージョン情報] ボックスにコマンドを表示する。

#### 最も重要な点

ファースト エクスペリエンスをできるだけシンプルに保ちます。プログラムをすぐに機能するようにします。安全性と利便性を兼ね揃えた既定を選択し、必要な場合のみセットアップおよび初回使用時に質問するようにします。

第一印象を良いものにして、その第一印象を長続きさせる機会は一度しかありません。

#### ガイドライン

##### 全般

- ファースト エクスペリエンスは、プログラムや機能の使用に必要なタスクと設定に限定し、他に良い手段がない場合にのみ使用する必要があります。別の手段については、上の表を参照してください。
  - 例外: カスタマイズがコア エクスペリエンスの一部であったり、プログラムのユーザー個人設定に不可欠な場合は、個人設定またはプログラム カスタマイズ設定をファースト エクスペリエンスに追加します。



Windows では、セットアップ時、ユーザーにコンピューター名や背景の選択について質問しますが、これらの設定は製品に対して感情的なつながりを生み出すのに役立ちます。

- ・セットアップエクスペリエンスを使用するのは、タスクと設定がすべてのユーザーに適用され、設定の変更に昇格が必要な場合に限定します。
- ・ファースト エクスペリエンスを使用るのは、タスクと設定が個々のユーザーに適用される場合に限定します。

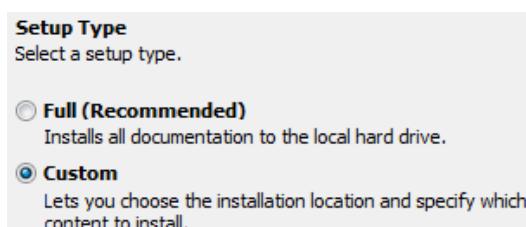
#### 提示方法

- ・必須のタスクと設定よりも任意選択のタスクと設定を優先します。ユーザーにプログラムの構成を強制しないようにします。



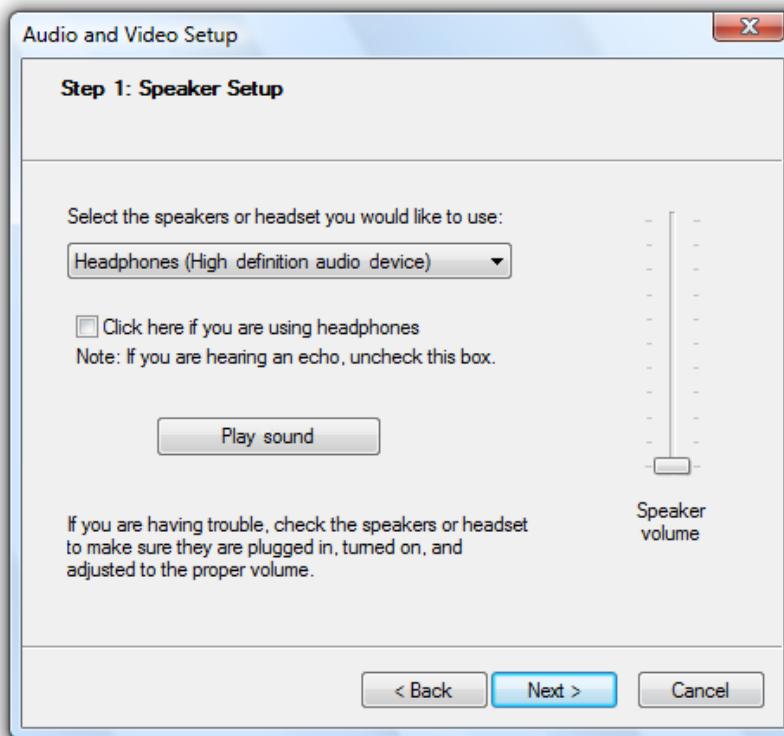
[新しいハードウェアが見つかりました] ダイアログ ボックスでは、ドライバーソフトウェアのインストールを必須のタスクではなく、任意で選択できるようにします。

- ・そうすることが実用的な場合は常に、任意選択のタスクと設定を主要なタスク フローから除外します。たとえば、セットアッププログラムの多くは、カスタム インストール方法を提供し、変更頻度の少ない設定を主要なタスク フローから削除します。



ユーザーがインストールのカスタマイズを行わない場合、セットアップ エクスペリエンスにより主要なタスク フローが容易になります。

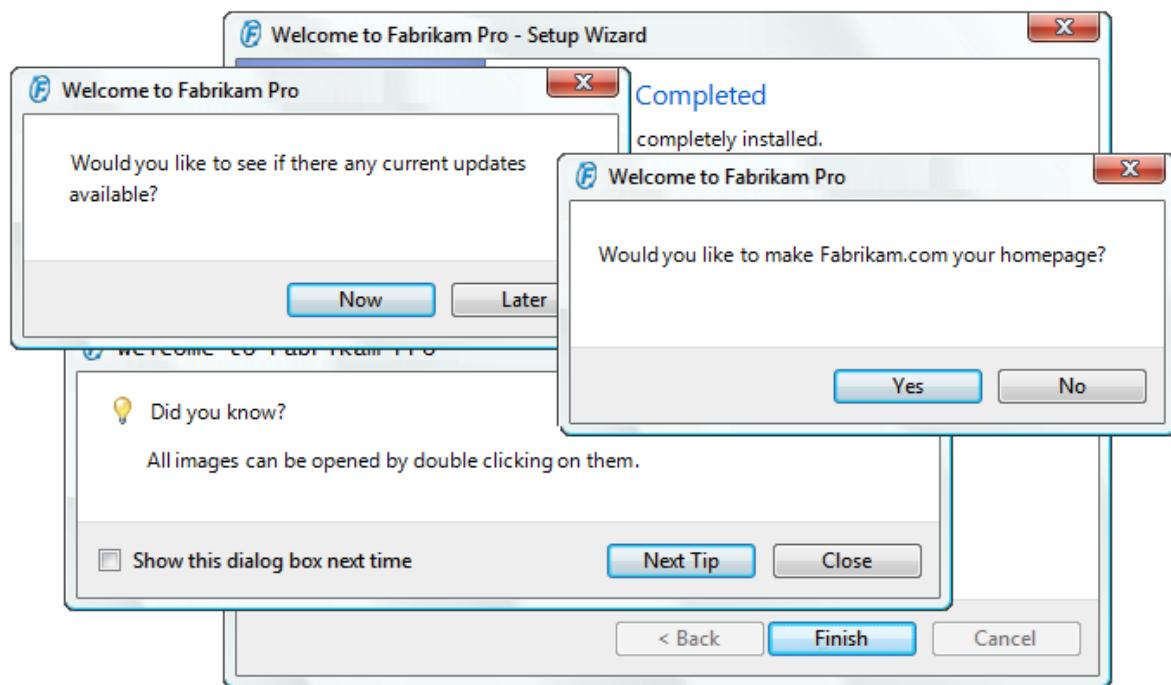
- 過剰なタスクや設定でユーザーを困らせないようにするには、以下の点に留意します。
  - 簡単な設定から始めます。簡単な個人設定から初めて、複雑で技術的なタスクと設定に進みます。たとえば、Windows セットアップでは個人情報から始まり、ネットワーク構成で終了します。
  - コンテキストに応じたファースト エクスペリエンスを使用します。これは、タスクと設定がメイン プログラムに不可欠でない機能にのみ適用される場合に該当します。



*Windows Live Messenger* では、補助的な機能として使用されるオーディオとビデオのセットアップはコンテキストに応じて実行されます。

- 一度にすべてを提示しないようにします。複数の UI 画面の代わりに単一の UI を使用するようにまとめるか、一度ではなく何回かに分割してタスクを表示します。

間違った例:



この例では、初回使用時のエクスペリエンスに過剰なタスクや設定があります。

- テクノロジーの観点からではなく、ユーザーの目的およびタスクの観点から質問やオプションを示します。ユーザーが理解し、明確に区別できるオプションを提供します。ユーザーが情報に基づいた判断を行うことができるよう、十分な情報を提供するようにします。
- 個人情報の必要性が明確でない場合は、プログラムで情報が必要な理由と情報の用途について説明します。

#### Your information

Name:

Email address:

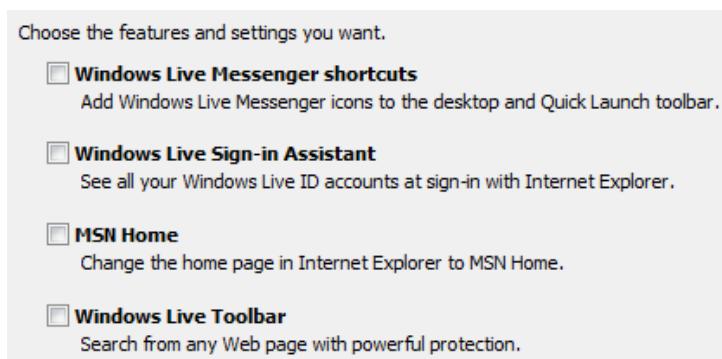
We will use your email address only if there is a problem with your order.

この例では、eコマースアプリケーションで個人情報の用途に関する説明が記載されています。

- ユーザーが他のタスクを効率よく実行できない場合のみ、ファーストエクスペリエンスを全画面表示で提示します。たとえば、Windowsのインストール中はユーザーが他のタスクを実行できないように、Windowsセットアップは全画面表示で提示されます。ファーストエクスペリエンスの大半は、全画面表示にする必要はありません。

#### 設定

- すべての設定に対して、最も安全(データの消失やシステムアクセスが失われるなどを防ぐため)で、最もセキュリティの高い、プライベートな値を既定で選択します。安全性やセキュリティを考慮する必要がない場合は、最もよく使用される値か、最も便利な値を選択します。適切な既定値の選択は、ファーストエクスペリエンスを簡素化する効果的な方法です。
- 以下についてユーザーのオプトインを求めます。
  - 使用許諾契約書(EULA)など法律的な意味合いを持つ設定。このような設定は、既定で選択することができません。
  - Windows自動更新など、自動的なシステム構成の変更を実行する機能。
  - 個人を特定できる情報(PII)またはシステム情報を公開する機能。
  - デスクトップやクイック起動バーへのアイコンの追加など、[スタート]メニューにエントリを追加する以外のユーザーのデスクトップへの変更。
  - 製品拡張機能、サブスクリプション、他社製品などのオプションソフトウェア。



この例では、ユーザーは製品拡張機能、サブスクリプション、他社製品に関するオプトインを選択します。

- 強く推奨される設定に対しては、ラベルに"(推奨)"を追加します。ラジオボタンやチェックボックスの場合は、補足的なメモに対してではなく、コントロールのラベルに追加してください。
- 詳しい知識のあるユーザー向けの設定に対しては、ラベルに"(詳細設定)"を追加します。ラジオボタンやチェックボックスの場合は、補足的なメモに対してではなく、コントロールのラベルに追加してください。

#### タスク

- ユーザーが待機時間を有効に過ごせるようにします。
  - 待機時間が通常、1~2分かかる場合は、ユーザーが待っている間、セットアップ時に新機能について提示するなど、役立つ情報を提供することを検討します。
  - 待機時間が通常、2分よりも長くかかる場合は、ユーザーが他のタスクを実行しやすいようにします。推定待ち時間を探し、その間別の作業を行うように勧め、画面を大幅に変更してタスクの完了を明確に示します。
- ファーストエクスペリエンス中にチュートリアルを提示することを再検討します。ほとんどの場合、ユーザーはプログラムをすぐに使用することを望んでおり、チュートリアルに関心を持つのは後になってからになります。
- ファーストエクスペリエンスでは、機能の告知に関する通知を使用しません。**機能の告知の通知**を使用する代わりに、機能を必要とするコンテキストで見つけやすいように機能をデザインし、ユーザーが自分で機能を見つけるようにします。
- 最初にWindowsを使用するときには通知を使用しないでください。Windows 7では初回使用時のエクスペリエンスを向上させるために、最初の数時間、通知を表示しないようにしています。ユーザーにはこのような通知が表示されないことを踏まえて、プログラムを設計してください。

## 印刷

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

印刷パターン

ガイドライン

全般

ページの書式設定

サイズ超過オブジェクト

ヘッダーとフッター

印刷コマンド

印刷オプション

印刷プレビュー

印刷エラー

テキスト

ドキュメント

この記事では、"印刷"は、出力先が画面表示ではなく用紙となる、用紙上のユーザー エクスペリエンスを指します。"印刷用形式"は、用紙出力により適するように、プログラムで画面表示出力に対して加えることができる変更のことです。

コンピューティングによる"ペーパーレス オフィス"の実現が予想されていましたが、今日においても従来と同様に印刷がなくなることはありません。たとえば、Microsoft® PowerPoint® プrezentation デッキのハード コピーを配布したり、オンラインで見つけた記事を後で慎重にリサーチするために印刷したり、電子形式で受信した重要な電子メールや履歴書を印刷しています。ユーザーインターフェイスをデザインする際、印刷を簡単に見落としてしまいますが、印刷はユーザー エクスペリエンス全体の重要な要素です。

注: [コモン ダイアログ ボックス](#)に関するガイドラインは、別の項目として記載しています。

## 適切なユーザーインターフェイスかどうかの判断基準

プログラムで印刷をサポートする必要がある場合は、以下の点に基づいて判断します。

- 設計の対象となるプログラムの種類は何か。プログラムの種類は、印刷サポートのレベルが適切かどうかを判断する有効な目安となります。ドキュメントやイメージを作成、表示、参照するプログラムでは優れた印刷サポートが必要ですが、その他の種類のプログラムではある程度の印刷サポートで問題ないものもあります(プログラムの種類の一覧については、この記事の「[印刷パターン](#)」を参照してください)。
- 直接、用紙に出力するとメリットが得られるシナリオで使用されるプログラムかどうか。該当する場合は、データを別のプログラムにコピーして印刷するようにユーザーに求めるよりも、プログラムに印刷サポートを追加する方が便利です。

## デザインコンセプト

不要な印刷をなくすプログラムのデザイン

ユーザーが印刷を必要とする理由はさまざまで、適切な理由もあれば、あまり適切でない理由もあります。ユーザーが必要に迫られてではなく、自ら望んで印刷するようにすべきです。ユーザーが印刷する必要に迫られている場合、機能不足が原因の可能性があります。たとえば、以前はコメントを追加したり変更を指定するためにドキュメントを印刷する必要がありました。しかし、現在ではこれらのタスクを、Microsoft Word ドキュメント内で直接行うことができます。印刷に関するプログラムのシナリオを可能な限り幅広く見直して、印刷の必要性がオプションであることを確認し、機能不足が原因で印刷することのないようにします。

また、用紙やインクなどの資源節約は環境保全に役立ち、長期的に組織のコスト削減につながることに留意することは価値があります。

## 画面表示と印刷の違いについて

画面表示出力と印刷には多くの類似点がある一方で、相違点も多くあります。印刷出力には以下の特徴があります。

- 高 dpi 対応。通常、画面表示出力は 96 または 120 dpi (ドット/インチ) で、印刷出力は 600 dpi 以上です。

- 異なるフォントオプション。優れたデザインのフォントは画面表示と印刷の両方に適していますが、テキストが高解像度で大量にある場合は、セリフフォントの方がサンセリフフォントよりも読みやすくなります。したがって、主に印刷用の大量のテキストではセリフフォントを使用し、主に画面表示用のテキストではサンセリフフォントを使用するようにします。詳細については、「[フォント](#)」を参照してください。
- アスペクト比。通常、画面表示では低アスペクト比(4:3または5:4)が、印刷では高アスペクト比(標準ページサイズをベースとして8.5:11または1:1.4142)が使用されます。これは、縦長印刷の方が横長よりも一般的なためです。
- ページ。ページに関して、印刷出力には以下の特徴があります。
  - 標準ページサイズ。米国およびカナダの標準用紙サイズは8.5×11インチで、それ以外の国の標準用紙サイズはA4です。
  - 改ページ。
  - 余白。
  - ヘッダーとフッター。
  - 片面または両面出力。
  - 複数部数への対応(必要な場合)。
  - 順序に関係なく、または選択して印刷(必要な場合)。
- 多くのオプション。ユーザーは、プリンターと用紙サイズ、プリンターのオプション(印刷品質、両面印刷、ホチキス止めなど)、印刷部数、ページ範囲、部単位、印刷形式について選択する場合があります。
- 時間とコストがかかる。大量のドキュメントまたは高品質の写真を印刷するにはかなりの時間がかかり、用紙とインクのコストは時間の経過と共に増加します。これに対し、画面表示出力はすぐに表示され、基本的にコストはかかりません。
- 白黒の可能性。今日のプリンターの多くは白黒ですが、モノクロの画面表示はほとんどありません。
- 対話型ではない。ユーザーはページやコントロールをスクロールして他のコンテンツを表示できません。また、リンクやボタンをクリックしたり、コントロールにマウスポインターを合わせることはできません。対話型のコンテンツは、印刷時にはその真価を発揮できません。
- 用紙、インク、トナー切れやオフラインになることがある。このため、用紙出力ではエラー処理やトラブルシューティングが必要になります。

上記の相違点は印刷デザインに影響する場合があります。優れた印刷エクスペリエンスを作成するには、プログラムの出力をプリンターに転送する以外の機能も持たせる必要があります。

### **WYSIWYG および進化する画面表示要件**

従来から、印刷に関するユーザー エクスペリエンスの最も基本的な原則は WYSIWYG ("what you see is what you get" - 見たとおりに得る) として知られています。この原則は、画面に表示される内容と印刷結果との密接な関連性を示唆しています。WYSIWYG が標準的な基準となる以前は、画面に表示されたドキュメントと印刷したバージョンには関連性がほとんどありませんでした。ユーザーは印刷しなければ、ドキュメントが用紙で実際にどのように見えるかを確認できませんでした。当時のプログラムのほとんどは、ドキュメント作成用および印刷用にデザインされていたので、WYSIWYG を使用することで生産性が飛躍的に向上しました。

今日では、Web サイトは画面表示に最適化することが一般的となり、印刷用形式では見え方がかなり異なる場合があります。さらに、コンピューティング デバイスの多様化(スマートフォンや携帯情報端末など)に伴って、小さい画面表示用に出力を最適化する必要が多々あります。WYSIWYG は、ドキュメント作成プログラムに最適な方法ですが、その他のプログラムではさまざまな対象デバイスに応じて最適化する方が理にかなっています。そのようなプログラムでは、PC ディスプレイと他のデバイスディスプレイとで異なった見え方になる場合があり、印刷したページの見え方とも異なることがあります。

### **印刷の最適化**

厳密な WYSIWYG 印刷エクスペリエンスを導入しないプログラムでも、以下の方法で印刷の最適化を行うことができます。

- 対象ページサイズに合わせてレイアウトを再フォーマットします。
- 印刷プレビューを提供し、可能であれば、ユーザーが直接、印刷ダイアログで試すことができる(余白をドラッグするなど)簡単なカスタマイズオプションを提供します。

- 適切な場合は、印刷用形式のオプションを表示します。
- 複数の部分的なドキュメントを1つのドキュメントにまとめます。
- 背景や、広告などのその他のデザイン要素は削除します(特に白黒プリンターに不適切な場合)。
- ナビゲーションコントロールやコマンドボタンなど対話型要素を削除します。
- スクロールバーを使用したり、マウスでポイントしなくてもすべてのデータが表示されるようにします。

画面表示バージョン:

Business Trip Budget	
Target trip budget	\$ 1,900.00
Airfare	Total cost of tickets \$ 200.00 for 1 ticket(s) \$ 200.00
	* \$ 275.00 for 1 ticket(s) \$ 275.00
	* \$ - for 0 ticket(s) \$ -
Hotel	Cost per night \$ 75.00 for 3 night(s) \$ 225.00
	* \$ 82.00 for 3 night(s) \$ 246.00
	* \$ - for 0 night(s) \$ -
Food	Cost per day \$ 48.00 for 6 day(s) \$ 288.00
Car rental	Cost per day \$ 52.00 for 6 day(s) \$ 312.00
Gas	Cost per gallon \$ 1.74 for 14 gallons \$ 24.36
Entertainment	Amount \$ 130.00 \$ 130.00
Gifts	Amount \$ 85.00 \$ 85.00
Miscellaneous	Amount \$ 55.00 \$ 55.00
	<b>Total cost of the trip \$ 3,840.36</b>
	<b>You're under budget by \$ 59.64</b>

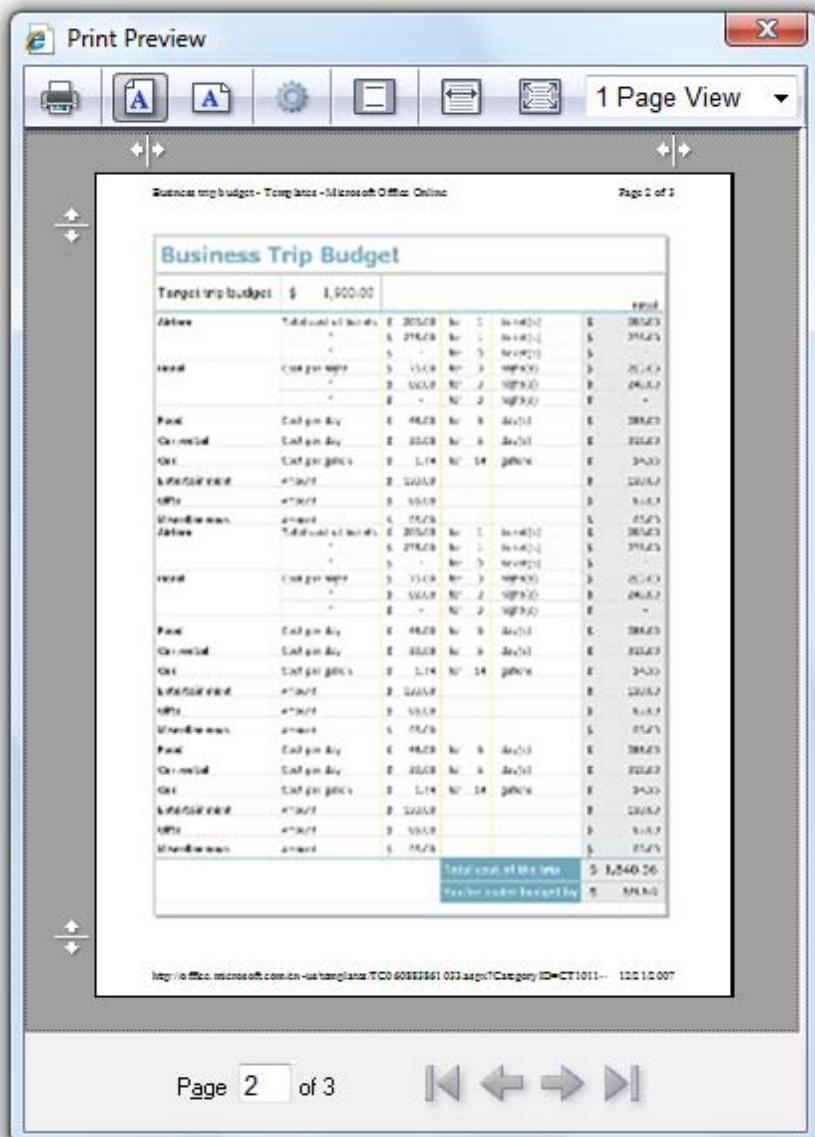
**Browse**

- Agendas
- Award certificates
- Brochures
- Budgets
- Faxes
- Flyers
- Forms
- Gift certificates
- Newsletters
- Plans
- Planners
- Postcards

» More categories

Internet | Protected Mode: Off 100%

印刷用バージョン:



印刷用バージョンでは、すべてのデータが印刷ページに表示され、対話型要素が削除されています。

- リンクをテキストに置き換えます。

許容される例:

詳細については、[UX ガイド](#)を参照してください。

印刷用に最適化される例:

詳細については、[UX ガイド](#) (<http://msdn.microsoft.com/windowsvista/uxguide>) を参照してください。

- 暗い背景と明るい色のテキストを、白い背景と暗い色のテキストに変換します。

適切な印刷オプションの追加

[印刷オプション] コモン ダイアログ ボックスでは、以下のオプションが提供されます。

- プリンターおよび用紙サイズの選択。
- プリンター プロパティの設定。
- ページ範囲、印刷部数、部単位の選択。
- 両面用紙の使用。

プログラムでは、ドキュメント内容のオプション(印刷する内容)、形式のオプション(印刷品質、画像のサイズ、フレームに合わせるなどの印刷方法)、色のオプションなどの追加のオプションが必要な場合があります。追加のオプションを提供する必要がある場合は、[印刷オプション] コモン ダイアログ ボックスを拡張します。カスタム印刷ダイアログ ボックスは作成しません。

印刷オプションをデザインするときは、複数のドキュメントを印刷する際のエクスペリエンスについて検討します。多くの場合、次の印刷ジョブは直前の印刷ジョブと同様です。再印刷および類似の印刷ジョブの既定の設定を最適化します。毎回、ユーザーが完全に最初からやり直す必要がないようにします。

### パフォーマンスおよびユーザビリティ向上のための印刷プレビューのデザイン

不適切な印刷ジョブは時間とコストの無駄になります。ドキュメント作成プログラムでは、ユーザーが実際に印刷を実行する前に印刷結果を評価できるようにする必要があります。印刷プレビューを使用して、以下のことが実行できますようにする必要があります。

- 余白、改ページ、ページの向き、ヘッダーとフッターの評価。
- すべてのページの閲覧。
- 印刷プレビューからの直接印刷。

一部の複雑なドキュメント (CAD (コンピュータ支援設計) 図面など) では、レンダリングに時間がかかることがあります。プレビューのパフォーマンスは重要です。各ページのレンダリングに時間がかかると、印刷プレビューは手間のかかる作業になります。このため、時間がかかる完全に正確なプレビュー表示よりも、ユーザーが印刷結果を評価できる程度に正確で、すばやく表示される印刷プレビューを実装することをお勧めします。

印刷プレビューをデザインするときは、印刷準備のタスク全般について検討します。ユーザーの目的や変更する内容について考慮します。ドキュメント作成プログラムでは、ユーザーがプレビュー内で余白や改行など、頻繁に変更する設定を調整できるように対話型の印刷プレビューを装備する必要があります。

ただし、できる限りプログラムの既定で、適切な設定が行われるようにします。必要に応じて、ユーザーの意図とは異なる印刷状況が起こる可能性がある場合は警告します。印刷プレビューを使用して問題を発見する責任をユーザーに負わせないようにします。たとえば、スプレッドシートに多数の列が含まれ、縦長で 1 ページ内に印刷できない場合があるとします。プログラムで確認のダイアログ ボックスを提示することができますが、解決方法として自動的に横長で印刷する方が適切です。

#### 5 つの重要な点

- プログラムの種類に応じて適切な印刷エクスペリエンスをデザインします。
- 印刷を伴うプログラムのシナリオを見直して、できる限り、印刷が必要かどうかは任意選択とします。
- 印刷コモン ダイアログ ボックスをカスタマイズして有用な印刷拡張機能を提供します。この目的のために、カスタム印刷ダイアログ ボックスを作成しないでください。
- 再印刷および類似の印刷ジョブの印刷オプションを最適化します。
- 適切な場合は常に、プレビュー機能を提供します。

## 印刷パターン

次のように、プログラムの種類によって、適切な印刷エクスペリエンスを判断することができます。

高度なドキュメント作成  
ハイエンドのドキュメントの作成、表示、印刷に使用されます。品質の高い  
印刷イメージを作成できることが、プログラムの主な存在理由の 1 つです。  
上級ユーザーを対象としています。

ユーザーの目的: 完璧な結果。印刷出力を細かく制御できること。

例: Microsoft Word

推奨される印刷エクスペリエンス:

- 印刷用に最適化された出力 (WYSIWYG)。
- 大きなオブジェクトを印刷できるオプション付きの高度なドキュメント書式設定機能。
- ヘッダーとフッターを含む高度な印刷オプション。ドキュメント関連の印刷オプションはドキュメント自体に保存されます。
- 高速、正確、高性能な印刷プレビュー。

## 中級のドキュメント作成

ユーザーの目的: 最小限の労力で適切な結果を生み出すこ

やや複雑なドキュメントの作成および表示に使用されます。品質の良い印刷イメージを作成することは重要ですが、必ずしもプログラムの主な存在理由の一つではありません。中級ユーザーを対象としています。

#### 簡単なドキュメント作成

簡単なドキュメントの作成および表示に使用されます。すべてのユーザーを対象としています。

と。印刷出力をある程度制御できること。

例: Outlook® や Excel® など、Microsoft Office プログラムの大半。

#### 推奨される印刷エクスペリエンス:

- 印刷用に最適化された出力 (WYSIWYG)。
- ある程度のドキュメント書式設定機能に加え、大きなオブジェクトを切り詰めることなく印刷できること。
- ヘッダーとフッターを含むある程度のカスタム印刷オプション。
- 正確で使いやすい印刷プレビュー。

ユーザーの目的: 基本的な印刷サポートと、標準的な印刷オプション。ユーザーは、微調整なしで適切な結果が得られることを想定しています。

例: ワードパッド、ペイント。

#### 推奨される印刷エクスペリエンス:

- 出力は印刷用に最適化される (WYSIWYG) 場合がありますが、必須ではありません。
- ある程度のドキュメント書式設定機能に加え、大きなオブジェクトを切り詰めることなく印刷できること。
- 標準的な印刷オプション。カスタム印刷オプションは任意です。
- シンプル、または印刷プレビューなし。

#### ドキュメントビューアー

ドキュメントの表示に使用されます。ユーザーはドキュメントの内容または書式設定に変更を加えることはできません。

ユーザーの目的: 基本的な印刷サポートと、標準的な印刷オプション。ユーザーは、微調整なしで適切な結果が得られることを想定しています。ユーザーはドキュメントを変更できないので、印刷に関する問題は自動的に処理されます。

例: Windows® Internet Explorer®

#### 推奨される印刷エクスペリエンス:

- 出力は印刷用に最適化される (WYSIWYG) 場合がありますが、必須ではありません。
- プログラムで自動的に改ページの処理、空白ページの削除、大きなオブジェクトの処理、背景、その他のデザイン要素の削除が行われます。
- 標準的な印刷オプション。カスタム印刷オプションは任意です。
- シンプル、または印刷プレビューなし。

#### ユーティリティまたは基幹業務アプリケーション

単純で具体的なタスクの実行に使用されます。すべてのユーザーを対象としています。

ユーザーの目的: 選択したデータを効率的にエクスポートできること。ユーザーは、微調整なしで適切な結果が得られることを想定しています。そのようなプログラムでは、多くの場合、ユーザーは印刷サポートを求めていませんが、サポートがあれば歓迎します。

#### 推奨される印刷エクスペリエンス:

- サポートされるシナリオに応じて、印刷サポートは任意です。
- 出力は印刷用に最適化される (WYSIWYG) 場合がありますが、必須ではありません。
- ある程度のドキュメント書式設定機能。大きなオブジェクトが切り詰められていても問題ない場合があります。

- 標準的な印刷オプション。
- 印刷レビューは任意です。

## ガイドライン

### 全般

- 空白ページまたはヘッダーとフッターだけのページは印刷しません。ただし、ヘッダーとフッターにページ番号が含まれていたり、そのページ番号が他の場所で参照されている場合は空白ページを印刷します。
- プログラムをシャットダウンする前に、保留中の印刷ジョブをスプールから完全に削除します。

### ページの書式設定

- 対象ページサイズ内に収まるようにテキストレイアウトを再フォーマットします。テキストは切り詰めないようにします。
- ユーザーがドキュメントの書式設定を制御しない場合は、次のようにします。
  - ページ上の大いオブジェクトは、自動的に縮小、回転、または分割して処理します。大きいオブジェクトの印刷に関するその他のガイドラインについては、「[サイズ超過オブジェクト](#)」を参照してください。
  - 空白ページおよび空白に近いページをなくすように改ページを最適化します。
  - 暗い背景と明るい色のテキストを、白い背景と暗い色のテキストに変換します。
  - 背景およびその他のデザイン要素を削除します。これは特に、白黒プリンターに不適切な場合に該当します。
- プログラムで異なるドキュメントの一部が複数提示されている場合は、印刷用に1つのドキュメントにまとめる印刷用形式のオプションを表示します。
- 以下のように、対話型要素を削除します。
  - ナビゲーションコントロールやコマンドボタンを削除します。
  - スクロールバーなしの状態で、すべてのデータが表示されるようにします。
  - リンクをテキストに置き換えます。

許容される例:

詳細については、[UXガイド](#)を参照してください。

印刷用に最適化される例:

詳細については、[UXガイド](#) (<http://msdn.microsoft.com/windowsvista/uxguide>) を参照してください。

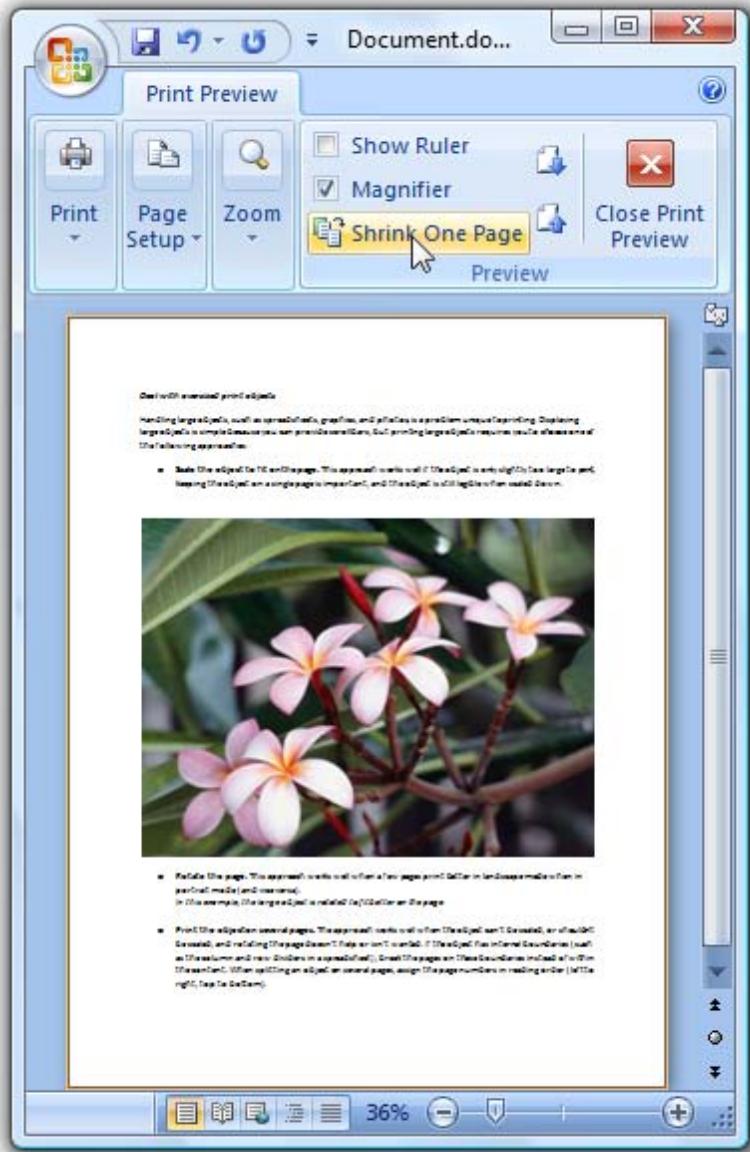
この例では、リンクはかっこで囲んでテキストに置き換えられています。

- マウスポインターをポイントして表示される有益な情報はオンラインに移動します。

### サイズ超過オブジェクト

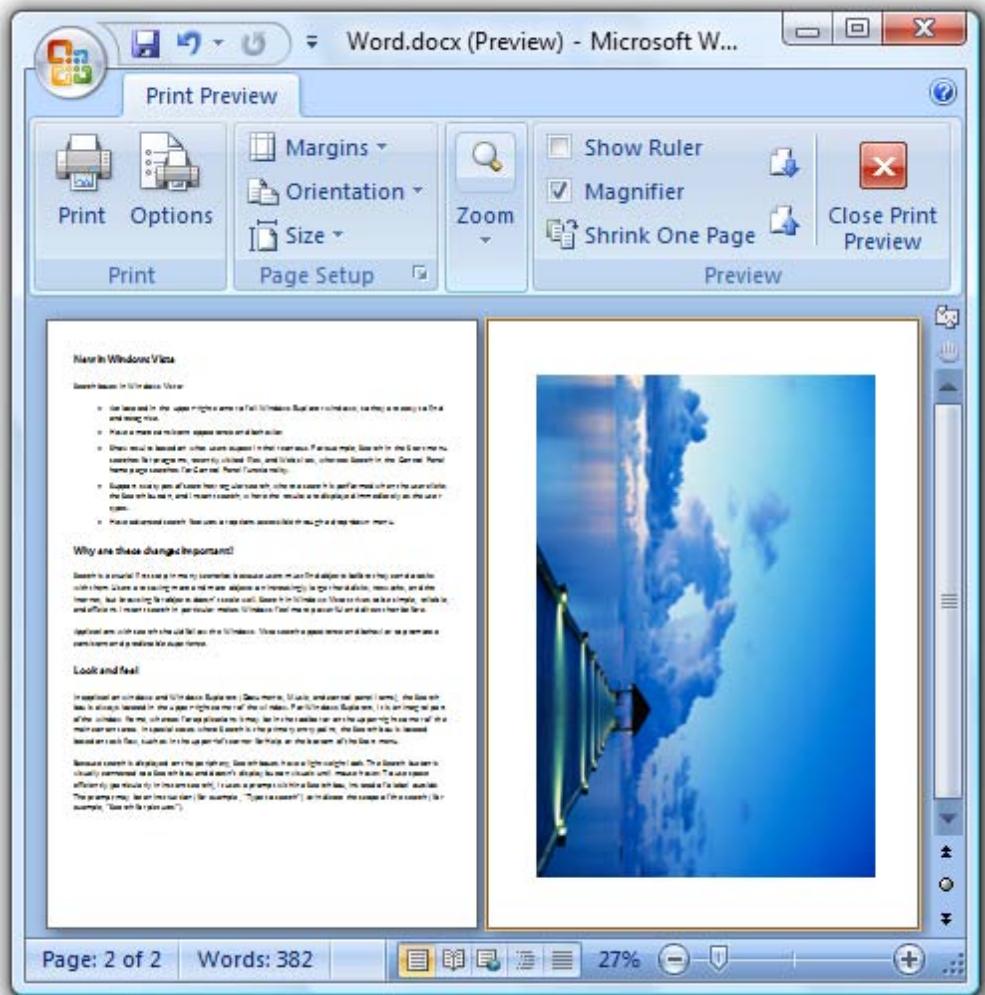
スプレッドシート、グラフィック、写真など大きいオブジェクトの処理は印刷特有の問題です。以下の方法からいずれかを選択します。

- オブジェクトがページ上に収まるように縮小します。この方法は、オブジェクトが少し大きすぎて印刷できない場合、1ページ上にオブジェクトを収めることが重要な場合、オブジェクトを縮小しても見やすさが損なわれない場合に最適です。



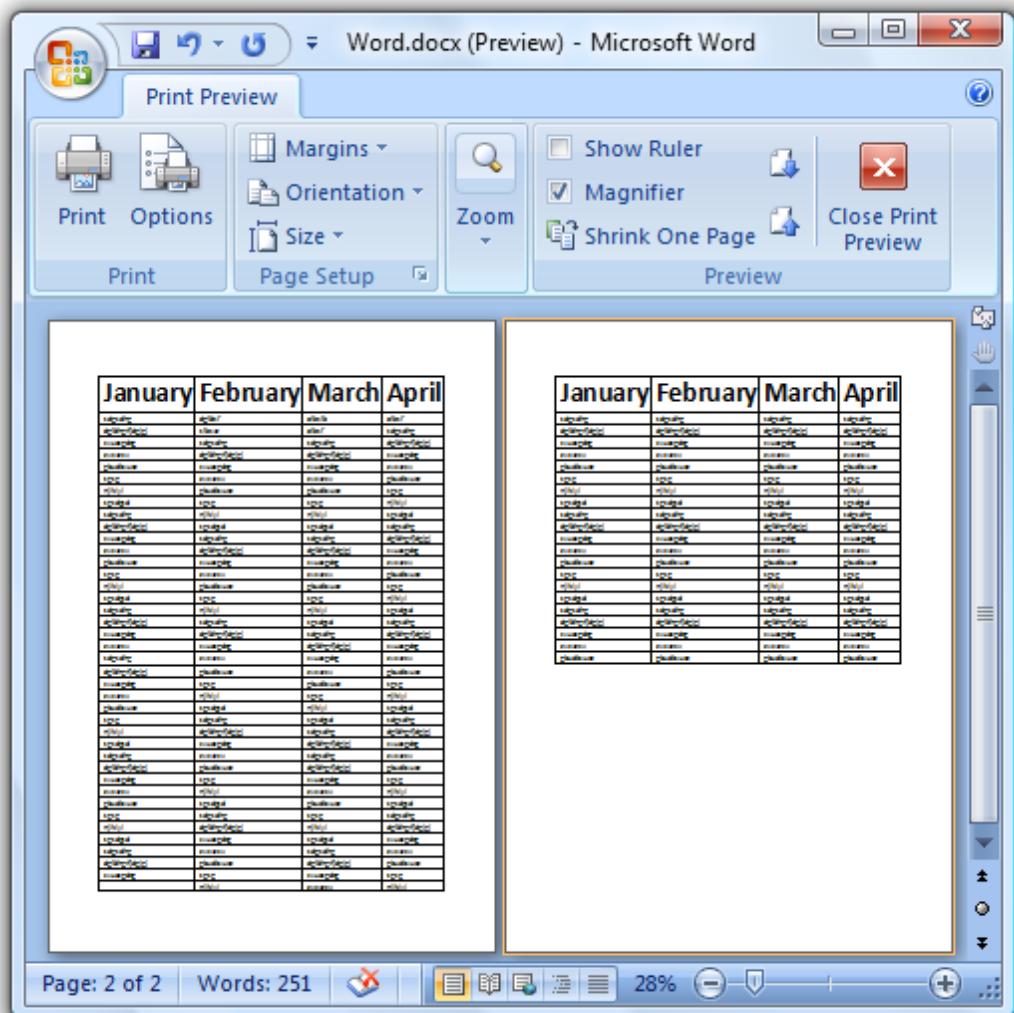
この例では、大きいイメージがページ上に収まるように縮小されています。

- ページを回転します。この方法は、縦長よりも横長で印刷する方が適切なページがある場合に適しています(逆の場合もあります)。



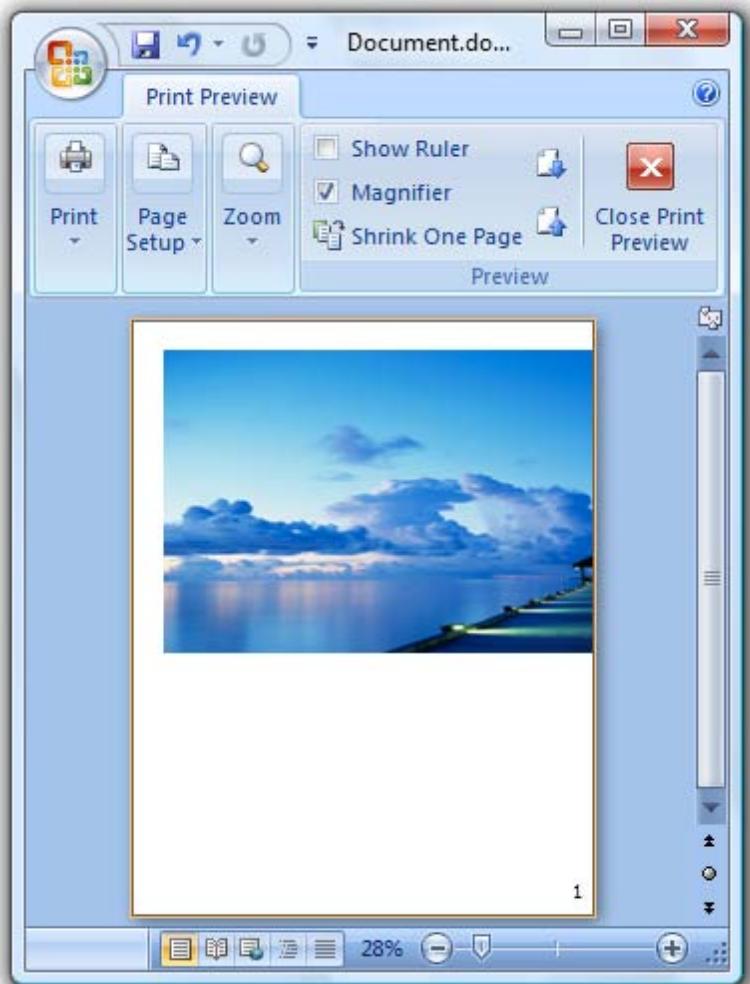
この例では、大きいイメージがページ上にすっきり収まるように回転しています。

- 複数のページにオブジェクトを印刷します。この方法は、オブジェクトを縮小できない場合、ページを回転しても効果的でない場合、縮小や回転を行うと不適切な場合に適しています。オブジェクトに内部境界(スプレッドシート内の列と行の境界線など)がある場合は、コンテンツ内ではなくその境界線で改ページします。また、凡例や列のヘッダーなどページを把握するために必要な要素は繰り返します。複数のページにオブジェクトを分割する場合は、読む順序(左から右、上から下)にページ番号を割り当てます。



この例では、大きな表が 2 ページにわたって印刷されています。列のヘッダーは、すぐにわかりやすいようにページごとに繰り返されています。

- オブジェクトを切り詰めます。これは、切り詰めてオブジェクトの一部だけを印刷しても表示に問題がない場合に該当します。この方法が、一番簡単に実行できる解決方法ですが、受け入れられる可能性は最も低くなります。また、テキストの切り詰めは許容されません。



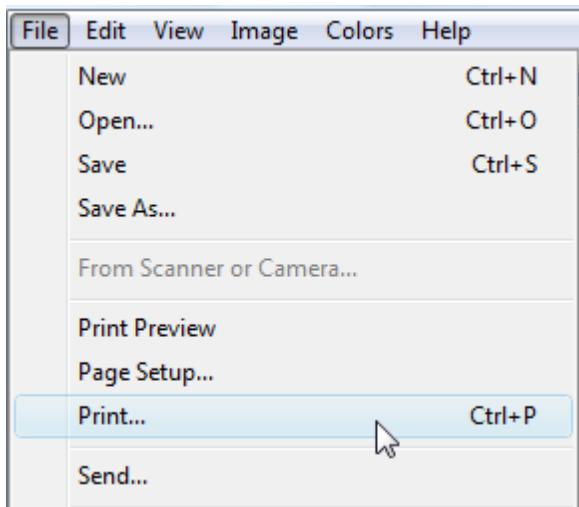
この例では、大きな表が切り詰められています。

#### ヘッダーとフッター

- 上級および中級のドキュメント作成プログラムでは、ヘッダーとフッターを設定します。複数ページのドキュメントを扱うその他の種類のプログラムでは、ヘッダーとフッターを設定することを検討します。
- ヘッダーとフッターはカスタマイズ可能にします。ユーザーが左側、中央部、右側の部分を定義できるようにします。
  - ヘッダーの場合、既定で左側にドキュメント名を配置します。
  - フッターの場合、既定で左側にはドキュメントの著作権または提供元を配置し、右側にはページ番号を配置します。
- わかりやすいファイルパスと URL を使用します。スペースはスペースとして表示し、"%20" とはしません。

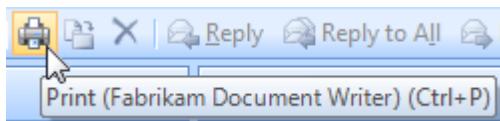
#### 印刷コマンド

- メニューバーとショートカットメニューでは、[印刷オプション] コモンダイアログボックスを表示する [印刷] コマンドを使用します。追加情報が必要であることを示す場合は、省略記号を使用します。



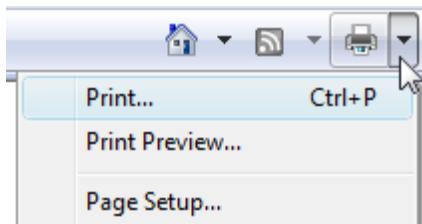
この例では、[印刷] コマンドには省略記号があり、詳細情報を得るために [印刷オプション] コマン ダイアログ ボックスが表示されることを示しています。

- メニュー バーと組み合わせて使用するツール バーの場合は、即時型の印刷コマンドを使用します。ボタンをクリックすると、ドキュメントのコピーが既定のプリンターに 1 部印刷されます。このようなツールバー コマンドは即時型である必要があります。コマンドが直ちに実行されることを示すために、既定のプリンターをツールヒントに表示します。ユーザーはメニュー バーから、全機能を使用できる印刷コマンドにアクセスできます。



この例では、ツールバー上の [印刷] コマンドは、[印刷オプション] コマン ダイアログ ボックスを表示せずにすぐに印刷を実行します。既定のプリンターをツールヒントに表示することで、ダイアログ ボックスを経由しないことをテキストで示しています。

- メニュー バーなしで使用するツール バーの場合は、[印刷] 分割ボタンを使用します。ボタンをクリックすると、ドキュメントのコピーが既定のプリンターに 1 部印刷されます。ボタンの矢印の部分をクリックすると、完全な [印刷]、[印刷プレビュー]、[ページ設定] のコマンドを配置したメニューがドロップダウン表示されます。



この例では、Windows Internet Explorer のツールバーは分割ボタン コントロールを使用して、すべての印刷コマンドを表示しています。

- リボン コマンドのユーザー インターフェイスの場合は、[印刷] コマンドを [アプリケーション メニュー](#) に配置します。

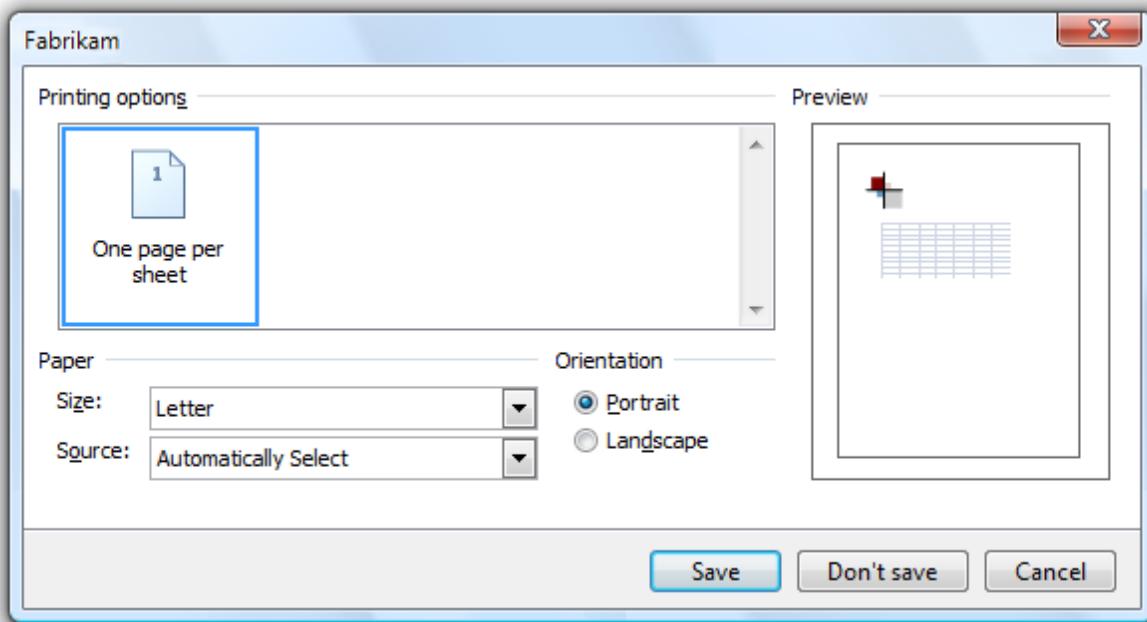


リボンの場合は、アプリケーションメニューを使用して [印刷] コマンドにアクセスします。

#### 印刷オプション

- カスタム [印刷オプション] ダイアログ ボックスは作成しません。追加のオプションを提供する必要がある場合は、[印刷オプション] コモン ダイアログ ボックスを拡張します。追加の印刷オプションに別のダイアログは使用しません。

間違った例:



この例では、*Fabrikam* は誤って追加の印刷オプションに別のダイアログを使用しています。

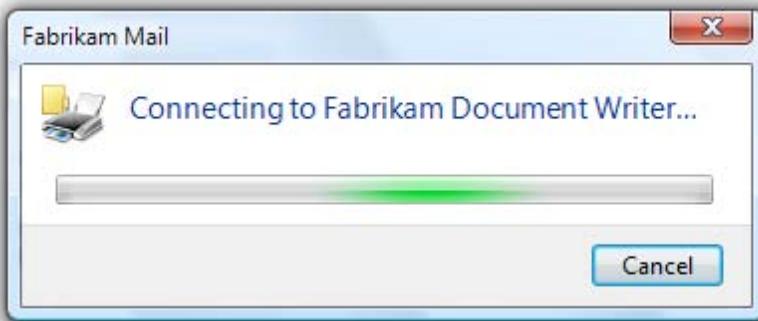
開発者向け情報: [印刷] コモン ダイアログ ボックスの拡張方法の詳細については、[PRINTDLGEX 構造体に関するページ](#)を参照してください。

- [印刷オプション] コモン ダイアログ ボックスを拡張する場合は、既に装備されている機能と重複しないようにします。
- ユーザーがある印刷ジョブから 2 回目以降の印刷ジョブで設定を維持する可能性がある場合は、その設定を既定にします。プログラム起動後の最初の印刷ジョブでは、既定のプリンターを含め標準の既定値を使用します。プログラム インスタンスの 2 回目以降の印刷ジョブでは、最後に選択したプリンターと用紙サイズを保持します。印刷部数やページ範囲は、2 回目以降、再度選択される可能性が大幅に低くなるので、保持しないようにします。
- 現在適用されていないオプションを削除して設定を最適化します。選択したプリンターの機能や現在のド

キュメントの特性に適合しないオプションは削除します。たとえば、写真印刷アプリケーションでは用紙サイズ、用紙の種類、印刷品質の組み合わせを制限して、最適な印刷結果が得られるようにします。このため、用紙オプションで光沢紙を選択すると封筒は用紙形式から除外されることがあります。何らかの理由により、ユーザーがすべてのオプションを確認する場合は、チェックボックスなどのコントロールを使用してこの機能を提供します。

開発者向け情報: 選択されたプリンターの機能を判断する方法については、「[印刷スキーマ](#)」を参照してください。

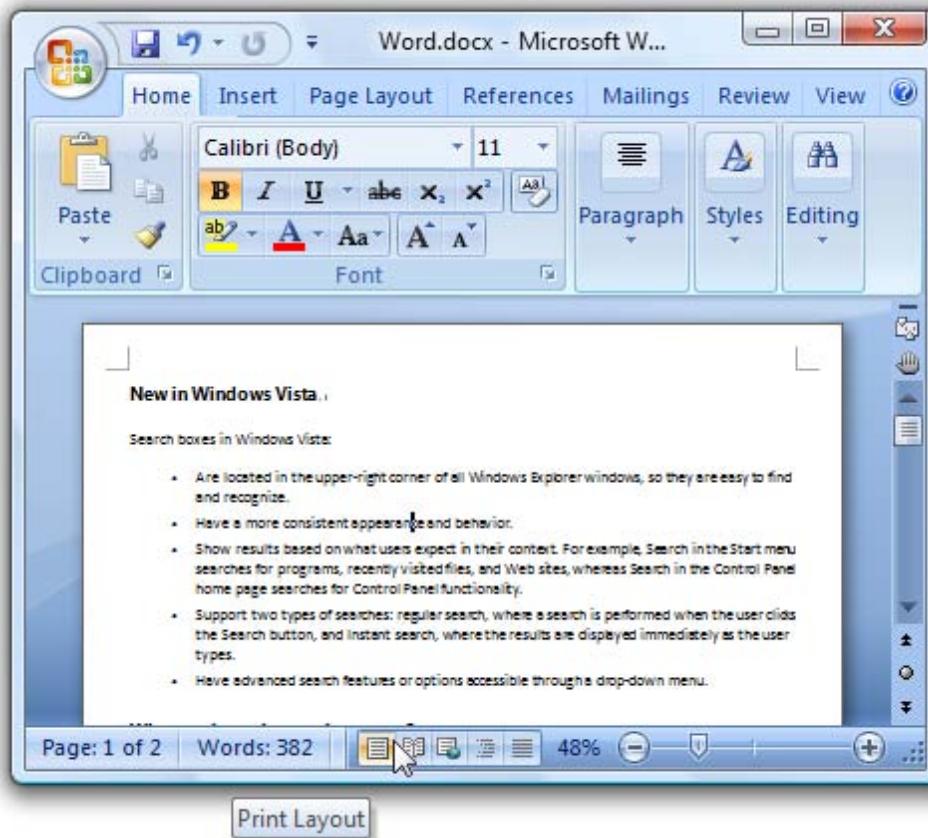
- 高度なドキュメント作成プログラムの場合、ドキュメント関連の印刷オプションはドキュメント自体に保存します。これらのプログラムでは、印刷オプションはドキュメントに不可欠な要素です。
- その他の種類のプログラムの場合は、ユーザーごとに設定を保存します。
- 特殊な印刷には既定以外のプリンターを選択することを検討します。たとえば、写真印刷アプリケーションでは多くの場合、通常使うプリンターに関係なく、このプログラムで最後に使用したプリンターが選択されます。これは、通常使うプリンターが写真プリンターではないことが多いためです。そのようなプログラムでは、最後に選択したプリンターの設定を保存する必要があります。
- プリンター機能の検出時にプログラムをロックアップしません。プログラムのロックアップにより、ユーザー エクスペリエンスの質が低下します。これを避けるには、以下のいずれかの方法があります。
  - 別のスレッドで、プリンター機能の検出を実行する。
  - 10 秒後にタイムアウトする。
  - ユーザーがキャンセルできるようにダイアログ ボックスを表示する。



この例では、ユーザーがタスクに時間がかかりすぎると判断した場合、ダイアログ ボックスでプリンター機能の検出を簡単にキャンセルできます。

## 印刷レビュー

- 適切な場合は常に、印刷レビュー機能を提供します。すべてのドキュメント作成プログラムでは、印刷レビューによるメリットが得られます。ユーザーは簡単なドキュメント作成プログラムに印刷レビュー機能があるとは考えていません。高度なドキュメント作成プログラムでは、メイン プログラム ウィザード内で直接印刷レビューをサポートすることを検討します。

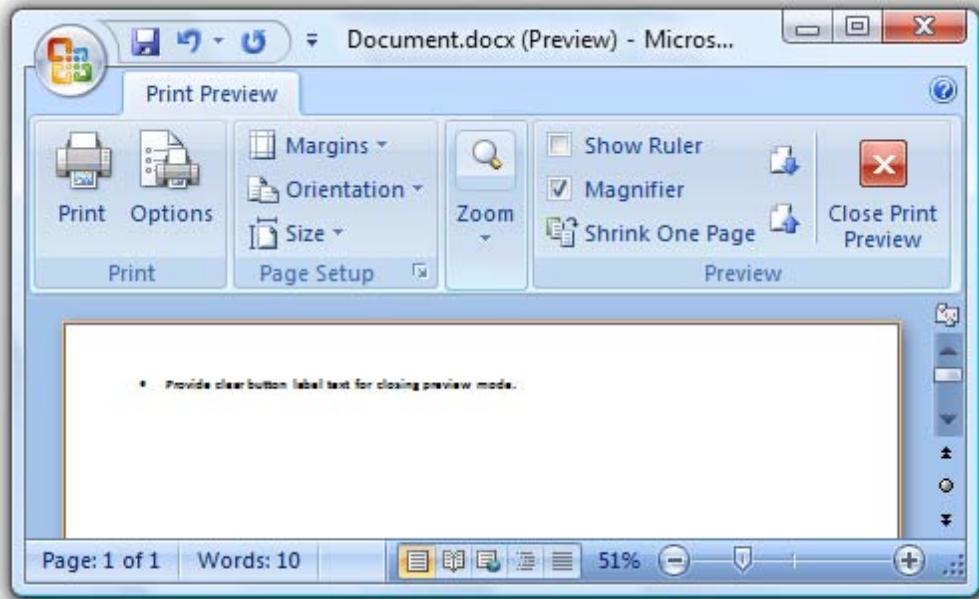


この例の Word では印刷プレビューがメイン プログラム ウィザード内でサポートされています。

- ユーザーが以下のことを実行できる印刷プレビュー機能を提供します。
  - 余白、改ページ、ページの向き、ヘッダーとフッターの評価。
  - すべてのページの閲覧。
  - 印刷プレビューからの直接印刷。

ユーザーがプレビュー内で余白や改行など、頻繁に変更する設定を調整できるよう対話型の印刷プレビューを装備することを検討します。

- 印刷プレビュー ページを 1 秒以内に表示するようにします。表示が遅い完全に正確なプレビューよりも、ユーザーが印刷結果を評価できる程度に正確で、すばやく表示される印刷プレビューを実装することをお勧めします。
- 高度なドキュメント作成プログラムでは、標準の [印刷] ダイアログ ボックス内に直接、プレビュー機能を組み込んで拡張することを検討します。プレビュー機能用に別のダイアログ ボックスは作成しません。
- プレビュー モードを閉じるためのボタンはわかりやすくします。



Word の印刷プレビュー モードには、わかりやすいプレビューを閉じるコマンドがあります。

## 印刷エラー

注: 印刷ジョブがプリンターにスプールされると、それ以降発生するエラーについては Windows で処理されます。プログラムでは、印刷ジョブがスプールされる以前に発生するエラーを処理するだけで済みます。

- 印刷ジョブをスプールする前に、ユーザーが解決できる印刷の問題があるかどうかを確認します。印刷を続行する前に、明確で簡潔な確認を提示します。できる限り、自動的に問題が解決されるようにします。そういうことで、時間とコストの無駄をなくすことができます。

## テキスト

- 用紙の両面を印刷するオプションには、"両面印刷" とオプションのラベルを付けます。"手差し両面印刷"/"手動両面印刷" という用語は使用しません。

## ドキュメント

- "プリントアウトする" ではなく、"印刷する" を動詞として使用します。
- 印刷ジョブの結果を示すために "プリントアウト" を使用しても問題ありません。
- "プリンターキュー" ではなく、"印刷キュー" を使用します。

## Windows 環境

ここでは、Microsoft® Windows® 環境内のさまざまな要素に関するガイドラインについて説明します。

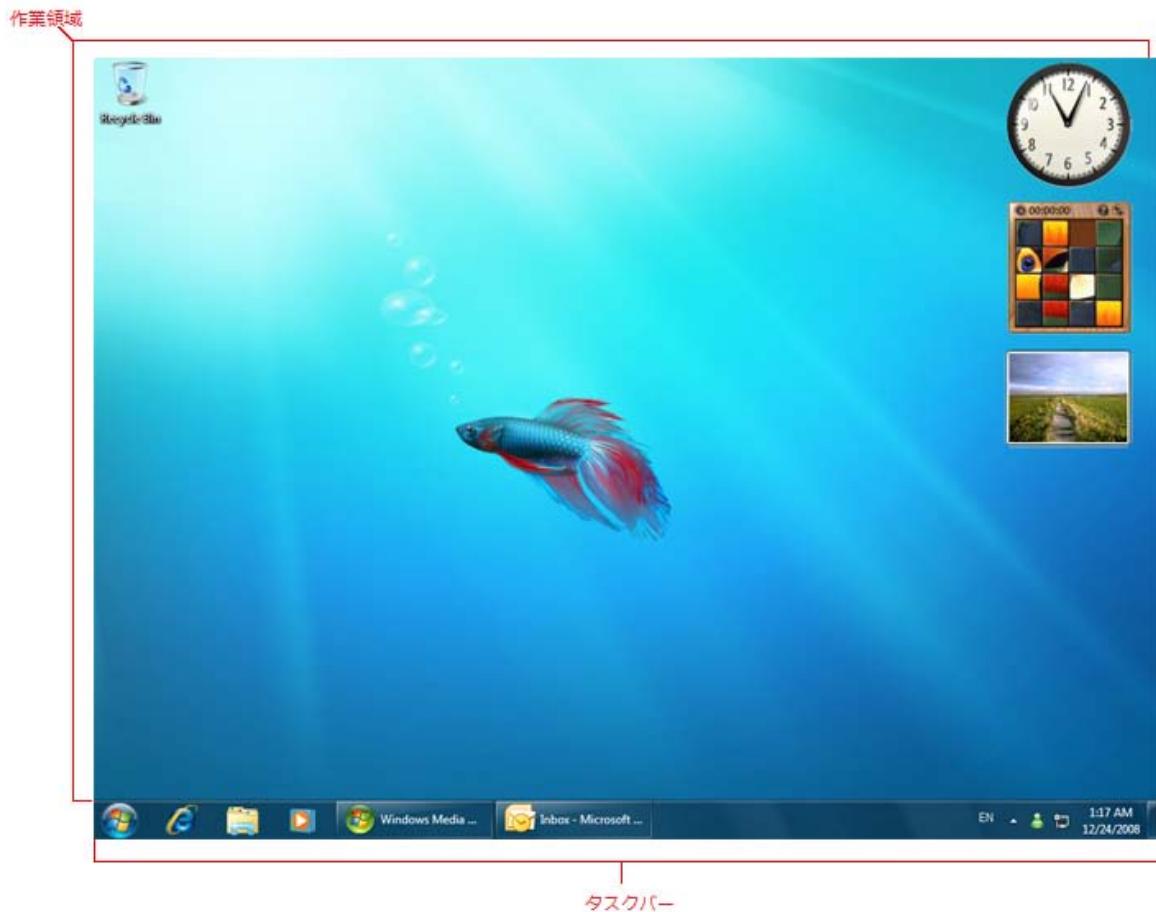
- デスクトップ
- スタートメニュー
- タスクバー
- 通知領域
- Windows デスクトップ ガジェット
- コントロールパネル
- ヘルプ
- ユーザー アカウント制御

## デスクトップ

デスクトップは、ユーザーがプログラムを使用するための作業領域です。プログラムやそのブランドが目立つようにするためのものではないことに注意してください。

[デザインコンセプト](#)  
[ガイドライン](#)  
[ドキュメント](#)

"デスクトップ" は、Microsoft® Windows® によって提供される画面上の作業領域です。物理的な机の上のようなものです。デスクトップは、[作業領域](#) (Windows Vista® の場合はオプションでサイドバーを追加可能) と[タスクバー](#)で構成されます。作業領域は、複数のモニターにまたがることもあります。



### 典型的な Windows デスクトップ

"アクティブな" モニターとは、アクティブ プログラムが実行されているモニターです。"既定の" モニターとは、スタートメニュー、タスクバー、通知領域が表示されているモニターです。

注: [スタートメニュー](#)、[タスクバー](#)、[Windows ガジェット](#)、および[通知領域](#)に関するガイドラインは、それぞれ別の項目として記載しています。

## デザイン コンセプト

Windows デスクトップには、以下のプログラムへのアクセス ポイントがあります。

- ・ 作業領域。ユーザーが作業を行うことができる画面上の領域で、プログラム、ドキュメント、ショートカットを保存することもできます。技術的には、デスクトップにはタスクバーも含まれますが、ほとんどのコンテキストでは単に作業領域のことを指します。
- ・ [スタート] ボタン。すべてのプログラムおよび Windows の特殊な場所(ドキュメント、ピクチャ、ミュージック、ゲーム、コンピューター、コントロールパネル)へのアクセス ポイントです。最近使用したプログラムやドキュメントにすばやくアクセスできる、"最近使った" 一覧も含まれます。
- ・ クイック起動。ユーザーが選択したプログラムへの直接的なアクセス ポイントです。クイック起動は Windows 7 から削除されました。
- ・ タスクバー。デスクトップ プレゼンスを持つ、実行中のプログラムへのアクセス ポイントです。技術的には、タスクバーは [スタート] ボタンから通知領域までのバー全体を指しますが、ほとんどのコンテキストでは "タスクバー" はその間のタスクバー ボタンが表示される領域のことを指します。この領域は "タスクバンド" とも呼ばれます。
- ・ デスクバンド。最小化して機能する、長時間実行されるプログラムです(言語バーなど)。デスクバンドに最小化されるプログラムは、最小化するとタスクバー ボタンは表示されません。デスクバンドは、Windows 7 では推奨されません。
- ・ 通知領域。通知や状態の短期的な情報源です。また、デスクトップに表示されないシステム/プログラム関連の機能へのアクセス ポイントです。



Windows デスクトップのアクセス ポイントには、[スタート] ボタン、タスク バー、デスクバンド、通知領域があります。タスク バー ボタンにはサムネイル機能もあります。

Windows デスクトップは、ユーザーにとって Windows へのエントリ ポイントとなる限られた共有リソースです。制御の主体はユーザーであるようにします。デスクトップ領域は、本来の目的どおりに使用する必要があります。それ以外の使用法は乱用と見なされます。デスクトップを、プログラムやその [ブランド](#) をアピールする手段として使用しないようにします。

#### 最も重要な点

デスクトップを乱用しないでください。制御の主体はユーザーであるようにします。対象ユーザーが頻繁に使用すると見込まれるプログラムについては、セットアップ中にデスクトップ上にショートカットを作成するオプションを提供しますが、このオプションは既定ではオフにします。

#### ガイドライン

- 対象ユーザーが頻繁に使用すると見込まれるプログラムについては、セットアップ中にデスクトップ上にプログラムのショートカットを作成するオプションを提供します。ほとんどのプログラムは、このオプションを提供する必要があるほど使用頻度は高くありません。
- このオプションは、既定ではオフの状態で提示します。ユーザーの多くは、一度デスクトップに作成されたアイコンは、たとえそれが不要なものであってもなかなかそれを削除しようとしないため、このオプションはユーザーに選択を求めることが重要です。不要なアイコンが残ることにより、デスクトップが不必要に乱雑になる可能性があります。
- このオプションが選択された場合、プログラム ショートカットを 1 つだけ作成します。複数のプログラムで構成される製品の場合は、メイン プログラムへのショートカットだけを作成します。
- デスクトップ上に配置するのは、プログラムのショートカットだけです。プログラム本体や、その他の種類のファイルを配置しないでください。

正しい例:



Fabrikam Disk  
Defragmenter

間違った例:



Fabrikam Disk  
Defragmenter

間違った例では、ショートカットではなくプログラムがデスクトップにコピーされています。

- 切り詰められずに表示できるラベルを選択します。ユーザーに対して省略形を表示しないようにします。

正しい例:



Fabrikam Disk  
Defragmenter

間違った例:



Fabrikam Disk  
Defragmenter...

間違った例では、プログラムのショートカットのラベルが長すぎて切り詰められています。

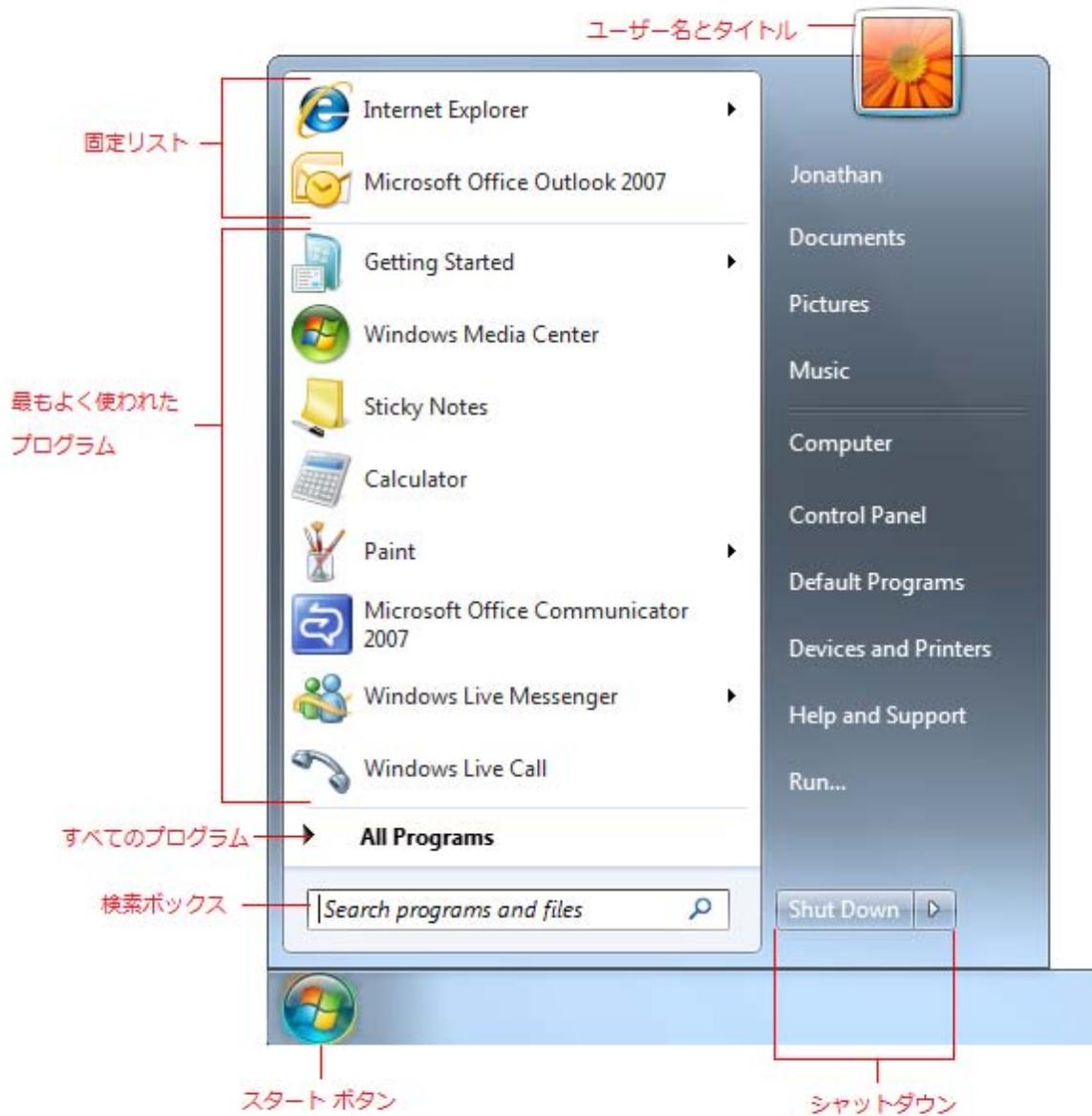
## ドキュメント

- デスクトップを示す場合は、単語の先頭を大文字にしない "desktop" を使用します (英語の場合)。
- デスクトップショートカットを示す場合は、単語の先頭を大文字にしない "shotcut" を使用します (英語の場合)。

## スタート メニュー

デザインコンセプト  
ガイドライン  
プログラム名  
スタートメニューのファイル  
スタートメニューのフォルダー  
スタートメニューの情報ヒント  
ドキュメント

"スタートメニュー" で、ユーザーはプログラム、ファイル(ローカル/共有)、Web ページ、電子メールメッセージを開始します。スタートメニューは、すべてのプログラム、Microsoft® Windows® の特殊な場所(ドキュメント、ピクチャ、ミュージック、ゲーム、コンピューター、コントロールパネル)、およびよく使用するプログラムにすばやくアクセスするための "最もよく使用する" 一覧へのアクセス ポイントです。



### Windows® スタート メニュー

注: メニュー、タスクバー、通知領域に関するガイドラインは、それぞれ別の項目として記載しています。

## デザイン コンセプト

スタートメニューの [すべてのプログラム] の部分は、実質的にメニュー ツリーの役割を果たします。ツリー ビューのガイドラインで説明するように、ツリーを使用することにはジレンマがあります。ツリーは、オブジェクトを整理し、発見しやすくするためのものですが、メニュー ツリー内のオブジェクトを発見することは難しくなります。

スタートメニューの、この見つけやすさに関する問題の解決方法には以下のようないことがあります。

- 主要なプログラムの実行可能ファイルへのショートカットを配置して、スタートメニューのアイテム数を減らします。他のプログラム ファイルには、プログラム本体、該当するコントロールパネルアイテム、またはセットアップ プログラムからアクセスします。
- プログラムを最上位のフォルダーまたは 1 つの製品フォルダーに配置して、不要なフォルダーを排除します。一般的に、スタートメニューに配置するプログラムのショートカットは 1 つにする必要があります。
- 参照しやすいように、内容がわかりやすいプログラム名を選択します。
- ユーザーが探す可能性の高い個々の単語が含まれているプログラム名、およびスタートメニューの情報ヒントを選択します。

## ガイドライン

### プログラム名

プログラム名を選択するにあたっては、最も重要なプログラムのイメージ、認知度、ブランド化など、さまざまな判断要素があります。プログラムの見つけやすさに影響するスタートメニューのガイドラインを以下に示します。特に、あまり知られていないプログラムや頻繁に使用されないプログラムに対して、この見つけやすさは重要です。あまり知られていないプログラムや頻繁に使用されないプログラムでは自由度の高い命名ができます。

- 参照しやすいプログラム名を選択するには、以下の点に留意します。
  - ユーザーがプログラムの名前だけでプログラムの主な目的を理解できるようにわかりやすい名前を使用します。
  - 適切なアルファベット表記のプログラム名を使用します(英語の場合)。名前は、スペース、数字、または記号ではなくアルファベット文字で開始します。
  - ユーザーが一般的にプログラムをバージョン番号で参照する場合を除き、バージョン番号をプログラム名に追加しないようにします。
- 探しやすいプログラム名を選択するには、以下の点に留意します。
  - 記憶しやすい独特の名前、または対象ユーザーが探す可能性の高い語が含まれている名前を使用します。
  - 複合語よりも個々の単語を優先的に使用します。
  - スペルミスしやすい名前やスペルが間違っている名前は避けます。
  - 専門用語および造語は避けます。
- タイトルスタイルの大文字化を使用します。

### 正しい例:



### 間違った例:



間違った例では、複合語、バージョン番号、または、"ディスク"ではなく "ボリューム"という用語が使用されています。"ボリューム"は技術的には適切ですが、ボリュームと聞くと、ほとんどのユーザーはディスク ドライブではなく音量を思い浮かべます。造語の名前を使用すると、あまりよく使用しないプログラムを識別しにくくなります。

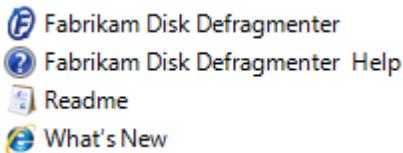
## スタート メニューのファイル

- スタート メニューには、プログラム ショートカットのみを配置します。以下の項目へのショートカットをスタート メニューに配置しないようにします。
  - プログラムのアンインストーラー。ユーザーは、プログラムのコントロール パネル アイテムを使用して、アンインストーラーにアクセスします。
  - ヘルプ ファイル。ユーザーは、プログラムから直接ヘルプ トピックにアクセスします。
  - コントロール パネル アイテム。ユーザーは、コントロール パネル のホーム ページからコントロール パネル アイテムにアクセスします。
  - プログラム オプション。ユーザーは、[オプション] コマンドからプログラム オプションにアクセスします。
  - ユーティリティ プログラム。ユーザーは、[ツール] メニューのコマンドからユーティリティにアクセスします。
  - Readme ファイル。ほとんどのユーザーは Readme ファイルを確認する事がないので、そのようなファイルが本当に必要かどうかを再検討します。どうしても Readme ファイルが必要な場合は、セットアップ プログラムから Readme ファイルにアクセスできるようにします。
  - Web サイト。ユーザーは、プログラム内の該当するリンクまたは技術サポート サイトのヘルプを使用して、Web サイトにアクセスします。

正しい例:



間違った例:



間違った例では、より適切なアクセス ポイントがある項目に対する不要なショートカットが表示されています。

- スタート メニューで使用するショートカットはプログラムごとに 1 つだけにします。最上位 フォルダー と アクセサリ フォルダー というように、異なる場所に複数のショートカットを配置しないようにします。プログラム 内の特定のタスクにアクセスするショートカットは配置しません。

正しい例:



間違った例:



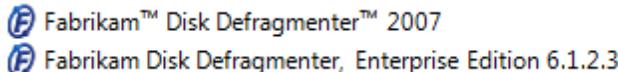
間違った例では、メイン プログラム 内の特定のタスクにアクセスするショートカットがあります。

- プログラムの名前を使用してプログラム ショートカットのラベルを付けます。その他のラベルを使用したり、ラベルに商標記号などの追加情報を含めないようにします。ユーザーが製品から会社名を連想できない場合を除いて、会社名は含めません。ユーザーが一般的にプログラムをバージョン番号で参照する場合を除き、バージョン番号をプログラム ショートカットに追加しないようにします。

正しい例:



間違った例:



間違った例では、不必要的情報やバージョン番号、商標記号が含まれています。

- ・セットアップ中、スタートメニューにプログラムショートカットを配置するオプションを提供しないようにします。プログラムショートカットの作成は自動的に実行されるようにします。
- ・セットアップ中、スタートメニューの上部にプログラムショートカットを固定するオプションは提供しないようにします。プログラムショートカットを手動で固定するかどうかはユーザーの選択に任せます。Windows Vista®以降では、プログラムは自動的に固定されなくなりました。
- ・**タイトルスタイルの大文字化**を使用します。

## スタートメニューのフォルダー

- ・プログラムショートカットを[すべてのプログラム]の最上位に配置します。Windows 7およびWindows Vistaでは、スタートメニューの拡張性が強化され、プログラムを最上位で見つけやすくなっています。次の場合は例外です。
  - ・コントロールパネルアイテムにアクセスするには、[コントロールパネル]を使用します。[すべてのプログラム]には、コントロールパネルアイテムへのショートカットを配置しません。また、トラブルシューティングプログラムにアクセスするときも[コントロールパネル]を使用します。
  - ・対象ユーザーがプログラムを付属的なものとして考えており、プログラムがユーザー エクスペリエンスの中核を成すものではない場合は、[アクセサリ]フォルダーを使用します。たとえば、Windows Media Playerはユーザー エクスペリエンスの中核を成すものなので[すべてのプログラム]に配置されます。サウンドレコーダーは中核を成すものではないので[アクセサリ]に配置されます。
  - ・プログラムがシステムメンテナンスユーティリティである場合のみ、[システムツール]フォルダーを使用します。システムメンテナンスユーティリティは、[システムツール]と[コントロールパネル]の両方に表示できます。
  - ・ITプロフェッショナル向けのプログラムには、[管理ツール]フォルダーを使用します。
  - ・[メンテナンス]フォルダーは使用しません。このフォルダーは、Windows Vistaのバックアップと復元センター、ヘルプとサポート、問題のレポートと解決策、Windowsリモートアシスタンスに使用されます。
- ・製品に3つ以上の個別のプログラムが含まれる場合は、製品フォルダーを作成します。これは、ユーザーがコレクション単位で製品を捉えている場合にも該当します。

間違った例:



正しい例:



間違った例では、個々のプログラムが2つしかないので、フォルダーに配置する必要はありません。

- ・製品フォルダーの階層レベルは1つだけ使用します。
  - ・例外: 製品が複数のプログラム(6つ以上)のコレクションで、そのうち2つ以上のプログラムが補助的なユーティリティと見なされている場合は、フォルダーの階層レベルを2つ使用します。

間違った例:



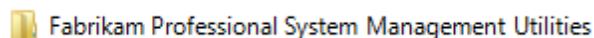
正しい例:



間違った例では、不必要的フォルダーがあります。

- 説明的でありながらも簡潔なフォルダ名を選択するには、以下の点に留意します。
  - 3語以内で表記します。

間違った例:



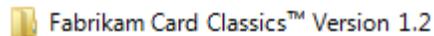
正しい例:



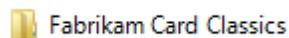
間違った例では、フォルダ名が長すぎます。

- 商標記号は含めないようにします。ユーザーが一般的にプログラムをバージョン番号で参照する場合を除き、バージョン番号をフォルダ名に追加しないようにします。代わりに、バージョン番号をスタートメニューの情報ヒントに追加します。

間違った例:



正しい例:



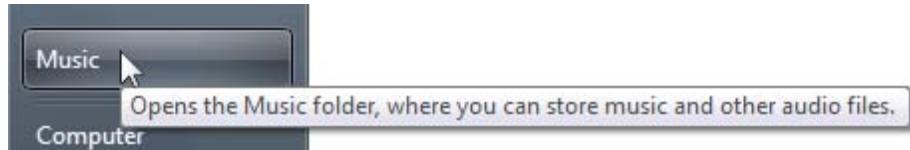
間違った例では、不必要的バージョン番号と商標記号が含まれています。

- タイトルスタイルの大文字化を使用します。
- フォルダ名に"フォルダー"という用語は使用しません。

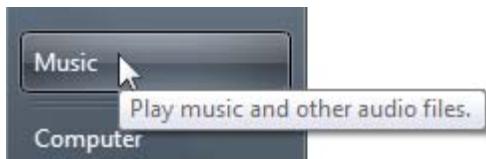
## スタートメニューの情報ヒント

- スタートメニューの情報ヒントでは、アイテムを簡潔に説明し、ユーザーがアイテムに対して実行できる主要なタスクを示します。
- 有益な情報を提供します。ユーザーができることに焦点を当て、アイテムの名前を繰り返したり、説明の中でアイテムの名前を使用することは避けます。
- 具体的にします。一般的な動詞や、"その他のタスク"および"その他の種類のドキュメント"など、何にでも当てはまる語句は使用しません。重要な情報は、具体的に一覧にします。一覧にしない場合は、情報ヒントの一覧が完全な一覧でないことをユーザーが理解している必要があります。
- 簡潔にします。25語あるいは75文字以内で表記します。これより長い情報ヒントは、ユーザーに読む気を失わせます。
- 現在時制の動詞の命令形で始めます(英語の場合)。"create(作成)"、"edit(編集)"、"show(表示)"、"send(送信)"などを使用します。"管理する"、"開く"などの一般的な動詞よりも、具体的な動詞を優先して使用します。

間違った例:



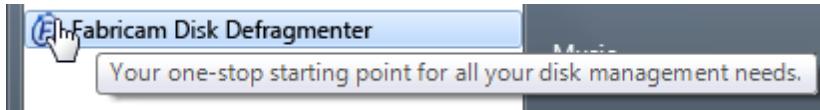
正しい例:



間違った例では、情報ヒントに一般的な動詞が使用されています。

- 要点だけを伝えます。"開始する"、"～できる"、"～するために使用する"、"提供する"など、スタートメニューのすべてのアイテムに適用できる動詞は使用しません。
- 宣伝文句のような表現は使用しません。

間違った例:



この例では、情報ヒントが宣伝文句のようになっています。

- センテンススタイルの大文字化を使用します。
- 開発者向け情報: スタートメニューの情報ヒントのテキストは、アイテムの Comment フィールドから表示されます。

## ドキュメント

スタートメニューを示す場合は、"Start (スタート)" は大文字化しますが、"menu (メニュー)" は大文字化しません(英語の場合)。

## タスクバー

タスクバーは、デスクトップに表示されるプログラムのアクセス ポイントです。新しい Windows® 7 タスクバーの機能を使用すると、ユーザーはタスクバーから直接、コマンドを実行したり、リソースにアクセスしたり、プログラムの状態を確認することができます。

### 適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

ガイドライン

タスクバー ボタン

アイコン

オーバーレイ アイコン

タスクバー ボタンの点滅

クリック起動のショートカット

ジャンプリスト

サムネイルツールバー

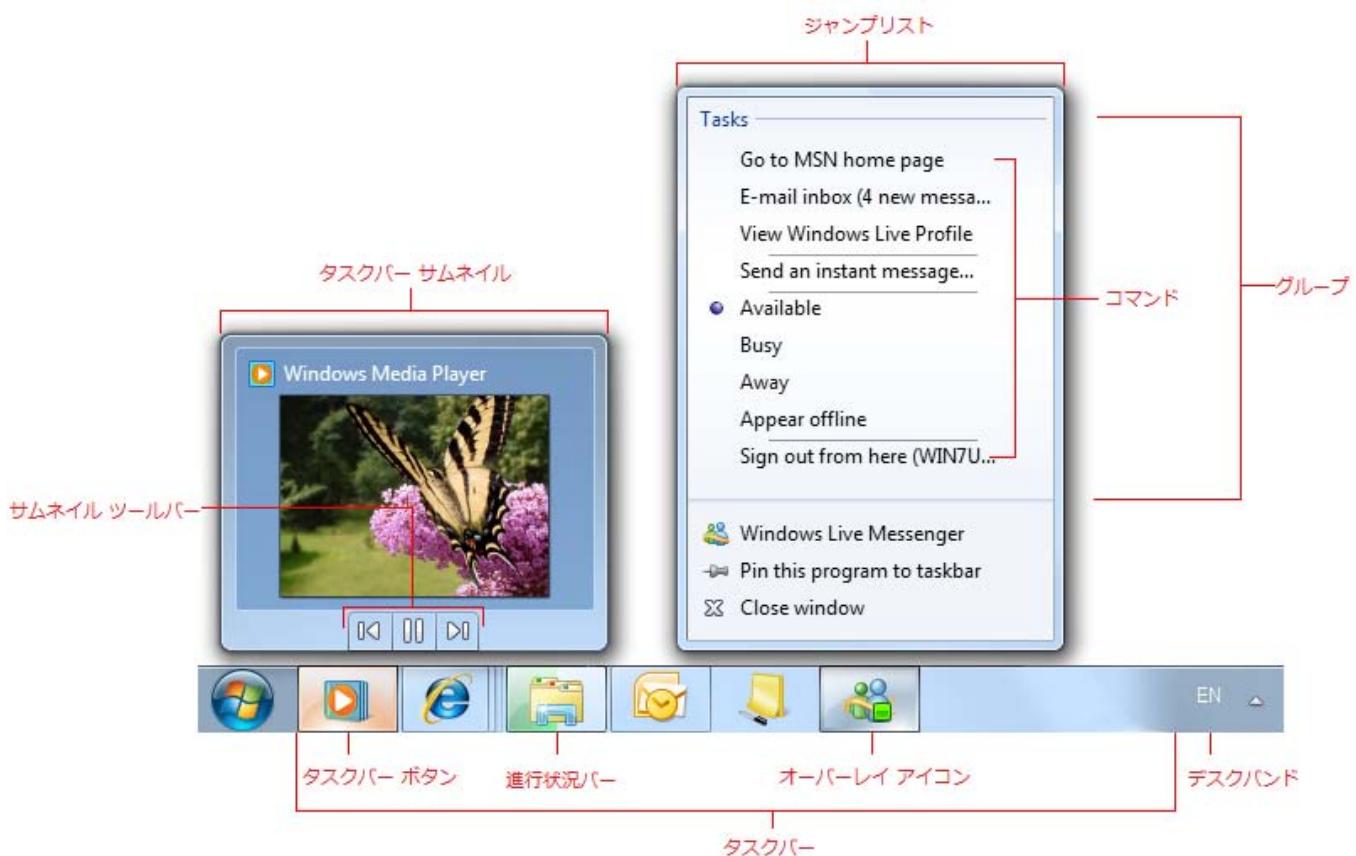
進行状況バー

デスクバンド

テキスト

ドキュメント

"タスクバー" は、デスクトップに表示されるプログラムのアクセス ポイントです。プログラムが最小化されている場合でも、アクセス ポイントの役割を果たします。このようなプログラムを、"デスクトッププレゼンス"を持つといいます。タスクバーを使用すると、ユーザーは開いているメイン ウィンドウと特定のサブ ウィンドウをデスクトップ上に表示でき、これらのウィンドウの表示をすばやく切り替えることができます。



### Microsoft® Windows タスクバー

タスクバー上のコントロールは "タスクバー ボタン" と呼ばれます。プログラムによってメイン ウィンドウ(または特定の特徴を持つサブ ウィンドウ)が作成されると、Windows はそのウィンドウのタスクバー ボタンを追加し、ウィンドウが閉じられるとタスクバー ボタンを削除します。プログラムによっては、タスクバー ボタンに最小化する代わりに "デスクバンド" に最小化する場合があります。これにより、ユーザーはプログラムを最小化した状態で重要なコマンドにアクセスできます。

Windows 7 用にデザインされたプログラムでは、以下の新しいタスクバー ボタンの機能を活用できます。

- ・ "ジャンプリスト" は、プログラムが現在実行されていなくても、プログラムのタスクバー ボタンやスタート メニューの項目からアクセスできるコンテキストメニューを使用して、よく使用するアクセス先(ファイル、フォルダー、リンクなど)やコマンドへのすばやいアクセスを提供します。
- ・ "サムネイルツールバー" は、特定のウィンドウのよく使用するコマンドへのすばやいアクセスを提供します。サムネイルツールバーは、タスクバー ボタンのサムネイルに表示されます。

- ・ "オーバーレイ アイコン" は、プログラムのタスクバー ボタンのアイコン上で状態の変化を示します。
- ・ "進行状況バー" は、プログラムのタスクバー ボタン上で時間がかかるタスクの進捗状況を示します。
- ・ "サブ ウィンドウのタスクバー ボタン" では、タスクバー ボタンのサムネイルを使用して、ウィンドウ タブ、プロジェクト ウィンドウ、マルチドキュメントインターフェイス (MDI) 子ウィンドウ、およびサブ ウィンドウに直接切り替えることができます。
- ・ "固定タスクバー ボタン" では、プログラム ボタンをタスクバーに固定して、プログラムが実行されていなくてもプログラムにすればやくアクセスできます。

技術的には、タスクバーの範囲は [スタート] ボタンから通知領域までのバー全体に及びますが、一般的にタスクバーはタスクバー ボタンが含まれている領域のみを示します。複数のモニター構成では、1つのモニターにのみタスクバーが表示され、そのモニターが "既定のモニター" となります。

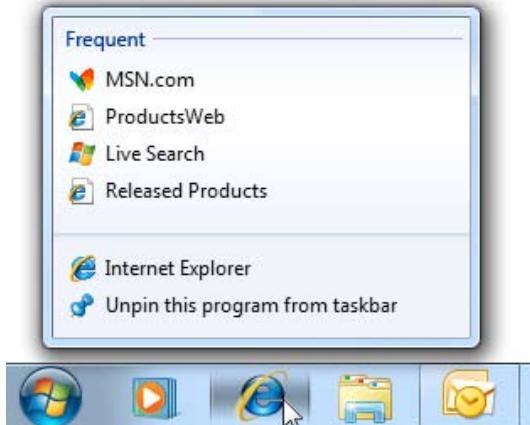
**注:** [スタートメニュー](#)、[デスクトップ](#)、[通知領域](#)、[ウィンドウの管理](#)に関するガイドラインは、それぞれ別の項目として記載しています。

## 適切なユーザーインターフェイスかどうかの判断基準

Windows 7 用にデザインされたプログラムでは、各種のタスクバー ボタンの機能を活用できます。タスクバー ボタンの使用が適切かどうかは、以下の重要な点に基づいて判断します。

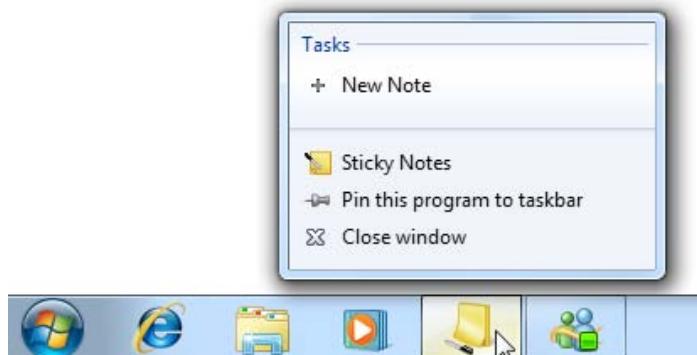
### ジャンプリスト

- ・ ユーザーはプログラムを使用して新しいタスクを起動する必要があるかどうか。該当する場合は、ジャンプリストの使用を検討します。ジャンプリストはその他の目的にも使用できますが、ほとんどのシナリオには新しいタスクの起動が伴います。
- ・ ユーザーは最近使用された、または頻繁に使用されるファイル/フォルダー/リンク/その他のリソースにアクセスする必要があるかどうか。該当する場合は、これらの有用なリソースにアクセスするジャンプリストの使用を検討します。



この例では、Windows Internet Explorer® は ジャンプリスト を使用して頻繁にアクセスするページを提示しています。

- ・ ユーザーは他のプログラムを使用しながら、プログラムが実行されていない場合でも、少数のコマンドにすればやくアクセスする必要があるかどうか。該当する場合は、これらの頻繁に使用されるコマンドを表示する ジャンプリスト の使用を検討します。これらのコマンドはプログラムが実行されていなくても動作し、特定のウィンドウではなくプログラム全体に適用する必要があります。代わりに、特定のウィンドウに適用するコマンドにはサムネイル ツールバーの使用を検討します。



この例では、付箋のアクセサリを使用すると、ユーザーは他のプログラムを使用しながら新しいメモをすぐに作成できます。

- ・ 新しい機能、単一用途の機能、または見つけにくい機能のプロモーションを目的としているかどうか。該当する場合は、ジャンプリストを使用しません。ジャンプリストはこのような目的を想定していません。代わりに、プログラムで直接該当するコマンドを見つけやすいようにします。

## サムネイル ツールバー

以下の条件がすべて適用されていることを確認します。

- コマンドが特定のウィンドウに適用されるかどうか。サムネイル ツールバーは、既存のタスクに適用されるコマンド用であり、ジャンプリスト コマンドは新しいタスクを起動するためのものです。
- ユーザーは他のプログラムを使用しながら、実行中のタスクをすばやく操作する必要があるかどうか。該当する場合は、サムネイル ツールバーが適切です。サムネイル ツールバーでは最大 7 つのコマンドを提示できますが、一般的には最大で 5 つにすることを推奨します。
- コマンドが即時型かどうか。つまり、追加の入力を必要としないコマンドかどうか。サムネイル ツールバーの効率を良くするには即時型のコマンドが必要です。追加の入力が必要なコマンドには、ジャンプリスト の方が適しています。

間違った例:



追加の入力を必要とするコマンドはサムネイル ツールバーに適していません。

- コマンドが直接的かどうか。つまり、ユーザーが 1 回のクリックでコマンドを操作できるかどうか。ツールバーの効率を良くするには直接的なコマンドが必要です。
- コマンドをアイコンでうまく表現することができるかどうか。サムネイル ツールバーのコマンドはテキストラベルではなく、アイコンを使用して提示されます。ジャンプリスト のコマンドはテキストラベルで表示されます。

間違った例:



この例では、アイコンによってコマンドが適切に表現されていません。

## デスクバンド

デスクバンドの使用が適切かどうかは、サムネイル ツールバーと同じ条件を適用して判断します。以下に、各ソリューションの長所と短所をまとめます。

## デスクバンド

### 長所

- すべての最新バージョンの Windows でサポートされています。

### 短所

- Windows 7 では推奨されません。
- 多くのタスクバー領域を消費します。
- デスクバンドを使用するには、ユーザーによるオプトインが必要です。
- 関連するプログラムのコンテキストで提示されません。

- プログラムごとにデスクバンドを1つしか表示できません。

## サムネイル ツールバー

### 長所

- 追加のタスクバー領域を消費しません。
- ユーザー構成なしで、既定で機能が動作します。
- コマンドで多くの画面領域を使用できます。
- 関連するプログラムのサムネイルのコンテキストで提示されます。
- 任意のタスクバー ボタンのサムネイルに対してツールバーを表示できます。

### 短所

- Windows 7 でのみサポートされています。

重要なコマンドへの効率的ではやいアクセスがプログラムの操作性で重要な部分を占めている場合は、上記の方法を両方とも使用することを検討します。ユーザーが Windows 7 を実行しているときはサムネイル ツールバーを有効にし、ユーザーが以前のバージョンの Windows を実行しているときはデスクバンドを表示します。

## オーバーレイ アイコン

- プログラムが "デスクトッププレゼンス" を持つかどうか。該当しない場合は、代わりに通知領域アイコンを使用します。該当する場合は、状態を通知領域アイコンに配置する代わりにオーバーレイ アイコンの使用を検討します (Windows 7 用にデザインされたプログラムの場合)。これにより、アイコンが常に表示され(大きいアイコンの使用時)、プログラムとその状態が1か所にまとめられます。
- 状態の変化を示すためにオーバーレイ アイコンが一時的に表示されるかどうか。該当する場合は、以下の判断要素に応じて、オーバーレイ アイコンの採用を検討します。
  - 有用で関連性のある状態かどうか。つまり、ユーザーがアイコンを監視し、その情報を見て対応を変える可能性が高いかどうか。該当しない場合は、状態を表示しないで済ませるか、ログ ファイルに書き込みます。

間違った例:



この例では、不必要的オーバーレイ アイコンによってディスクの空き領域が表示されています。

- 他のプログラムの使用時、有用で関連性の高い状態かどうか。該当しない場合は、プログラムのステータスバーまたはその他のプログラムのステータス領域に情報を表示します。



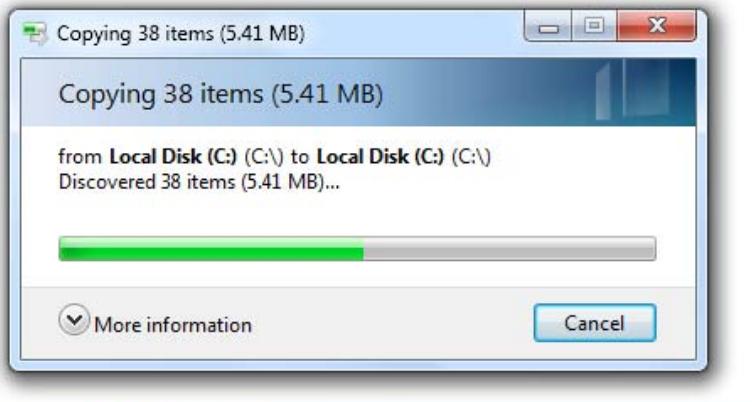
この例では、他のプログラムの使用時、状態は有用ではないのでステータスバーを使用しています。

- 進行状況を示す状態かどうか。該当する場合は、代わりにタスクバー ボタンの進行状況バーを使用します。
- 重大すぐに対応が必要な状態かどうか。該当する場合は、ダイアログ ボックスなど、注意を喚起し、簡単には無視できない方法で情報を表示します。

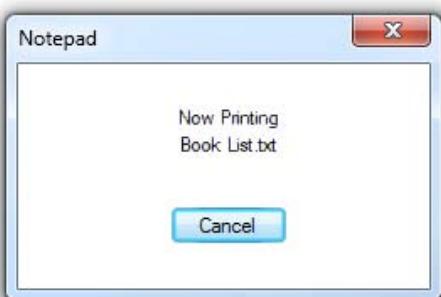
## 進行状況バー

- 他のプログラムの使用時、進行状況のフィードバックが有用で関連性が高いかどうか。つまり、ユーザーが他のプログラムの使用時に進行状況を監視し、その情報を見て対応を変える可能性が高いかどうか。そのような有用で関連性の高い状態の表示には、通常、モードレスな進行状況ダイアログボックスか専用の進行状況ページを使用しますが、ビジー ポインターやアクティビティ インジケーター、ステータスバー上の進行状況バーは使用しません。他のプログラムの使用時、状態が有用でない場合は、単に進行状況のフィードバックを直接、プログラム自体に表示します。

正しい例:



間違った例:



間違った例では、タスクバー ボタンの進行状況バーはあまり有用ではありません。

- 継続的なタスクかどうか。完了しないタスクの場合は、進行状況を示す必要はありません。継続的なタスクの例には、ウイルス対策スキャンやファイルのインデックスがあります。

間違った例:



この例では、継続的なタスクで進行状況を示す必要はありません。

## サブ ウィンドウ タスク バー

- プログラムに、タブ、プロジェクト ウィンドウ、MDI 子ウィンドウ、またはサブ ウィンドウが含まれ、ユーザーは頻繁にこれらの表示を直接切り替えることがあるかどうか。該当する場合は、これらのウィンドウに固有のタスクバー ボタンのサムネイルを表示する方が適していることがあります。

## デザイン コンセプト

### ジャンプリストとサムネイル ツールバーの効率的な使用

ジャンプリストとサムネイル ツールバーを使用すると、ユーザーは効率的にリソースにアクセスしたり、コマンドを実行できます。ただし、これらの機能のプログラムでのサポート方法についてデザインするとき、効率が良くなつて当然と考えないでください。ユーザーがどの機能に必要なコマンドがあるかを正確に予測できなかったり、複数の場所を確認する必要があると、やがてユーザーを苛立たせることになり、これらの機能を使用しなくなります。

以下の場合、ジャンプリストとサムネイル ツールバーを組み合わせると最大の効果が得られます。

- 明確に区別されている。ユーザーは、ジャンプリストでアクセス先やコマンドを探す場合とサムネイル ツールバーで探す場合を

使い分けます。それぞれに明確な目的があるので、ユーザーはこの 2 つのコンテンツを混同することはほとんどありません。一般的に、ジャンプリストは新しいタスクを起動するときに使用され、サムネイルツールバーは他のプログラムを使用しながら実行中のタスクを操作するときに使用されます。

- 有益である。提示されるアクセス先およびコマンドはユーザーが必要としているものにします。ユーザーが必要とする可能性が低い項目は含めないようにします。必要でない場合、項目数を最大限には使用しません。
- 予測可能である。提示されるアクセス先およびコマンドはユーザーが想定しているものにします。ユーザーが複数の場所を探さなくて済むようにします。
- 適切に構成されている。ユーザーが必要なものをすぐに見つけることができるようになります。説明的でありながら簡潔なラベルと適切なアイコンを使用して認識しやすくします。

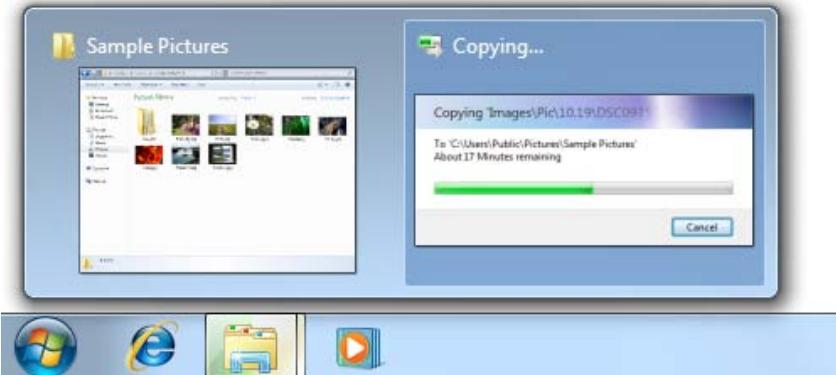
完成するまで、ユーザー リサーチを実施します。最終的に、上記の目的を実現する ジャンプリスト と サムネイルツールバー の組み合わせをデザインできないことがわかった場合は、どちらかのみの使用を検討します。予測可能なコマンドの提示方法が 1 つある方が、混乱を招くようなコマンドの提示方法が 2 つあるよりも効果的です。

## ガイドライン

### タスク バー ボタン

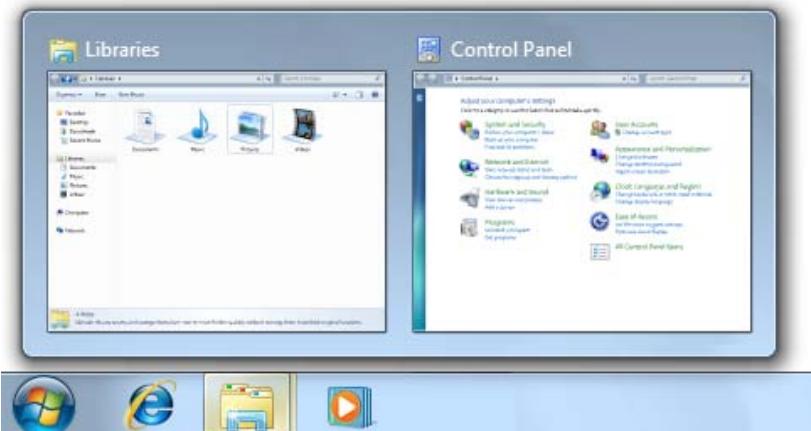
- 以下の種類のウィンドウをタスク バーに表示します (Windows 7 の場合は、タスク バー ボタンのサムネイルを使用します)。
  - メイン ウィンドウ (オーナーを持たないダイアログ ボックスを含みます)
  - プロパティ シート
  - モードレスな進行状況ダイアログ ボックス
  - ウィザード
- Windows 7 の場合は、タスク バー ボタンのサムネイルを使用して、以下の種類のウィンドウを起動元のメイン ウィンドウのタスク バー ボタンとグループ化します。プログラムごとに (特に、各プログラムが別々のプログラムと見なされている場合) タスク バー ボタンが 1 つ配置されるようにします。
  - サブ ウィンドウ
  - 作業領域タブ
  - プロジェクト ウィンドウ
  - MDI 子 ウィンドウ

正しい例:



この例では、サブ ウィンドウはメイン ウィンドウのタスク バー ボタンとグループ化されています。

間違った例:



この例では、「コントロール パネル」は誤って「Windows エクスプローラー」とグループ化されています。ユーザーは、これらのプログラムを別々のものとして見ています。

- ・メイン ウィンドウを元に戻した場合は、すべてのサブ ウィンドウも元に戻す必要があります。これは、サブ ウィンドウ固有のタスクバー ボタンがある場合にも該当します。元に戻す場合は、サブ ウィンドウをメイン ウィンドウの前面に表示します。
- ・Windows 7 の場合、通常デスクトップ プレゼンスを持つプログラムでは、状態を示すタスクバー ボタンを一時的に表示できます。これは、プログラムが通常デスクトップ上に表示され、ユーザーが頻繁にプログラムを操作する場合に限ります。通常デスクトップ プレゼンスを持たずに実行されるプログラムの場合は、常に表示されることがなくとも、代わりに通知領域アイコンを使用するようにします。

間違った例:



この例では、[Windows 同期センター] は一時的なタスクバー ボタンを誤って使用し、状態を表示しています。この場合は、代わりに通知領域アイコンを使用する必要があります。

#### アイコン

- ・タスクバー上で見栄えするプログラム アイコンをデザインします。アイコンのデザインは、適切であることに加え、プログラムの機能とブランドを反映させるようにします。明確で特別なものにし、すべてのアイコンのサイズで適切に表示されるようにします。完成するまで必要な時間を費やします。[Aero style のアイコンのガイドライン](#)に従います。
- ・プログラムでオーバーレイ アイコンを使用する場合は、オーバーレイが適切に処理されるようにプログラムのベース アイコンをデザインします。オーバーレイ アイコンは右下隅に表示されるので、この領域が隠れても影響がないようにアイコンをデザインします。



この例では、プログラムのタスクバー ボタンのアイコンは、右下隅の領域で重要な情報を示しています。

- ・プログラムのベース アイコンにオーバーレイを使用しないようにします。これは、プログラムでオーバーレイ アイコンを使用するかどうかに関係ありません。プログラムのベース アイコンにオーバーレイを使用すると、状態を伝えているのではないかとわかったときに混乱を招きます。

間違った例:



この例では、プログラムのベース アイコンは状態を示しているように見えます。

一般的なアイコンのガイドラインと例については、「[アイコン](#)」を参照してください。

#### オーバーレイ アイコン

- ・オーバーレイ アイコンは、有用で関連性の高い状態を示すためだけに使用します。オーバーレイ アイコンを表示するとユーザーの作業を中断する可能性があることについて考慮してください。そのため、状態の変化は、作業を中断する価値があるほどに重要である必要があります。

間違った例:



これらの例では、オーバーレイ アイコンは作業を中断する価値があるほど重要ではありません。

- ・オーバーレイ アイコンは、一時的な状態を示すために使用します。オーバーレイ アイコンは、頻繁に表示すると価値がなくなるので、通常のプログラムの状態ではアイコンを表示しないようにします。以下の状態になったときに、オーバーレイ アイコンを削除します。

- ・問題の発生を示すアイコン: 問題が解決したときにアイコンを削除します。
- ・新しい何かを通知するアイコン: ユーザーがプログラムをアクティビ化したときにアイコンを削除します。

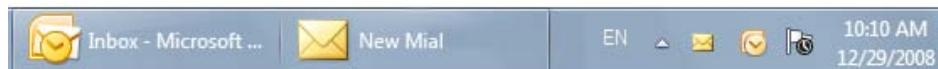
例外: ユーザーが常にプログラムの状態を知る必要がある場合は、継続的にオーバーレイ アイコンを表示できます。



この例では、Windows Live Messenger で常にオーバーレイ アイコンを表示して、ユーザーがいつでも報告された在席状況を確認できるようにしています。

- 問題が解決されたことを示すアイコンは表示しません。代わりに、問題の発生を示していた以前のアイコンを削除します。ユーザーは、通常、問題なくプログラムが実行されるものと想定していることを前提としてください。
- オーバーレイ アイコンまたは通知領域アイコンのどちらかを表示し、両方を表示しないようにします。プログラムでは、下位互換性を保つために両方のメカニズムをサポートできます。ただし、プログラムでオーバーレイ アイコンを使用して状態を表示する場合は、通知領域アイコンを使用して状態を表示しないようにします。

間違った例:



この例では、新着メールのアイコンが重複して表示されています。

- 状態の変化に注意を引くためにタスクバー ボタンを点滅させないようにします。これは目障りです。ユーザーが自分でオーバーレイ アイコンを見つけるようにします。
- できる限り標準的なオーバーレイ アイコンを使用して、状態や状態の変化を示します。以下の標準的なオーバーレイ アイコンを使用します。

オーバーレイ	状態
⚠	警告
✗	エラー
✗	無効または切断
🚫	ブロックまたはオフライン

- カスタム オーバーレイ アイコンの場合、簡単に識別できるデザインを選択します。高画質な、 $16 \times 16$  ピクセルのフルカラー アイコンを使用します。正方形や長方形のアイコンよりも、特徴的な形をしたアイコンを使用するようにします。その他の [Aero style のアイコンのガイドライン](#) も適用します。
- カスタム オーバーレイ アイコンのデザインをシンプルにします。複雑な概念、一般的ではない概念、抽象的な概念を伝えようとしないでください。適切なカスタム アイコンが思いつかない場合は、可能であれば代わりに標準アイコン、エラー アイコン、警告アイコンを使用します。これらのアイコンを効果的に使用すると、さまざまな種類の状態を伝えることができます。
- 状態の変更は最小限にとどめます。オーバーレイ アイコンは、目立ったり、頻繁に変化したり、注意を引くものにならないようにします。視界の隅で変更があっても目は敏感に反応するため、状態の変更はわずかなものにとどめる必要があります。
  - アイコンを頻繁に変更しないようにします。対象の状態が頻繁に変化する場合は、アイコンには高レベルの状態を反映するようにします。

間違った例:



この例では、オーバーレイ アイコンが頻繁に変化し、必要以上にユーザーの注意を引いています。

- アニメーションは使用しません。これは目障りです。
- アイコンを点滅させないようにします。これは目障りです。緊急な対処が必要なイベントの場合は、代わりにダイアログ ボックスを使用します。それ以外で対処が必要なイベントの場合は、通知を使用します。

#### タスクバー ボタンの点滅

- すぐにユーザーの対応が必要なタスクバー ボタンの点滅は控えめに使用して、進行中のタスクを継続して実行できるようにします。タスクバー ボタンが点滅している状態ではユーザーは集中できなくなり、実行している操作を中断して点滅を停止しようとします。入力フォーカスを移動するよりもタスクバー ボタンを点滅させる方が適切ですが、タスクバー ボタンの点滅がわざらわしいことに変わりはありません。ウィンドウを閉じる前にデータを保存する必要があることをユーザーに示すなど、正当な理由で中断が行われるようにします。非アクティブなプログラムでは、すぐに対応する必要がほとんどないないようにします。ユーザーに必要な操作が、プログラムをアクティビ化したり、メッセージを読んだり、状態の変化を確認するだけの場合は、タスクバー ボタンを点滅させないようにします。

迅速な対応が必要ではない場合は、以下の代替手段を検討します。

- [操作の成功通知](#)を使用してタスクが完了したことを示します。
- 何の動作も行わず、次回ユーザーがプログラムをアクティビ化し、問題に注意を向けるまで待ちます。多くの場合、この方法が最適です。
- 非アクティブなプログラムで迅速な対応が必要な場合は、注意を引くためにタスクバー ボタンを点滅させて、強調表示します。これ以外の動作は行いません。ウィンドウを元のサイズに戻したり、アクティビ化したり、音を鳴らしたりしないでください。代わりに、ユーザーが選択したウィンドウの状態を尊重し、準備が整ったときにユーザーがウィンドウをアクティビ化できるようにします。
- タスクバー ボタンがあるサブ ウィンドウの場合は、メイン ウィンドウのタスクバー ボタンの代わりにそのボタンを点滅させま

す。これにより、直接ウィンドウにユーザーの注意を向けることができます。

- タスクバー ボタンがないサブ ウィンドウの場合は、メイン ウィンドウのタスクバー ボタンを点滅させ、サブ ウィンドウをそのプログラムの他のウィンドウの最前面に表示します。注意が必要なサブ ウィンドウは最前面に表示して、ユーザーが確認できるようにします。
- 一度に 1 つのウィンドウのタスクバー ボタンを 1 つだけ点滅させます。複数のボタンの点滅は必要で、目障りです。
- プログラムがアクティブになったときに、タスクバー ボタンの強調表示を解除します。
- プログラムがアクティブになったときに、明確な何らかの操作を提示するようにします。通常、質問または操作を開始するダイアログ ボックスを表示します。

#### クリック起動のショートカット

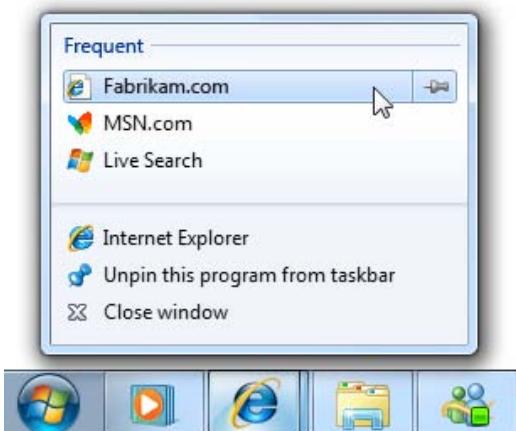
- ユーザーがオプトインする場合のみクリック起動領域にプログラム ショートカットを配置します。クリック起動は Windows 7 から削除されたので、Windows 7 用にデザインされたプログラムでは、クリック起動領域にプログラム ショートカットを追加したり、そのようなオプションを提供する必要はありません。

#### ジャンプリスト

##### デザイン

- ユーザーの日常的なタスクの目的を満たすようにジャンプリストをデザインします。以下の点に留意します。
  - プログラムの目的。ユーザーが次に行う可能性が最も高い操作について考えます。ドキュメント作成プログラムの場合、ユーザーは最近使ったドキュメントに戻る傾向にあります。既存のコンテンツを表示するプログラムの場合、ユーザーはよく使用するリソースにアクセスすることが考えられます。その他のプログラムの場合、ユーザーは新着メッセージを読む、新しいビデオを見る、次の会議を確認するなど、以前実行したことのないタスクを実行する傾向にあります。
  - ユーザーが最も関心のある内容。ユーザーがその他の手段の代わりにジャンプリストを使用する理由について考えます。たとえば、一般的にユーザーは重要と明確に認識しているアクセス先(ユーザーがリンクバーやお気に入りに配置したり、入力した Web アドレス)に関心があります。ユーザーは間接的、またはほとんど労力をかけずに取得したアクセス先(リダイレクト、またはリンクをクリックしてアクセスした Web アドレス)については関心が低くなります。

正しい例:



間違った例:



間違った例では、ジャンプリストにはユーザーにとって関心のなさそうなアクセス先が多く含まれています。

- 過度にアクセス先を細分化しないようにします。アクセス先の範囲を過度に狭めて特定すると冗長になり、同じ場所にアクセスするのに複数の方法が存在するようになります。たとえば、個々の Web ページを一覧にする代わりに、最上位のホーム ページを一覧にします。曲を一覧にする代わりに、アルバムを一覧にします。

正しい例:

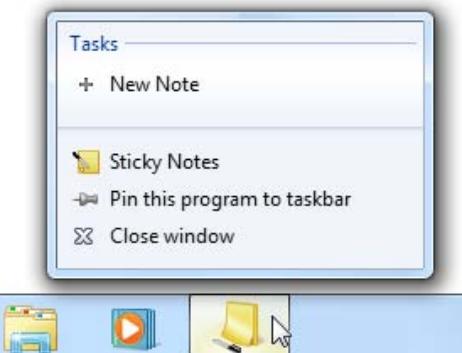


間違った例:



間違った例では、ジャンプリストで一覧表示されている曲は 1 つのアルバムに収まります。

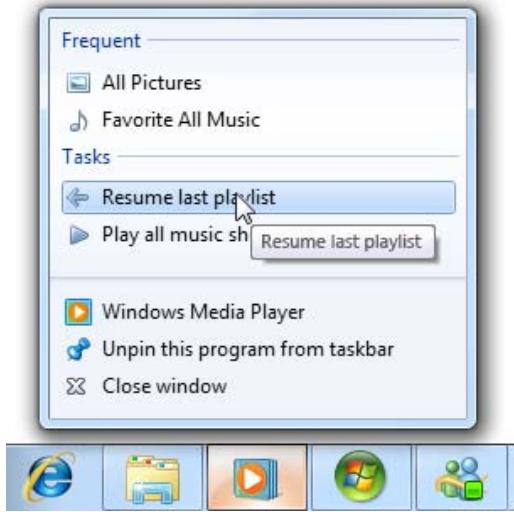
- 必要でない場合は、利用できるジャンプリストのスロットをすべて使用しないようにします。ジャンプリストのコンテンツの重点を最も有用な項目に置きます。プログラムに有用な項目が 3 つしかない場合は 3 つだけ表示します。ジャンプリストの項目が多いほど、特定の項目を見つける手間が必要になります。



この例では、付箋のアクセサリに ジャンプリスト のコマンドが 1 つ表示されています。これは 1 つのコマンドで十分だからです。

- ジャンプリストの項目をユーザーにわかりやすくする必要がある場合のみ、ツールヒントを表示します。不需要でわざらわしくなるので、重複したツールヒントの使用は避けます。その他のツールヒントのガイドラインについては、「ツールヒントと情報ヒント」を参照してください。

間違った例:



この例では、ジャンプリストのツールヒントが重複しています。

#### ジャンプリストの機能とプログラムの機能

- アクセス先やコマンドを使用できる UI をジャンプリストに限定しないようにします。プログラム自体から直接同じアクセス先やコマンドを使用できるようにします。
- 一貫性のあるアクセス先とコマンドラベルの名前を使用します。ジャンプリストの項目は、プログラムから直接アクセスする同等の項目と同じラベルを付けるようにします。
- プログラムが実行されていなくても、プログラムからアクセス先とコマンドを処理できるようにします。これは、一貫性、信頼性、利便性を兼ね備えた操作性のために必要です。

#### グループ化

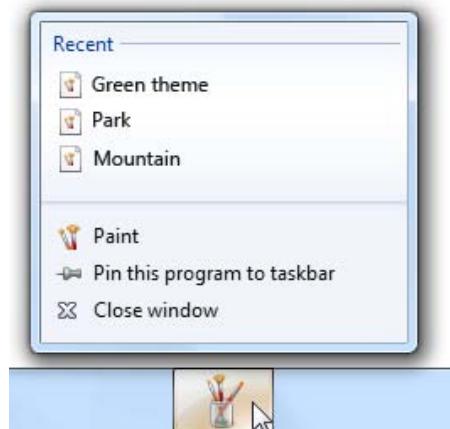
- 少くとも 1 つ、多くて 3 つのグループを使用します。ラベルでその目的を示すためにジャンプリストの項目を常にグループ化します。グループが 4 つ以上あると項目を見つけにくくなります。
- 適切な場合は、以下の標準的なグループ名を使用します。標準的なグループ名の方がユーザーは理解しやすくなります。

最近使ったもの

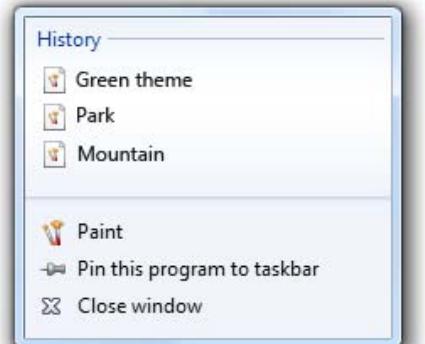
よく使うもの

コマンドにはタスク グループ名が付けられます。これは Windows によって割り当てられ、変更はできません。

正しい例:



間違った例:



/最近使ったもの]の方が馴染みがあるのでグループ名に適しています。履歴と最近使ったものを微妙に使い分けても意味はありません。

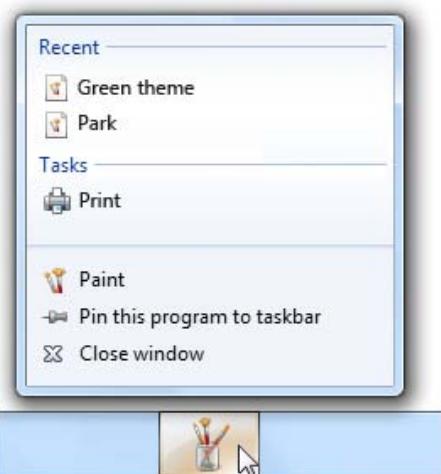
#### コマンド

- プログラムの実行状態、現在のドキュメント、または現在のユーザーに関係なく、固定のコマンドのセットを使用します。特定のウィンドウまたはドキュメントではなく、プログラム全体にコマンドが適用されるようにします。これは、一貫性、信頼性、利便性を兼ね備えた操作性のために必要です。コマンドは削除または無効にしません。

例外: 以下の場合一に、コマンドを代用したり、削除できます。

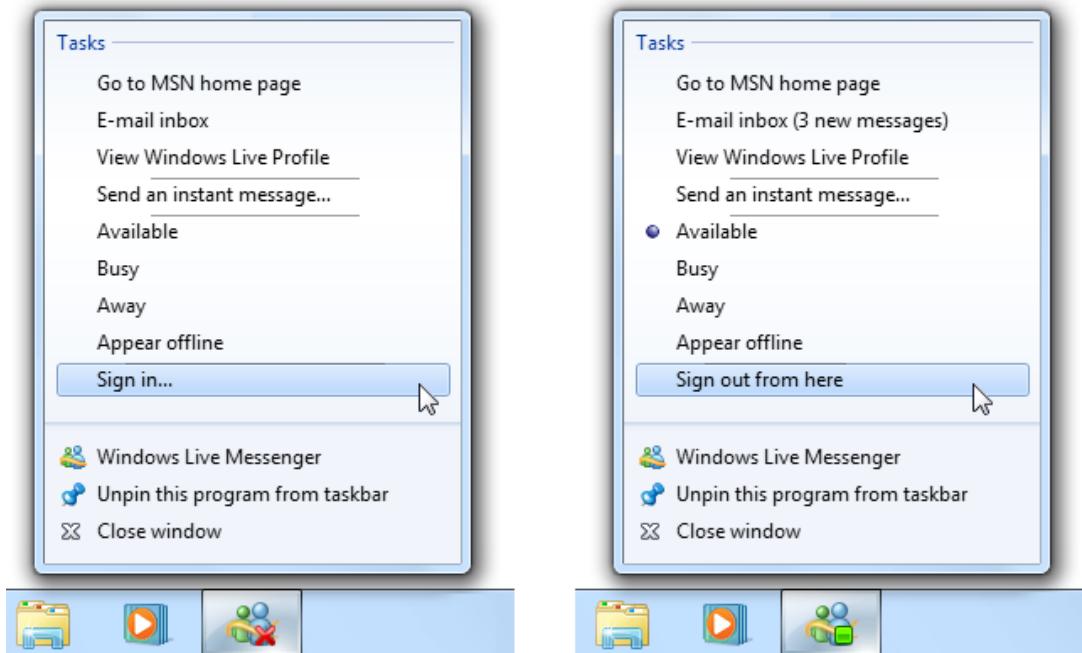
- 相互に排他的なコマンドのセットが1つのコマンドスロットを共有している。ただし、1つのコマンドが常に適用される場合に限ります。
- 特定の機能が使用されるまでコマンドが適用されない。ただし、それ以外でコマンドが常に適用される場合に限ります。

間違った例:



この例では、[印刷]は、現在のドキュメントに応じて実行されるので、ジャンプリストのコマンドに適していません。

正しい例:



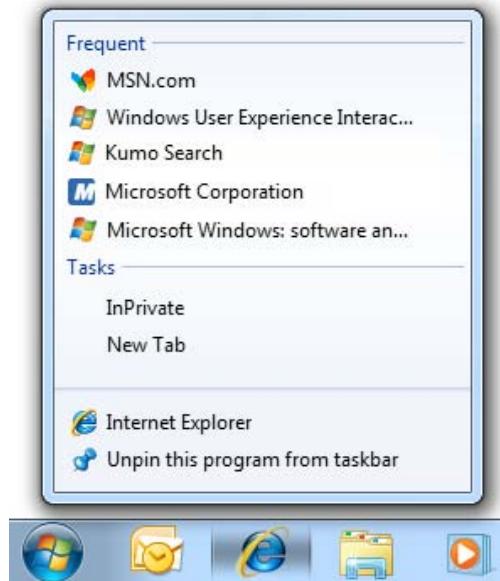
この例では、サインインおよびサインアウトは相互に排他的なコマンドです。また、区切り記号を使用して、関連するコマンドをグループ化します。

- 適切な場合は、以下の標準的なコマンド ラベルを使用します。標準的なコマンド ラベルの方がユーザーは簡単に理解できます。
  - サインイン/サインアウト
  - 新規 <オブジェクト名>
  - <オブジェクト名> を再生
  - <特定のオブジェクト名> に移動
  - 開始
  - 同期
- コマンドを論理的な順序で提示します。一般的な順序には、使用頻度や使用順序があります。関連性の高いコマンドを並べて配置します。必要に応じて、タスク グループ内の関連するコマンドのグループ間には区切り記号を配置します。
- プログラムを開いたり閉じたりするためのコマンドは提供しません。これらのコマンドはすべての ジャンプリスト に組み込まれています。

#### コマンド アイコン

- タスク グループ内にコマンド アイコンを配置するのは、ユーザーがコマンドを簡単に理解、識別、区別できる場合に限ります。特に、プログラム内で使用されるコマンドに対して確立されたアイコンがある場合に限ります。
  - 例外: プログラムにおいて、アクセス先とコマンドの両方を使用し、アクセス先だけ常にアイコンが表示されていて不自然に見える場合は、すべてのコマンドにアイコンを表示することを検討します。

間違った例:

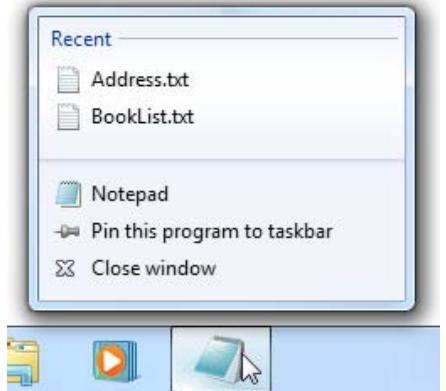


この例では、Internet Explorer ですべてのコマンドにアイコンを表示して、不自然な外観にならないよう

にしています。

## アクセス先

- 現在のユーザーに特有のアクセス先のセットを表示します。アクセス先は動的にする必要がありますが、プログラムの実行状態や現在のドキュメントに影響されないようにします。前述したように、アクセス先はプログラムの目的に合致し、ユーザーが最も関心があり、具体性の度合いが適切なものにします。
- 適切な場合は、"自動" のアクセス先の一覧を使用します。自動アクセス先は Windows によって管理されますが、プログラムは渡された特定のアクセス先を制御します。
  - ドキュメント作成プログラムでは、ユーザーが最近使用したアクセス先を表示する [最近使ったもの] を使用することを検討します。



この例では、Windows のメモ帳は [最近使ったもの] のアクセス先を使用しています。

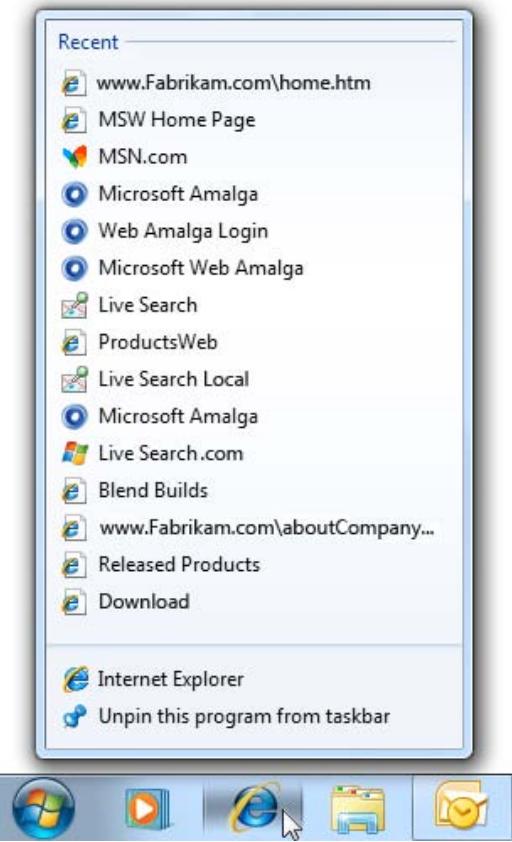
- 既存のコンテンツを示すプログラムでは、ユーザーが頻繁に使用する項目に戻る傾向にある [よく使うもの] を使用することを検討します。[よく使うもの] のアクセス先は、最もよく使用する項目を先頭にして、使用頻度順に並べられます。



この例では、Windows エクスプローラーは [よく使うもの] のアクセス先を使用しています。

- [最近使ったもの] を使用すると無意味なアクセス先が多くなる場合は [よく使うもの] を使用します。ユーザーが多数の異なるアクセス先に移動して、使用したアクセス先にほとんど戻ってこない場合は、[よく使うもの] の一覧の方が一貫性があり適切です。

間違った例:



Windows Internet Explorer で、[最近使ったもの] を使用すると無意味なアクセス先が多くなります。

- [最近使ったもの] と [よく使うもの] の選択が同じくらい適切な場合は、[最近使ったもの] を使用します。この手段の方がユーザーは理解しやすく、予測できるからです。
- [最近使ったもの] を使用し、[もの] メニューで [最近使ったもの] に相当する項目がある場合は、同じ順序で同じ内容が一覧表示されるようにします。ユーザーに対して、これらは同じ一覧であるように表示する必要があります。
- 必要に応じて、カスタムアクセス先の一覧を使用します。プログラムでカスタムアクセス先の一覧の内容および並べ替え順を完全に制御できるので、あらゆる要素で一覧を基準にすることができます。
  - 適切な場合は、[最近使ったもの] または [よく使うもの] のカスタムバージョンを作成します。ただし、自動管理はプログラムで効果を発揮しません。たとえば、プログラムでファイルを開くコマンドだけでなく、さまざまな要素の履歴を残す必要がある場合があります。この場合、同じ名前 ([最近使ったもの] または [よく使うもの]) と並べ替えを使用します。ユーザーにはその違いがわからないからです。
  - または、別の種類のアクセス先を使用して、ユーザーの目的に適切に合わせます。多くの場合、これらの一覧によって、新着メッセージを読む、新しいビデオを見る、次の会議を確認するなど、ユーザーが以前に実行したことのないタスクを簡単に実行できます。

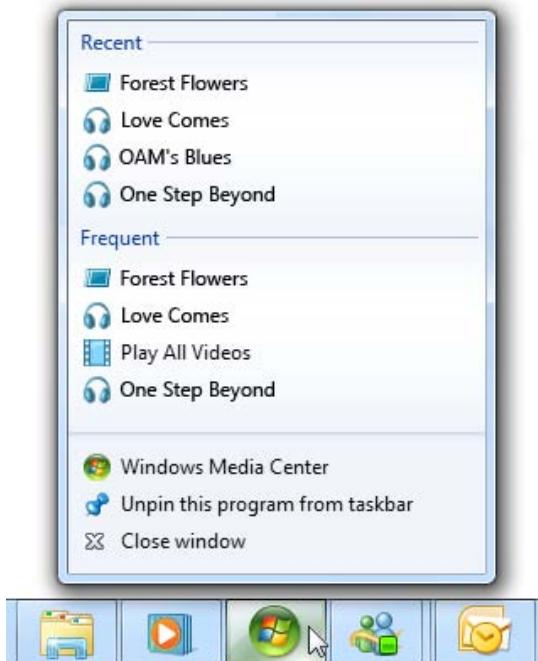


この例では、Windows Media Center はユーザーがまだ見ていない最近録画した番組を一覧表示しています。

す。

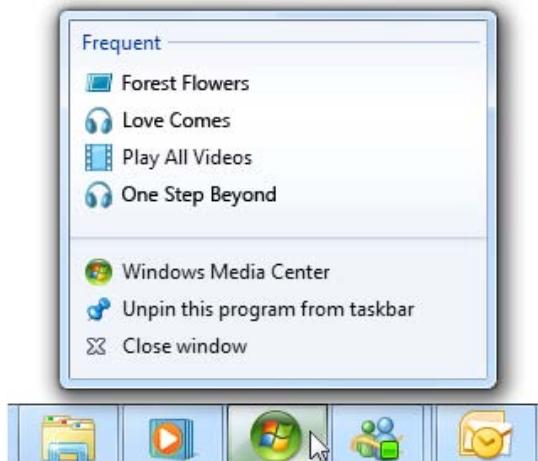
- ユーザーが思い描く一覧のメンタルモデルに対応した並べ替えを選択します。たとえば、作業スタイルの一覧では、次に行う作業を先頭に一覧表示します。明確なメンタルモデルがない場合は、アクセス先の一覧をアルファベット順に並べます。
- 同じデータを異なるビューで表示する複数のアクセス先の一覧は使用しません。代わりに、複数のアクセス先の一覧には、大部分が異なるデータを表示して別々のシナリオをサポートする必要があります。たとえば、[最近使ったもの]の一覧と[よく使うもの]の一覧の両方ではなく、どちらかを提供できます。そうすると、重複する項目がある場合は無駄に見えますが、重複する項目を削除すると混乱を招きます。

間違った例:



この例では、同じアクセス先を異なるビューで表示して無駄に見えます。

正しい例:



この例では、アクセス先の一覧は異なるタスクに対して異なるデータを表示しています。

- プログラムにおいて、プライバシーを保護するためにデータを消去するコマンドがある場合は、アクセス先の一覧も消去します。アクセス先の一覧には機密データが含まれている可能性があります。

サムネイル ツールバー

対話操作

- サムネイル表示のウィンドウに適用される最も重要でよく使用するコマンドを最大7つ表示します。できる限り多くのコマンドを表示する必要があると考えないでください。プログラムにおいて、重要でよく使用するコマンドが3つしかない場合は、3つのみ表示します。

間違った例:



この例では、サムネイルツールバーに重要でないコマンドがあります。

- 直接的で即応性が高いコマンドを使用します。これらのコマンドには、即時型の効果が必要です。コマンドをクリックして追加の入力が必要なドロップダウンメニュー やダイアログ ボックスを表示しないようにします。

間違った例:



サムネイルツールバーのコマンドには即時型の効果が必要です。

- 現在のコンテキストに適用されないコマンド、または直接エラーを引き起こすコマンドを無効にします。そのようなコマンドは非表示にしないでください。そうすることにより、ツールバーの提示に一貫性がなくなります。
- ユーザーが結果を確認したり、すぐに別のコマンドをクリックする可能性が高い場合、ユーザーがコマンドをクリックしたときにサムネイルを閉じないでください。他のウィンドウを表示するコマンドを使用するなど、ユーザーの操作が当面終了していることを示すコマンドのサムネイルを削除します。



この例では、Windows Media® Player で [次へ] をクリックすると継続してサムネイルを表示します。  
ユーザーは他のコマンドを指定する可能性があるからです。



この例では、Windows Live Messenger で [チャット] をクリックするとサムネイルを閉じます。ユーザーはメッセージを送信する可能性が最も高いからです。

#### 提示方法

- サムネイルツールバーのアイコンが [Aero style のアイコンのガイドライン](#)に準拠するようにします。コマンドごとに、高画質の 16 × 16、20 × 20、24 × 24 ピクセルのフルカラー アイコンを表示します。大きいサイズのバージョンは高 dpi 表示モードで使われます。
- 通常の状態やポイントしたときの状態で、ツールバーの背景色に対して、アイコンがはっきりと表示されるようにします。常に、コンテキスト モードおよびハイコントラスト モードでアイコンを評価します。
- 効果を明確に伝えるコマンド アイコン デザインを選択します。優れたデザインのコマンド アイコンは内容が一目でわかり、ユーザーは効率的にコマンドを見つけて理解することができます。
- 認識しやすく、区別できるアイコンを選択します。アイコンには特徴のある形や色を設定します。そうすることで、ユーザーがアイコンのシンボルを記憶していないくとも簡単にコマンドを見つけることができます。一度使用した後は、コマンドを区別するためにツールヒントに頼らずに済みます。
- ツールヒントを使用して各コマンドにラベルを付けます。優れたツールヒントは、ラベルのないコントロールをポイントしたときにラベルを表示します。ガイドラインと例については、[「ツールヒントと情報ヒント」](#)を参照してください。

#### 進行状況バー

- 一般的な進行状況バーのガイドラインに従います。これには、バーの進行を再開または後退させたりしないことや問題の発生を示す場合は赤色の進行状況バーを使用することなどが含まれます。
- 不確定型の進行状況バーは使用しないようにします。不確定型の進行状況バーは進行状況ではなく、動作状況を示します。不確定型の進行状況バーは、ユーザーが動作状況を通常の状態と考えていないまれな状況で使用します。

他のガイドラインについては、[「進行状況バー」](#)を参照してください。

#### デスクバンド

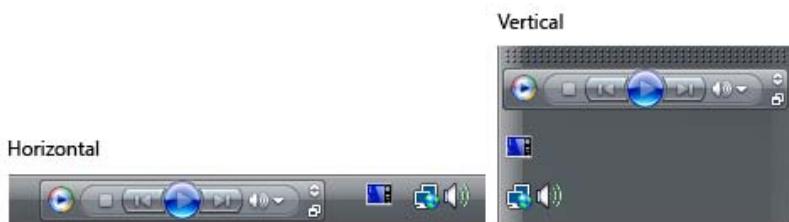
注: デスクバンドは、Windows 7 では推奨されなくなりました。代わりに、タスク バー ボタンとサムネイルツールバーを組み合わせて使用します。プログラムでは、下位互換性を保つために両方のメカニズムをサポートできます。

- ユーザーがオプトインする場合のみデスクバンドを表示します。プログラムのセットアップおよびプロパティでデスクバンドの表示オプションを提示しますが、このオプションは既定で無効にする必要があります。
- デスクバンドはコンパクトかつシンプルに保ちます。ほとんどのシナリオで大半のユーザーがアクセスしないデスクバンド ウィンドウに直接機能を追加しないでください。ただし、あまり一般的に使用されない機能にアクセスするメニューは使用できます。
- デスクバンドに最小化されるプログラムをタスク バーに表示しないようにします。タスク バー ボタンとデスクバンドの両方ではなく、どちらかに最小化します。
- デスクバンドで画面領域を横向きと縦向きの両方で効率的に使用できるようにします。これには、通常、向きに特有のレイアウトが必要です。

正しい例:



間違った例:



間違った例では、デスクバンドは常に同じレイアウトを使用しているので、縦向きの状態で非効率な画面領域を使用しています。

## テキスト

### ウィンドウ タイトル

ウィンドウ タイトルを選択する際には、以下のようにタスク バー上のタイトルの外観について考慮します。

- タスク バーでの表示を最適化するために、特徴的な情報をタイトルの最初の方に簡潔にまとめるようにします。
- モードレスな進行状況ダイアログ ボックスの場合は、まず進行状況をまとめます。例: "66% 完了。"
- 不自然な切り詰めが行われているウィンドウ タイトルは避けます。

間違った例:



この例では、ウィンドウ タイトルが切り詰められて中途半端な表示結果になっています。

### ジャンプ リスト のコマンド

- コマンドは動詞から始めます (英語の場合)。
- センテンス スタイルの大文字化**を使用します。

他のコマンド ラベルのガイドラインについては、「[メニュー](#)」を参照してください。

## ドキュメント

タスク バーに言及するときは、以下のことに留意します。

- バー全体に言及する場合は "タスク バー" とします (英語の場合、小文字の複合語で "taskbar" と表記します)。
- タスク バー上の項目はラベルで具体的に示すか、一般的に "タスク バー ボタン" とします。
- タスク バーのラベルは半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。
- オーバーレイ アイコンに言及する場合は "タスク バー ボタン アイコン" とします。オーバーレイ アイコンの目的がユーザーに通知するものであっても通知とは表現しません。ただし、アイコンによって特定のイベントをユーザーに通知するという表現は使用できます。

例: [新着メール] タスク バー ボタン アイコンは新しい電子メール メッセージを受信したことを通知します。

## 通知領域

通知領域は、通知および状態を表示します。優れたデザインのプログラムでは、ユーザーに不快感を与えることなく通知領域が適切に使用されています。

### 適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

全般

表示するタイミング

表示する場所

アイコン

対話操作

コンテキストメニュー

リッチツールヒント

通知領域のポップアップ

オプションダイアログボックス

通知領域へのプログラムの最小化

テキスト

ドキュメント

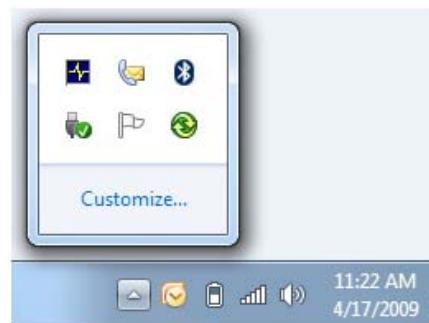
"通知領域"とは、通知および状態の一時的な情報源を表示するタスクバーの一部分のことです。また、通知領域を使用して、デスクトッププレゼンスを持たないシステムやプログラム機能のアイコンを表示することもできます。

既に通知領域のコンテキストが明確に確立されている場合、通知領域内の項目は "通知領域アイコン"、または単に "アイコン"と呼ばれます。



### 通知領域

ユーザーが主体的にWindows® 7のデスクトップを制御できるように、既定では一部の通知領域アイコンは表示されません。アイコンは、ユーザーによって通知領域に昇格されない限り、"通知領域オーバーフロー"に表示されます。



### 通知領域オーバーフロー

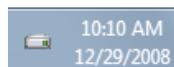
注: [スタートメニュー](#)、[タスクバー](#)、[通知](#)、[パルーン](#)に関するガイドラインは、それぞれ別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- 通知の表示が必要なプログラムかどうか。該当する場合は、通知領域アイコンを使用する必要があります。
- 状態の変化を示すために一時的に表示されるアイコンかどうか。該当する場合は、以下の判断要素に応じて、通知領域アイコンの採用を検討します。
  - 有用で関連性のある状態かどうか。つまり、ユーザーがアイコンを監視し、その情報を見て対応を変える可能性が高いかどうか。該当しない場合は、状態を表示しないで済ませるか、ログファイルに書き込みます。

#### 間違った例:



この例では、ディスクドライブの動作状況を示すアイコンは、ユーザーがこの情報に基づいて対応を変える可能性が低いので不適切です。

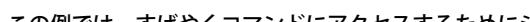
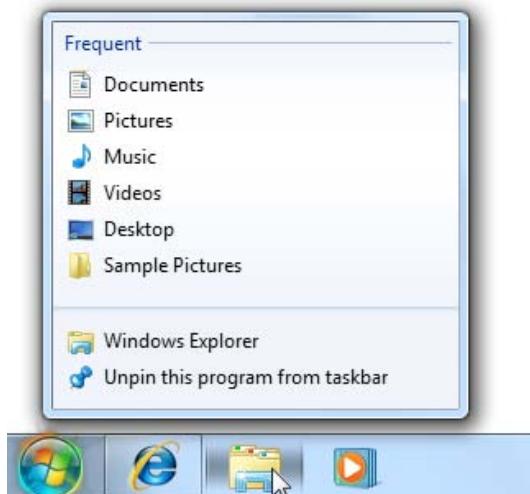
- 重大ですぐに対応が必要な状態かどうか。該当する場合は、[ダイアログボックス](#)など、注意を喚起し、簡単には無視できない方法で情報を表示します。

- Windows 7 用にデザインされたプログラムでは、オーバーレイ アイコンを使用してプログラムのタスク バー ボタンで状態の変化を示したり、タスクバー ボタンの進行状況バーで時間がかかるタスクの進行状況を示すことができます。
- 機能が "デスクトップ プレゼンス" を持っているかどうか。つまり、実行時、デスクトップ上のウィンドウに(場合により最小化されて)機能が表示されるかどうか。該当する場合は、状態をプログラムのステータスバー、他のステータス領域、または Windows 7 の場合は直接タスクバー ボタン上に表示します。デスクトップ プレゼンスを持たない機能の場合は、アイコンを使用してプログラムにアクセスしたり、状態を表示することができます。
  - 主にすばやくプログラムを起動したり、プログラムの機能や設定にアクセスするためのアイコンかどうか。該当する場合は、代わりに [スタート] メニューを使用してプログラムを起動します。通知領域は、すばやくプログラムやコマンドにアクセスすることを目的としていません。



この Windows Vista® の例では、Windows エクスプローラーと Windows Internet Explorer® をすばやく起動するためにクイック起動を使用しています。

Windows 7 用にデザインされたプログラムでは、ユーザーはすばやくプログラムにアクセスするためにタスクバー ボタンを固定できます。プログラムでは、ジャンプリスト またはサムネイル ツールバーを使用して、プログラムのツールバー ボタンからよく使用するコマンドに直接アクセスできます。Windows 7 の既定では、クイック起動領域は表示されません。



この例では、すばやくコマンドにアクセスするためにジャンプリストが使用されています。

## デザイン コンセプト

### Windows デスクトップ

Windows デスクトップには、以下のプログラムへのアクセス ポイントがあります。

- 作業領域。ユーザーが作業を行うことができる画面上の領域で、プログラム、ドキュメント、ショートカットを保存することができます。技術的には、デスクトップにはタスク バーも含まれますが、ほとんどのコンテキストでは単に作業領域のことを指します。
- [スタート] ボタン。すべてのプログラムおよび Windows の特殊な場所(ドキュメント、ピクチャ、ミュージック、ゲーム、コンピューター、コントロール パネル)へのアクセス ポイントです。最近使用したプログラムやドキュメントにすばやくアクセスできる、"最近使った"一覧も含まれます。
- クイック起動。ユーザーが選択したプログラムへの直接的なアクセス ポイントです。クイック起動は Windows 7 から削除されました。
- タスクバー。デスクトップ プレゼンスを持つ、実行中のプログラムへのアクセス ポイントです。技術的には、タスクバーは [スタート] ボタンから通知領域までのバー全体を指しますが、ほとんどのコンテキストでは "タスクバー" はその間のタスク バー ボタンが表示される領域のことを指します。この領域は "タスクバンド" とも呼ばれます。
- デスクバンド。最小化して機能する、長時間実行されるプログラムです(言語バーなど)。デスクバンドに最小化されるプログラムは、最小化するとタスクバー ボタンは表示されません。デスクバンドは、Windows 7 では推奨されません。
- 通知領域。通知や状態の短期的な情報源です。また、デスクトップに表示されないシステム/プログラム関連の機能へのアクセス ポイントです。



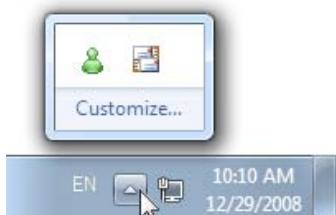
Windows デスクトップのアクセス ポイントには、[スタート] ボタン、タスク バー、デスクバンド、通知領域があります。タスク バー ボタンにはサムネイル機能もあります。

デスクトップは、ユーザーにとって Windows へのエントリ ポイントとなる限られた共有リソースです。制御の主体はユーザーであるようにします。想定された方法でデスクトップ領域を使用する必要があります。それ以外の場合は不適切な使用とみなされます。たとえば、デスクトップ領域を、プログラムや [ブランド](#) をアピールする手段として使用しないようにします。

#### 通知領域を適切に使用する

通知領域は本来、通知と状態を示すための一時的な情報源として想定されていました。その効率と便利さから、開発者はプログラムの起動やコマンドの実行など、通知領域に他の目的を持たせるようになりました。残念ながら、こうした目的が追加されることで通知領域が広がり乱雑になって、他のデスクトップのアクセス ポイントとその目的が混同されるようになりました。

Windows XP では、領域を折りたたみ可能にして使用されていないアイコンを非表示にすることで、大きな問題に対処しました。Windows Vista では、不必要なわざらわしい通知を削除することで煩雑さに対処しました。Windows 7 では、通知元としての本来の目的に通知の重点を置いて、一歩先に進めました。Windows 7 では、ほとんどのアイコンは既定で非表示になっていますが、ユーザーは手動でアイコンを通知領域に昇格できます。ユーザーをデスクトップ制御の主体にするには、プログラムによってこの昇格が自動的に実行されないようにします。ただし、Windows ではアイコンを一時的に昇格させて非表示のアイコンの通知を表示します。



Windows 7 では、通知領域のアイコンのほとんどは既定で非表示になっています。

さらに、Windows 7 のタスク バー ボタンではさまざまな機能が直接サポートされています。具体的には、以下の機能を使用できます。

- よく使用するコマンドにすばやくアクセスするためのジャンプリストおよびサムネイルツールバー。
- 実行中のプログラムの状態を示すオーバーレイ アイコン。
- 時間がかかるタスクの進行状況を示すタスク バー ボタンの進行状況バー。

つまり、プログラムがデスクトップ プrezensを持つ場合は、上記の目的には Windows 7 のタスク バー ボタンの機能を最大限に活用します。通知領域アイコンについては、通知および状態を表示するという目的に重点を置きます。

#### ユーザーを制御の主体にする

ユーザーを制御の主体にすることは、通知領域の適切な使用にとどまりません。アイコンの特性に応じて、以下の操作をユーザーに任せることもできます。

- アイコンの削除。アイコンが関連性の高い有用な状態を表示する場合でも、ユーザーが表示を望まない場合があります。Windows では、ユーザーはアイコンを非表示にできますが、この機能は見つけにくくなっています。ユーザーを制御の主体にするには、アイコンを通知領域に表示するオプションをアイコンのコンテキストメニューに用意します。アイコンの削除によって、実行されるプログラム、機能、または処理に影響がないようにします。
- 表示する通知の種類の選択。有用で関連性の高い通知であっても、ユーザーにとって表示する必要がない通知の可能性もあります。特に、FYI 通知の場合がそうです。重要度の低い通知を有効にするかどうかはユーザーの選択に任せます。

- オプション機能の中止。アイコンを使用して、デスクトッププレゼンスを持たない機能の状態を表示します。このような機能は、印刷、インデックス化、スキャン、同期化など、時間がかかるオプションのバックグラウンドタスクによくあります。システムパフォーマンスを向上させる、消費電力を減らす、または機能がオフラインであるという理由から、ユーザーはこのような機能を中断する場合があります。
- プログラムの終了。以下のうち適切なオプションを提供します。
  - プログラムの一時的終了。プログラムを終了し、Windows の再起動時にプログラムを再起動します。この方法は、セキュリティプログラムなど重要なシステムユーティリティに適しています。
  - プログラムの完全終了。プログラムを終了し、Windows の再起動時に再起動しません(ユーザーが後で再起動するように選択した場合を除く)。ユーザーがプログラムを実行する必要がなくなったか、必要に応じてプログラムを実行する(多くの場合システムパフォーマンスを向上させるため)場合に使用します。

上記のほとんどの設定をアイコンのコンテキストメニューに配置することができますが、大多数のユーザーにはプログラムの既定の操作が適しています。ユーザーが機能を無効にすることを期待して、既定ですべてを有効にしないでください。そうではなく、重要な機能を既定で有効にし、ユーザーは必要に応じて機能を追加で有効にします。

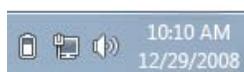
#### 4つの重要な点

- 通知領域は本来の目的以外で使用しません。通知および状態の情報源として、デスクトッププレゼンスを持たない機能に対してのみ使用します。
- ユーザーを制御の主体にします。アイコン、アイコンの通知、実行される機能を制御する適切なオプションを提供します。
- 大多数のユーザーに適した既定のエクスペリエンスを提示します。ユーザーが不必要的機能を無効にするのではなく、ユーザーが必要な機能を有効にします。
- Windows 7 のタスクバー ボタンの機能を最大限に活用して状態を表示し、プログラムで最もよく実行されるタスクの効率を上げます。

## 使用パターン

通知領域アイコンにはいくつかの使用パターンがあります。

**システムの状態** 通知領域アイコンを必要とするシステム機能は、持続したデスクトッププレゼンスを持ちません。通知およびアクセス通知領域アイコンは、通知元としても使用できます。



この例では、バッテリ、ネットワーク、音量のアイコンは、可能な限り継続的に表示されています。

**継続的に表示され、重要なが重大ではないシステムの状態を示し、関連する機能および設定へのアクセスを提供します。**

**バックグラウンドタスクの状態** デスクトッププレゼンスを持たないバックグラウンドプロセスの場合は、通知領域アイコンが必要です。通知領域アイコンは、通知元としても使用できます。



この例では、アクションセンターのアイコンにより、デスクトッププレゼンスを持たなくても、ユーザーはその状態を確認できます。

**一時的なイベントの状態** デスクトッププレゼンスを持つプログラムで一時的に表示され、重要なイベントや状態の変化を示すことがあります。



この例では、印刷および更新のインストールのアイコンが一時的に表示され、重要なイベントや状態の変化を示しています。

**一時的なイベントや状態の変化を示すことができます。**

**一時的な通知元** 単なる通知元には、一時的なアイコンの使用をお勧めします。今後、機能で通知の表示が必要になる可能性があるので、有用で関連性の高い動的な状態を示すアイコンでなければ表示しません。

通知を示すときに一時的に表示されます。一定時間の経過後、または問題に対処したり、タス

クを実行すると  
削除されます。



この例では、プラグ アンド プレイのアイコンが新しいハードウェアの検出に関する通知と共に表示されています。

最小化された單一インスタンスのアプリケーション  
タスクバーの煩雑さを軽減する  
ために、長時間実行される單一インスタンスのアプリケーションを通知領域アイコンに最小化できます。

この Windows Vista の例では、Outlook および Windows Live™ Messenger は、通知領域アイコンに最小化されている単一インスタンスのアプリケーションです。

以下の条件がすべて適用される場合のみ、このパターンの使用を検討します。

- アプリケーションに存在するのが単一インスタンスのみ。
- アプリケーションが長時間実行される。
- アイコンによって状態が表示される。
- アイコンが通知元になる。
- この操作は任意で、ユーザーによるオプトインが必要。

上記の条件がすべて適用される場合、アイコンに最小化することで、1つあれば十分なアクセス ポイントを2つ用意しなくて済みます。

注: このアイコンのパターンは、Windows 7 では推奨されなくなりました。デスクトップ プレゼンスを持つプログラムの場合は、代わりに通常のタスクバー ボタンを使用します。



この Windows 7 の例では、通常のタスクバー ボタンは少ない領域で、ジャンプリスト、オーバーレイ アイコン、リッチ サムネイルなど、Windows 7 のタスクバー ボタンの機能によるメリットが得られます。

## ガイドライン

### 全般

- コンポーネントごとに通知領域アイコンを1つだけ表示します。
- 16 × 16, 20 × 20, 24 × 24 ピクセルのバージョンのアイコンを使用します。大きいサイズのバージョンは高 dpi 表示モードで使われます。

### 表示するタイミング

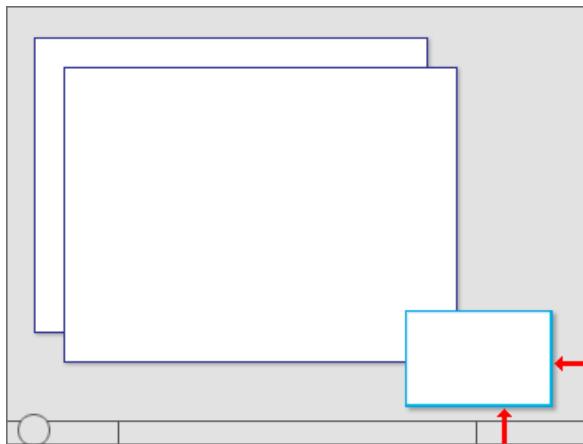
- 一時的な通知元のパターンの場合は、以下の点に留意します。
  - 通知を表示するときに Windows でアイコンを表示します。
  - 以下の通知のデザイン パターンに基づいて、アイコンを削除します。

パターン	削除するタイミング
操作の成功	通知が削除されたとき。
操作の失敗	問題が解決されたとき。
重大ではないシステム イベント	問題が解決されたとき。
任意選択のユーザー タスク	タスクが完了したとき。
FYI (参考)	通知が削除されたとき。

- 一時的なイベントの状態のパターンでは、イベントが発生しているときにアイコンを表示します。
- その他のすべてのパターンでは、ユーザーがアイコンを通知領域に表示するオプションをオフにしない限り、プログラム、機能、または処理が実行中で、アイコンが適切なときにアイコンを表示します(詳細については、「[コンテキスト メニュー](#)」を参照してください)。Windows 7 では、ほとんどのアイコンは既定で非表示になっていますが、ユーザーはアイコンを通知領域に昇格できます。
- 管理者向けのアイコンを[標準ユーザー](#)に対して表示しないようにします。情報は Windows のイベント ログに記録します。

### 表示する場所

- 通知領域アイコンから起動されるウィンドウは、通知領域の近くに表示します。



通知領域アイコンから起動されるウィンドウは、通知領域の近くに表示されます。

## アイコン

- 以下のデザインパターンに基づいてアイコンを選択します。

パターン	アイコンの種類
システムの状態およびアクセス	システム機能のアイコン
バックグラウンドタスクの状態およびアクセス	プログラムまたは機能のアイコン
一時的な通知元	プログラムまたは機能のアイコン
一時的なイベントの状態	プログラムまたは機能のアイコン
最小化された単一インスタンスのアプリケーション	プログラムのアイコン



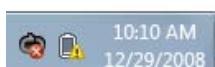
この例では、Outlookは一時的な通知元に電子メール機能のアイコンを使用し、最小化されたアプリケーションに電子メールアプリケーションのアイコンを使用しています。

- 簡単に認識できるアイコンデザインを選択します。正方形や長方形のアイコンよりも、独自の形をしたアイコンを使用するようにします。デザインをシンプルに保ちます。実際のイメージよりもシンボルを使用するようにします。その他の [Aero style のアイコンのガイドライン](#)も適用します。
- 状態や状態の変化を示すには、アイコンのバリエーションまたはオーバーレイを使用します。量や強度などの変化を示すには、アイコンのバリエーションを使用します。その他の種類の状態を示すには、以下の標準的なオーバーレイを使用します。オーバーレイを1つだけ使用し、一貫性を保つために右下隅に配置します。

オーバーレイ	状態
⚠	警告
✗	エラー
✗	無効または切断
🚫	ブロックまたはオフライン



この例では、ワイヤレスおよびバッテリのアイコンは量や強度などの変化を示しています。



この例では、オーバーレイを使用して、エラーおよび警告の状態を示しています。

- 赤、黄、緑の純色はベースアイコンに使用しないようにします。混乱を避けるために、これらの色は状態を伝えるためにのみ使用します。ブランド化でこれらの色を使用する場合は、通知領域のベースアイコンに控えめな色調の色を使用することを検討します。
- 段階的なエスカレーションを使用する場合は、状況の緊急度が増すにつれて、アイコンの外観が段階的に強調されるものを使用します。

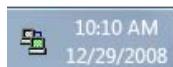


これらの例では、緊急度が増すにつれて、バッテリのアイコンの外観が目立つようになります。

- 状態の変更は最小限にとどめます。通知領域アイコンは、目立ったり、頻繁に変化したり、注意を引くものにならないようにします。視界の隅で変更があっても目は敏感に反応するため、状態の変更はわずかなものにとどめる必要があります。
  - アイコンを頻繁に変更しないようにします。対象の状態が頻繁に変化する場合は、アイコンには高レベルの状態を反映するよう

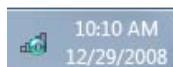
にします。

間違った例:



この例では、モードムのアイコンがハードウェアのモードムのように点滅していますが、この状態の変化はユーザーにとって重要ではありません。

- 長時間のアニメーションを使用して連続的な動作状況を示さないようにします。そのようなアニメーションは目障りです。通知領域にアイコンが存在するだけで十分に連続的な動作状況を示します。
- 重要な状態が一時的に変化している間の進行状況を示す、簡単で控えめなアニメーションは問題ありません。



この例では、ワイヤレスのアイコンはアクティビティ インジケーターを表示して、動作が進行中であることを示しています。

- アイコンを点滅させないようにします。これは目障りです。緊急な対処が必要なイベントの場合は、代わりにダイアログ ボックスを使用します。それ以外で対処が必要なイベントの場合は、通知を使用します。
- 通知領域アイコンを無効にしないようにします。アイコンが現在適用されていない場合は削除します。ただし、ユーザーがアイコンから有効にできるようにする場合は、有効を示すアイコンの上に無効な状態のオーバーレイを表示します。



この例では、ユーザーはアイコンから音声出力を有効にできます。

一般的なアイコンのガイドラインと例については、「[アイコン](#)」を参照してください。

#### 対話操作

注: マウス ボタンを押し下げたときではなく、離したときに以下のクリック イベントを発生させる必要があります。

#### ホバー

- アイコンの内容を示すツールヒントまたは情報ヒントを表示します。

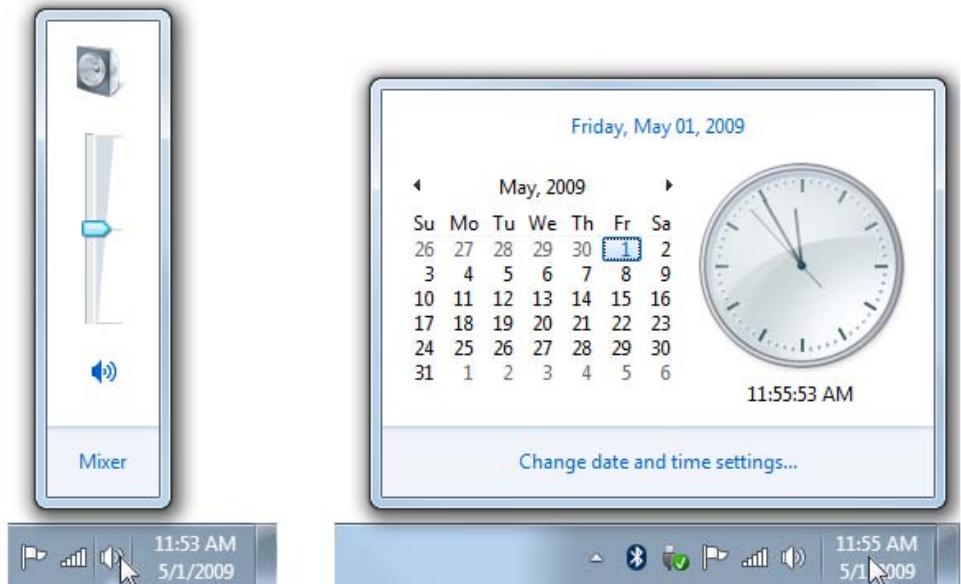


この例では、アイコンをポイントしたときにツールヒントを使用してアイコンを説明しています。

情報ヒントのテキストのガイドラインについては、このトピックの「[テキスト](#)」を参照してください。

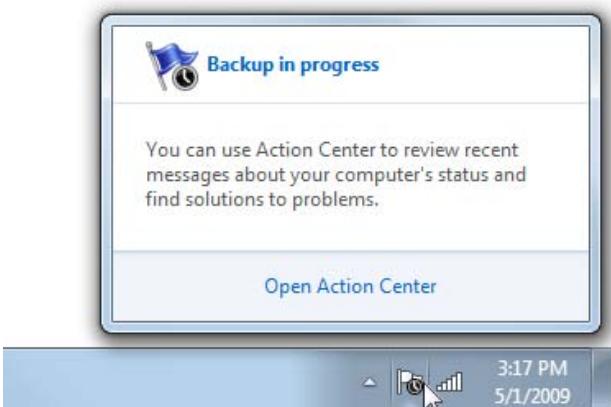
#### 左ボタンのシングルクリック

- ユーザーが表示を必要とする可能性が最も高いものを表示します。たとえば、以下のようなものがあります。
- ポップアップ ウィンドウ、ダイアログ ボックス、または最もよく使用される設定および一般的に実行されるタスクがあるプログラム ウィンドウ。提示方法のガイドラインについては、「[通知領域のポップアップ](#)」を参照してください。



これらの例では、左クリックによって、最もよく使用される設定が含まれているポップアップ ウィンドウが表示されています。

- 状態を示すポップアップ。



この例では、左クリックによって、状態を示すポップアップが表示されています。

- 関連するコントロールパネルアイテム。
- コンテキストメニュー。

ユーザーは何かを表示するとき、左ボタンをシングルクリックするものと想定しています。何も表示されないと、通知領域アイコンが反応していないように見えます。

- その他の選択肢が適用されない場合のみ、コンテキストメニューを表示します。既定のコマンドは太字で表示します。この場合、混乱を避けるために、右クリックで表示される内容と同じコンテキストメニューを表示します。
- ダイアログ ボックスよりもポップアップ ウィンドウを優先的に使用します。これにより、軽快さが増します。最も一般的な設定のみ表示し、簡単な操作で即時型の効果が得られるようにします。ユーザーがウィンドウの外側のどこかをクリックしたときに、ポップアップ ウィンドウを閉じます。
- 関連するアイコンの近くに小さいウィンドウを表示します。ただし、コントロールパネルアイテムなど大きいウィンドウは既定のモニターの中央に表示できます。

#### 左ボタンのダブルクリック

- コンテキストメニューの既定のコマンドを実行します。通常、関連するコントロールパネルアイテム、プロパティ シート、プログラム ウィンドウなど、アイコンに関連付けられたプライマリ UI が表示されます。
- 既定のコマンドがない場合は、左ボタンのシングルクリックと同じ操作を実行します。

#### 右クリック

- コンテキストメニューを表示します。既定のコマンドは太字で表示します。

#### コンテキストメニュー

- 関連するアイコンの近くにコンテキストメニューを表示します。ただし、タスクバーから離れた場所に表示します。
- 必要に応じて、コンテキストメニューに以下の項目を(この順序で)含めることができます(引用符 ("") で囲まれたテキストはそのまま

表示する)。

主コマンド  
開く (既定、最初に太字で表示)  
実行  
副コマンド  
<区切り記号>  
中断/再開、有効/無効コマンド (チェックマーク)  
"通知領域アイコンに最小化" (チェックマーク)  
通知のオプトイン (チェックマーク)  
"アイコンを通知領域に表示" (チェックマーク)  
<区切り記号>  
"オプション"  
"終了"

- 適用されないコンテキストメニューの項目は無効にしないで削除します。
- [開く]、[実行]、[中断]/[再開] コマンドについては、開く/実行する/中断する/再開する対象を具体的にする必要があります。



この例では、Windows Defender の [開く] および [実行] コマンドの対象が具体的に示されています。

- 実行中のバックグラウンドタスクには [中断]/[再開] を使用し、それ以外はすべて [有効]/[無効] を使用します。
- チェックマークを使用して状態を示します。状態をすべて一覧表示し、有効にして、現在の状態の隣にチェックマークを配置します。現在の状態を示すために、オプションを無効にしたり、オプションのラベルを変更したりしません。

正しい例:

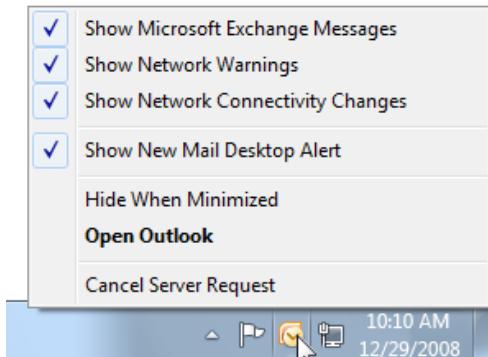


間違った例:



間違った例では、Windows Defender はチェックマークを使用して現在の状態を示す必要があります。

- バックグラウンドタスクにはすべて [中断]/[再開] コマンドを用意する必要があります。コマンドを選択すると一時的にタスクを中断するようにします。システムパフォーマンスを向上させたり、消費電力を減らすために、ユーザーはバックグラウンドタスクを一時的に中断する場合があります。中断されたバックグラウンドタスクは、Windows の再起動時、またはユーザーによって再開されると再起動します。
- 一部のユーザーにとって表示が必要でない通知がプログラムにある場合は、ユーザーが異なる種類の通知をオプトインまたはオプトアウトできるようにします。FYI 通知のパターンはユーザーのオプトインを要求するので、既定で通知を無効にする必要があります。



この例では、Outlookを使用するユーザーはアイコンから受け取る通知を選択できます。

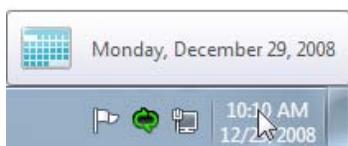
- 「アイコンを通知領域に表示」オプションをオフにしたときに、アイコンを通知領域から削除します。ただし、これにより対象となるプログラム、機能、または処理に影響がないようにします。ユーザーは、プログラムの[オプション]ダイアログボックスからアイコンを再表示できます。Windowsの再起動時にアイコンが自動的に再表示されないようにします。
- [終了]コマンドでは現在のWindowsセッションでプログラムを終了し、アイコンを削除します。プログラムをシャットダウンできない場合、[終了]コマンドを表示しません。Windowsの再起動時にプログラムは再起動します。ユーザーは、[オプション]ダイアログボックスからプログラムを完全に終了することができます。
- [バージョン情報]コマンドは使用しません。このような情報はアイコン、情報ヒント、コンテキストメニューを使用して伝える必要があります。ユーザーが詳細情報を必要としている場合は、プライマリUIを確認できます。
  - 例外: デスクトッププレゼンスを持たないプログラム用のアイコンの場合は、[バージョン情報]コマンドを表示できます。

一般的なコンテキストメニューのガイドラインと例については、「[メニュー](#)」を参照してください。

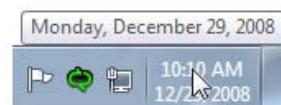
#### リッチツールヒント

- リッチツールヒントは、情報をわかりやすくするためにだけに使用します。単に機能を修飾するために、リッチツールヒントを使用しないでください。情報をわかりやすくするためのリッチ機能を使用できない場合は、代わりにプレーンツールヒントを使用します。

間違った例:



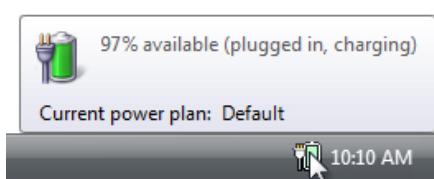
正しい例:



間違った例では、カレンダーのアイコンを表示しても日付がわかりづらくなっています。

- 簡潔な提示方法を使用します。簡潔なテキストと32×32ピクセルアイコンの簡潔なレイアウトを使用します。画面領域が必要なツールヒントを使用すると、ユーザーの操作の邪魔になるおそれがあります(特に、意図せずに表示される場合)。
- 対話型に見えるコントロールまたは要素をリッチツールヒントに配置しないようにします。ツールヒントは対話型ではないので、対話型に見えないようにします。青色や下線付きのテキストは使用しません。

正しい例:



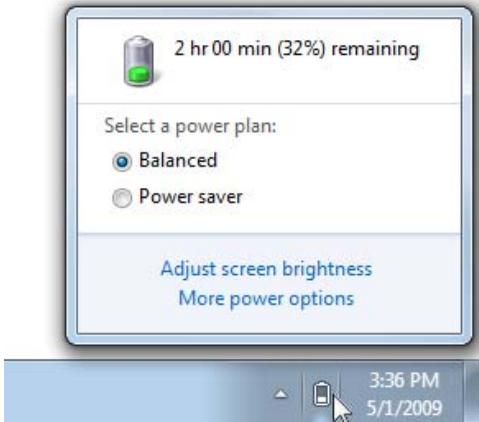
間違った例:



間違った例では、現在の電源プランがリンクに見えますが、クリックはできません。

#### 通知領域のポップアップ

- 適切な場合は、3つのセクションから成る通知領域のポップアップを提示します。
  - 概要。アイコンのツールヒントまたは情報ヒントに表示されている情報と同じ情報(場合によっては詳細情報)を表示します。一貫性を保つために、同じテキストとアイコン、および全体的に同じレイアウトを使用します(リッチツールヒントを使用する場合)。情報ヒントとは異なり、この情報はタッチを使用してアクセスできます。
  - 一般的なタスク。最も一般的に実行されるタスクを直接ポップアップに提示します。
  - 関連するリンク。以下のいずれかの種類のオプションリンクを多くても1つ提示します。
    - コントロールパネルで最もよく実行されるタスクへのリンク。一般的なタスクセクションに提示できないよく実行されるタスクがある場合に提示します。
    - 関連するコントロールパネルアイテムへのリンク。このコントロールパネルアイテムを使用して、ユーザーが一般的なタスクセクションで実行できないタスクを実行できるようにします。
    - 具体的で、関連性の高いヘルプトピックへのリンク。標準的な[ヘルプリンクのガイドライン](#)に従います。



この例では、推奨される提示方法を使用した通知領域のポップアップを示しています。

#### オプション ダイアログ ボックス

- コンテキストメニューから直接アクセスできないオプションは、[オプション] ダイアログボックスに配置する必要があります。このダイアログが機能のコントロールパネルとなる可能性があります。
- 必要に応じて、[オプション] ダイアログボックスに以下の項目を含めることができます(引用符("")で囲まれたテキストはそのまま表示する)。
  - 有効 [機能名] (チェックボックス)
    - このオプションをオフにすると完全にプログラムを終了します。プログラムのコントロールパネルアイテムからプログラムを再起動できます。コンテキストメニューの[終了]コマンドは、現在のWindowsセッションでのみプログラムを終了します。
  - "アイコンを通知領域に表示" (チェックボックス)
    - アイコンを通知領域から削除しても対象の機能には影響しません。
    - このオプションを選択するとユーザーはアイコンを元に戻すことができますが、もちろん、アイコン自体からは復元できません。
- ほとんど使用しない機能は無効にしないと、ユーザーに不快感を与える可能性があります。このような機能は、ユーザーが[オプトイン](#)するようにします。

一般的な[オプション] ダイアログボックスのガイドラインと例については、「[プロパティ ウィンドウ](#)」を参照してください。

#### 通知領域へのプログラムの最小化

注: 通知領域へのプログラム ウィンドウの最小化は、Windows 7では推奨されなくなりました。代わりに通常のタスクバー ボタンを使用します。プログラムでは、下位互換性を保つために両方のメカニズムをサポートできます。

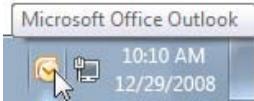
- タスクバーの煩雑さを軽減するには、以下の条件がすべて適用される場合のみ、プログラムを通知領域に最小化できるようにします。
  - プログラムに存在するのは单一インスタンスのみ。
  - プログラムが長時間実行される。
  - アイコンによって状態が表示される。
  - アイコンが通知元になる。
  - この操作は任意で、ユーザーによる[オプトイン](#)が必要。
- [閉じる] ボタンではなく、アプリケーションのタイトルバーの[最小化] ボタンを使用します。

#### テキスト

##### 情報ヒント

- アイコンの情報ヒントには、以下のいずれかの形式を使用する必要があります(会社名は省略可能)。

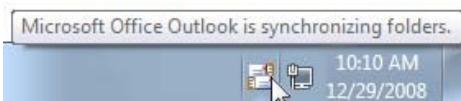
- (会社名) 機能、プログラム、またはデバイス名



- (会社名) 機能、プログラム、またはデバイス名 - 状態の概要



- (会社名) 機能、プログラム、またはデバイス名の状態の説明



- (会社名) 機能、プログラム、またはデバイス名

各項目の状態一覧(1行に1項目)



#### 情報ヒントの表現:

- 最も有用な情報に焦点を当てます。左ボタンのシングルクリックで他の情報を表示します。
- 簡潔にします。語句または簡単な文を使用します。
- ヒントが完結した文として記述されている場合を除いて、末尾に句読点は付けません。
- 不要な語は省略します。ソフトウェアバージョンや他の直接関係のない情報は含めません。

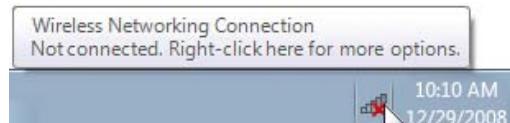
#### 間違った例:



この例では、情報ヒントに直接関係のない情報を表示しています。

- アイコンの操作方法について説明しないようにします。

#### 間違った例:



この例では、ワイヤレスネットワーク接続のアイコンで右クリックするように指示しています。

## ドキュメント

通知領域に言及するときは、以下のことに留意します。

- 通知領域は"通知領域"と示し、"システムトレイ"とはしないでください。

通知領域アイコンに言及するときは、以下のことに留意します。

- アイコンに言及する場合は、大文字と小文字の区別を含め、情報ヒントに記載される名前を正確に引用し、名前の後ろに"アイコン"と付けます。
- 最初に言及する場合は、通知領域についても言及します。
- 可能な場合は見出しテキストを太字にします。そうしない場合で、混乱を避ける必要がある場合は、半角の角かっこ([ ])で囲みます。

例: ネットワークの状態をすぐに確認するには、通知領域の[ネットワーク]アイコンをクリックします。

## Windows デスクトップ ガジェット

適切なユーザーインターフェイスかどうかの判断基準

デザイン コンセプト

使用パターン

ガイドライン

コントロール

状態

対話操作

アニメーションとサウンド

オプション ダイアログ ボックス

Windows との統合

推奨されるサイズと間隔

ドキュメント

"ガジェット" は、シンプルなミニ アプリケーションで、個人に関連する情報やシンプルなタスクへのアクセスを高速に、邪魔にならないように提供します。たとえば、リアルタイムの情報がひとめでわかる天気ガジェットや、ユーザーが監視したいシステム情報を提供する CPU メーター ガジェットがあります。ガジェットは外観が美しく、単一のタスクを非常に効率よく実行できるように最適化されています。

ガジェットは、[スタート] ボタン、タスク バー、通知領域と同様、デスクトップの一部です。通常のウィンドウとは異なり、ガジェットではタスク バー ボタンは表示されません。

Windows Vista® では、通常、サイドバーでガジェットの管理を行います。サイドバーは、デスクトップの片側に表示される領域です。ガジェットはサイドバーにつなげて配置 ("ドッキング") する以外にも、サイドバーから切り離してデスクトップ上の任意の場所に配置することができます。

Windows 7 ではサイドバーがなく、ガジェットはユーザーが選択したデスクトップ上の任意の場所に自由に移動できます。既定では、ガジェットは整理のために画面の端に沿って配置されます。

ガジェットには、大小 2 種類のサイズがあります。小さいサイズは情報を簡潔に表示するために、大きいサイズは情報を詳細に表示するために使用されます。Windows Vista の場合、大小のサイズはガジェットがドッキングされているか離れているかによって切り替わります。ドッキング状態では小さいサイズで、切り離された状態では大きいサイズになります。Windows Vista および Windows 7 の場合、既定では、ガジェットは簡潔なドッキングされた状態でデスクトップに追加されます。

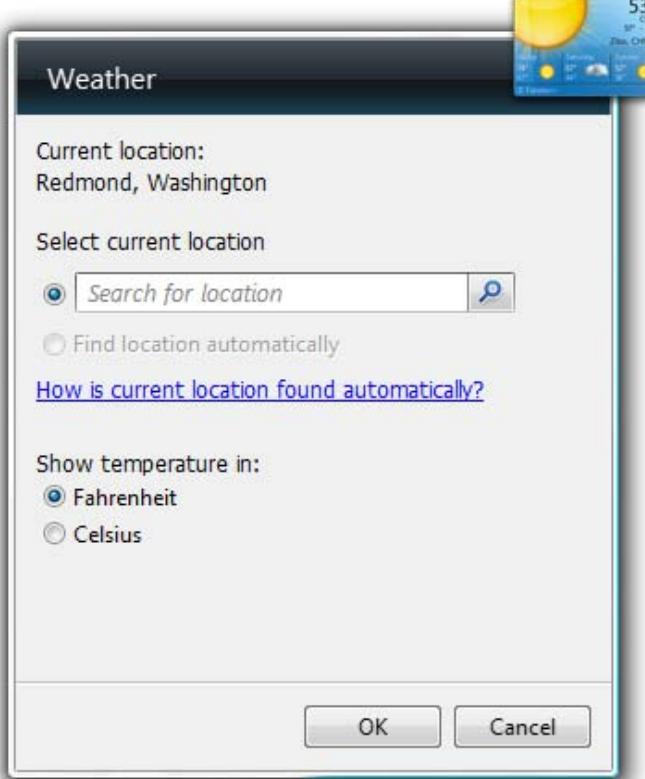
これらのガイドラインは、Windows Vista と Windows 7 の両方のガジェットに適用されます。両方のバージョンの Windows で動作する単一のガジェットを開発することができますが、Windows 7 には一部の新機能があるのでそれらを考慮する必要があります。Windows 7 用に設計する場合、ガジェットがより頻繁に詳細表示の状態で表示されることを考慮する必要があります。

ガジェットには、一時的に詳細情報を表示する "ポップアップ" を追加できます。ポップアップは、ガジェットをクリックすると表示され、ポップアップ以外の任意の場所をクリックすると非表示になります。ポップアップは、どちらのガジェット サイズでも動作します。

最後に、ガジェットには設定やカスタマイズを行うためのオプション ダイアログ ボックスを追加できます。



この例では、天気ガジェット(左下)に気象情報が表示され、RSS ガジェット(中央右)がブログの見出し ポップアップ付きで表示されています。



### オプション ダイアログ ボックスの例

開発者向け情報: ガジェットの作成方法については、[ガジェット開発の概要に関するページ](#)を参照してください。

注: レイアウトに関するガイドラインは、別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- 常に使用できる必要がある機能かどうか。ガジェットにする必要があるのは、ユーザーがすぐにアクセスできる必要

がある、非常に頻繁に使用する機能だけです。

- ユーザーにとって重要度の高い機能かどうか。ユーザーは、すばやく頻繁に確認したい情報を表示するためにだけ、画面の一部を確保します。
- 定期的に変更がある情報かどうか。そうでなければ、ユーザーはその情報のために画面の一部を確保することはありません。
- プログラムを起動するためのもの、またはプログラムに対する意識を高めるためのものかどうか。該当する場合は、代わりに[スタートメニュー](#)を使用します。ガジェットは、プログラムの起動や、プログラムの広告を行うためのものではありません。
- 通知を行うことが目的かどうか。該当する場合は、代わりに[通知](#)を使用します。
- 進行状況を表示することが目的かどうか。該当する場合は、代わりに[進行状況ダイアログ ボックス](#)を使用します。

## デザイン コンセプト

優れたガジェットを作成するためには、重要なポイントがいくつかあります。

優れたガジェットが提供する機能は、いつでも便利です。そうでなければ、限られたデスクトップスペースを有効に使うために、ユーザーはガジェットを閉じることになります。ガジェットは、情報の監視、一日中利用する機能の提供、パズルなどの息抜きに使用すると特に有効です。

効果的なガジェットは、画面の周辺で良好に動作します。ガジェットの機能にアクセスしやすく、かつ邪魔になりません。ガジェットは常にデスクトップに表示されますが、常にアクティブに使用するわけではない点に注意してください。ユーザーは、メインの作業に集中する必要があります。ユーザーは視覚的記憶と運動的記憶によってガジェットの場所を記憶し、必要なときにすばやくアクセスできるようになります。

優れたガジェットは、単一の明確なタスクを効果的に実行します。無関係な機能を省き、コア機能にアクセスしやすくなります。最も重要な情報には大きなフォントを使用し、最もよく行う作業に必要となるクリック数を少なくします。何でも実行しようとする1つのガジェットより、個別のタスクに最適化された複数のガジェットを使う方が便利です。その方が、ユーザーは使用する機能をより正確に制御できます。

間違った例:



正しい例:



間違った例では、オールインワンのガジェットでいろいろなことをしようとしていますが、どれもうまくできません。単一タスク用のガジェットの方が個々のタスクをより効果的に実行でき、ユーザーも、使用的機能をより正確に制御できます。

優れたガジェットは、唯一のタスクを的確に示す視覚的テーマを使用しています。ラベル、アイコン、コンテンツでできたボックスのような見た目のガジェットにならないようにします。重要な情報を伝える、意味のあるイラストを使用します。実物の電化製品の機能を持つガジェットの場合は、ガジェットのデザインをその電化製品に似たもの、またはその電化製品を暗示させるものにします。たとえば、時計ガジェットは実物の時計に、カレンダー ガジェットは実物のカレンダーに似せます。

直接的な対応関係がない場合は、現実世界にあるものの中から、いくつかの重要な点で深く関係しているものを探し、その視覚的要素を使用します。たとえば、CPU メーターは速度計のような見た目であります。これは、どちらもパフォーマンスを示すものだからです。通貨換算ガジェットは、クレジットカードのような見た目になっています。これは、どちらも通貨と旅行に関連するものだからです。このようなゆるい類似性を見つけるのは難しい場合もありますが、これによってガジェットは見た目に刺激的で非常に魅力的なものになります。

許容される例:



より良い例:



より良い例では、クレジットカードのたとえを使って通貨と旅行を暗示し、背景を世界地図にすることでこのテーマを強調しています。また、クレジットカードで使われているテキストにならって大文字のテキストを使用しています。対照的に、最初のガジェットの例は 2 つのフィールドがあるシンプルなボックスです。情報は表示されていますが、推奨例ほどには十分ではなく、説得力も足りません。

適切な視覚的テーマが見つからない場合は、ガジェットの目的を視覚的に表す色や形を使うことができます。



この例では、株価指数と市場指数が動いている方向を示す形と色をガジェットに使用しています。

最後に、どのような視覚タイプを選んだ場合でも、ガジェットの見た目を魅力的にすることが大切です。デスクトップは画面上の最も重要なスペースで、頻繁にユーザーの目に入ります。可能であれば、プロのデザイナーと協力して適切な視覚的テーマを見つけ、ガジェットの見栄えを良くします。

Microsoft® Windows® の他の領域と比較すると、ガジェットは特に形式が自由です。組み合わせて標準的な見た目のユーザーインターフェイスを作成できるような、固定の構成要素のセットはありません。このような柔軟性があるため、非常にすばらしいガジェットも、そうでもないガジェットも作成できるのです。機能や視覚についてはユニークなガジェットにしますが、対話操作は Windows の標準に準拠します。

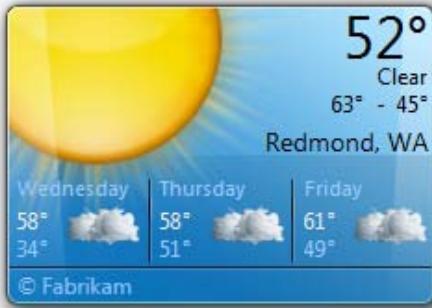
## 5つの重要な点

1. 単一の明確なタスクを効果的に実行する。
2. 便利な機能をいつでも提供する。
3. すぐにアクセスでき、かつ邪魔にならないようにして、ガジェットが画面の周辺で良好に動作できるようとする。
4. ガジェットの唯一のタスクを示す、視覚的に魅力的なテーマを使う。
5. 機能や視覚についてはユニークなガジェットにするが、対話操作は標準的なものにする。

## 使用パターン

ガジェットにはいくつかの使用パターンがあります。

**情報ガジェット**  
1日を通して変化するタイムリーな(多くの場合集約された)情報を表示します。



このガジェットは、最新の気象状況を表示します。

**ユーティリティガジェット**  
役立つ機能を1日中提供します。



ユーザーはすぐにカレンダーを見ることができます。

**ファンガジェット**  
楽しく時間を過ごすことができるツールです。



このピクチャパズルガジェットは、暇つぶし用のゲームの例です。

これらのパターンを分けて考えることが重要です。ファンユーティリティガジェットは、何週間か

経って飽きてくることや、単に邪魔になっていくことが容易に想像できます。

## ガイドライン

### コントロール

- ポイント時にコントロールを表示させます。既定の状態では、必須情報だけを表示します。ほとんどのコントロールは、ユーザーがガジェットをポイントするまで非表示にできます。このように、両方の状態のときに、ガジェットの最も重要な部分に注意を引くことができます。



この例では、ユーザーにとってガジェットがアクティブでない状態のときにデータを読み取ることの方が重要です。ガジェット下部のコントロールは、ユーザーが操作するまでは非表示になっています。

- ラベルを表示するだけのスペースがないため、ツールヒントを使ってすべてのコントロール ラベルを表示します。その他のガイドラインについては、「[ツールヒント](#)」を参照してください。



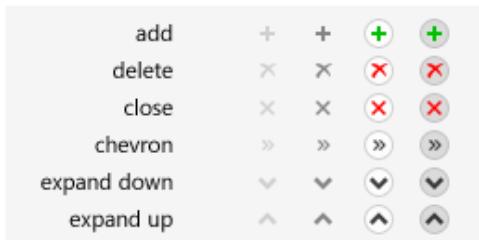
### コントロール ラベル付きのツールヒントの例

- [一般のコントロール](#)と同様に動作し、しかもガジェットのテーマに合ったコントロールを使用します。ガジェットのデザインに合う色を選びます。サイズの制約があるため、標準的な一般コントロールのサイズよりもコントロールを小さくしなければならない場合があります。



この株価ガジェットでは、より小さいカスタム スクロールバーと、ガジェットに合った色を使用しています。

- コマンドには標準的な[グリフ](#)を使用します。標準的なグリフで表すことができないコマンドの場合は、一貫したスタイルの適切なグリフを作成します。ガジェットの外観に合わせて、グリフの色や明るさの調節が必要な場合もあります。すべてのグリフに、停止、ポイント、押下の状態が必要です。ユーザーが考えを変えられるように、グリフはマウスのボタンが元に戻ったときにコミットします。



ガジェットでの一般的なタスクに使用される標準的なグリフの例

- 既定ではスクロールバーの使用を避けます。既定のコンテンツがガジェット内に無理なく収まるようにします。ユーザーがカスタマイズしたコンテンツについては、次のように対応します。
  - 必要な場合は、垂直スクロールバーを使用します。
  - 画面領域が貴重で、ユーザーがコンテンツの位置を完全に制御することが重要ではない場合は、改ページの使用を検討します。



このガジェットでは、ページを切り替えて一度に 4 つの見出しを閲覧できます。

- コンテンツを切り詰めても有用な情報が提供できる場合は切り詰めます(電子メール メッセージの最初の数行など)。コンテンツの全体をポップアップに表示するか、関連するアプリケーションにリンクします。
- ガジェットのテキストには、Windows のシステム フォント (Segoe UIやメイリオなど) を使用します。

## 状態

- 簡易表示(ドッキング)状態では、コア機能を表示します。ガジェットの設計は、ガジェットが常に簡易表示(ドッキング)状態で実行されることを想定して行います。コア機能を決める際は、対象ユーザー、重要なシナリオ、スペースの制約、簡潔さを考慮します。
- 詳細表示(フローティング)状態は追加機能を提供するために使用しますが、理由がない場合は使用しないでください。十分な注意が必要な、またはより時間のかかる複雑なタスクについては、ガジェットに組み込もうとせずに、アプリケーションや Web サイトにリンクします。追加機能が必要ない場合は、簡易表示(ドッキング)状態と詳細表示(フローティング)状態と同じになります。



この例では、簡易表示(ドッキング)状態と詳細表示(フローティング)状態で機能は同じですが、詳細表示(フローティング)状態のガジェットではより多くの内容が表示されています。

- 両方の状態で同じ影を使用し、画面の端に沿って配置したときに正しく整列するようにします。

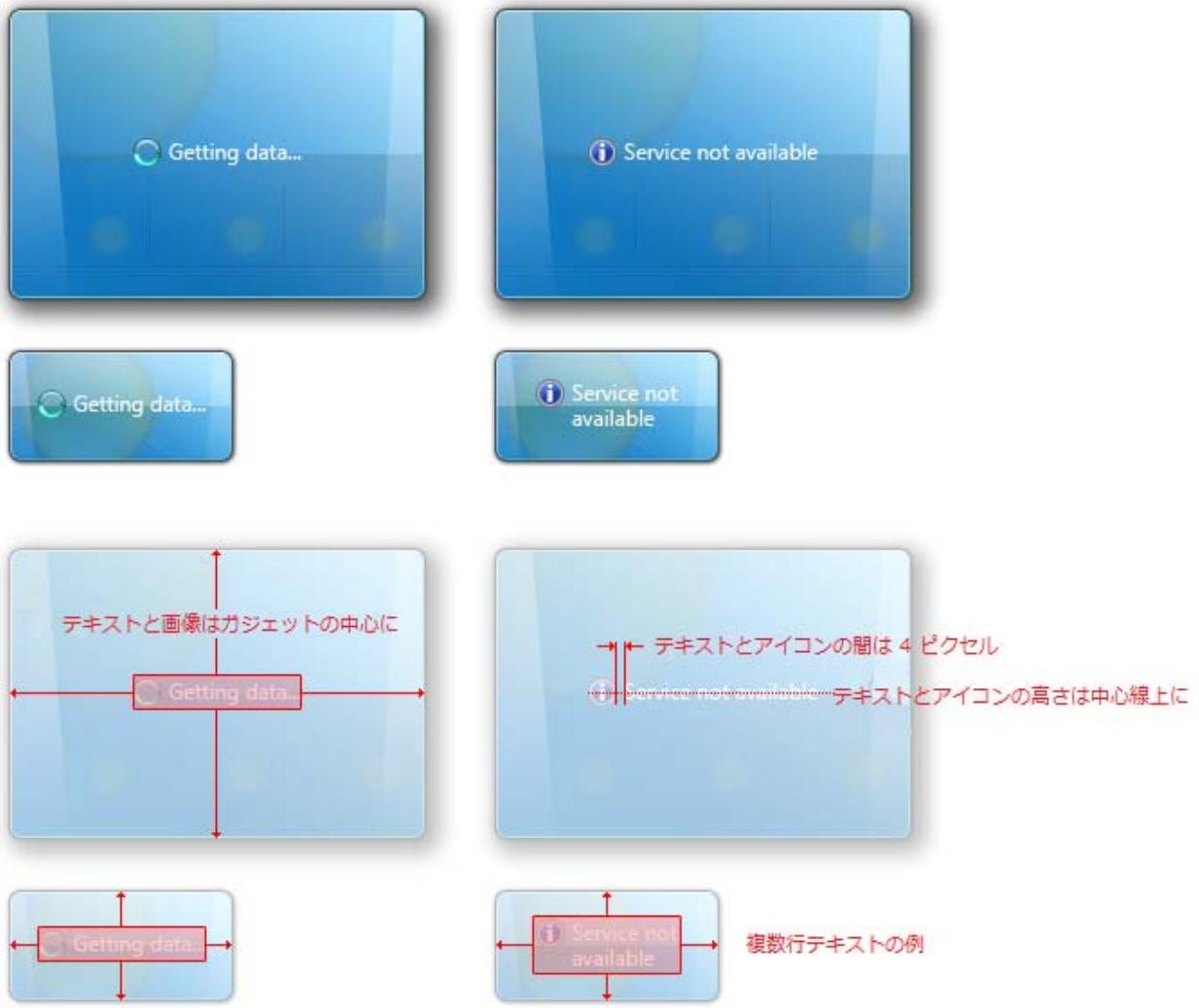
[ポップアップ](#)を使用して、2次的な情報や機能を一時的に表示します。常に利用できる必要がある情報や機能は、直接ガジェット上に配置し、ポップアップには配置しません。また、オプション用にポップアップは使用せず、オプションダイアログボックスを使用します。

次の表では、簡易表示(ドッキング)ガジェット、詳細表示(フローティング)ガジェット、ポップアップ、フル機能のアプリケーションに適した機能を比較します。

状態	適した機能
簡易表示(ドッキング)	常に役立つコア機能。
詳細表示(フローティング)	常に役立つ追加機能。
ポップアップ	場合により役立つ2次的な機能。
フル機能のアプリケーション	十分な注意が必要な、またはより時間のかかる複雑な機能。

- 簡易表示(ドッキング)状態と詳細表示(フローティング)状態で、類似した外観と対話操作を使用します。簡易表示(ドッキング)状態と詳細表示(フローティング)状態でガジェットの外観や動作が大きく異なると、ユーザーを混乱させる可能性があります。簡易表示(ドッキング)状態では領域の制約があるため、よりコンパクトなレイアウトが必要ですが、インターラクティブコンポーネントはだいたい同じ位置になるようにします。どちらの状態でも、同じアクションを実行すると同じ結果が得られる必要があります。
- インストール時に、ガジェットの目的を実際に示すアクションを、ガジェットに実行させることを検討します。これによって、ガジェットの存在感が増し、説明が不要になります。たとえば、カレンダー ガジェットはページを今日の日付に切り替えるようにできます。
- ガジェットの初期設定を求めるないようにします。最も可能性の高い、または最も便利と思われるオプションを既定に選択します。初期設定を必須にすると、ガジェットが必要以上に複雑に感じられます。
- 読み込み中またはオフラインの状態のときはそれをユーザーに通知します。ガジェットがオフラインまたは読み込み中であることをユーザーに知らせます。
  - 読み込み中の状態では、[アクティビティ インジケーター](#)と、何を読み込み中であるか("データを取得中..."など)を説明するテキストを表示します。
  - オフラインの状態では、16×16ピクセルの[情報アイコン](#)と、"サービスを利用できません"などの適切なテキストを表示します。必要に応じて、プレースホルダーのコンテンツの色を薄く表示したり、キャッシュデータを表示することで、ガジェットを認識できるようにします。接続が利用できるようになったら、ガジェットは自動的にアクティブになる必要があります。

接続が失われたことをガジェットで示すことはできますが、注意を引きつける必要はありません。ユーザーは、Web ブラウザーのネットワーク通知領域のアイコンなど、従来の手段によって接続の損失を検出します。



96 dpi のとき、テキストはすべて Segoe UI で 12 ピクセル、行間は 14 ピクセル

これらの例は、オフライン状態と読み込み中の状態を示しています。

- セッション全体を通じて状態を維持します。ユーザーは、ガジェットが前回コンピューターを使用したときと同じ状態であることを期待します。たとえば、途中まで遊んだパズルは、そのときの状態である必要があります。

#### 対話操作

- 標準の Windows ポインターの動作を使用します。たとえば、手の形のポインターはリンクにだけ使用します。
- ガジェットのサイズは自動的に変更しないでください。自動的にサイズが変わると、ユーザーは混乱し、困惑します。なぜなら、ガジェットのサイズが変わると、他のガジェットが (Windows Vista サイドバーと一緒に) 移動したり、重なったり、画面外に出てしまう (Windows 7 の場合) 可能性があります。ポイント時にサイズを変えないでください。必要であれば、クリックしてサイズを変更することができます。
- できる限り、エラー処理やその他のタイプのメッセージの必要性を排除するようにガジェットを設計します。このようなメッセージは、ガジェットの軽快さに反するものです。
- エラー、警告、情報のメッセージを表示する必要がある場合は、メッセージをインプレースで、またはガジェットの別の状態として表示します。このようなメッセージには、ダイアログ ボックスを使用しません。エラーや警告のメッセージについては、適切な 16 × 16 ピクセルの標準アイコンを付けて表示します。ユーザー入力に関する軽度の問題には、アイコンを使用しません。



この例では、ユーザーのインプレース入力の問題なのでアイコンは不要です。

エラー、警告、情報メッセージのアイコンに関するその他のガイドラインについては、「[標準アイコン](#)」を参照してください。

- ガジェットのヘルプは提供しません。その代わりに、見るだけで意味がわかるデザインにします。

## アニメーションとサウンド

- アニメーションは慎重に使用します。不必要で邪魔なアニメーションは使わないようにします。人間の目は、(特に視界周辺の)動きに敏感です。情報を更新する際は、目立たないように更新を行い、自然にユーザーが気が付くのを待つ方が賢明です。何かに注意を引くためにアニメーションを使用する場合は、ユーザーの思考の流れを中断してまでその注意を引く意義や価値があることを確認してください。クロスフェードなどの切り替え効果は、更新が円滑に見える場合がありますが、過度に注意を引いてしまう可能性があります。ガジェットをテストして、アニメーションのスピードや表面積などの可変要素を調整し、円滑さとインパクトの適切なバランスを見つけます。特定のユーザーアクションによって起動するアニメーションを推奨します。アニメーションを適切に使用すると、ガジェットは円滑で洗練されたものになります。
- サウンドは慎重に使用します。サウンドでユーザーの気を散らすことのないようにします。サウンドは、控えめに、ユーザーの操作時にのみ使用します。その他のガイドラインについては、「[サウンド](#)」を参照してください。

## オプション ダイアログ ボックス

- オプション ダイアログ ボックスは、必要な場合にのみ使用します。オプションが不要なガジェットもあります。便利なオプションとしては、コンテンツの追加や削除といった、個人設定および機能のカスタマイズがあります。オプション ダイアログ ボックスがガジェットの大部分と重複する場合は、ユーザーがガジェット上で直接設定できるようにすることを検討します。たとえば、ユーザーはドラッグ アンド ドロップで直接リスト内のアイテムを並べ替えるかもしれません。この機能をオプション ダイアログ ボックスで提供すると、ガジェットの大部分と重複するだけでなく、ユーザーが最終結果を想像しづらくなります。
- オプションはポップアップでは提供しません。
- オプション ダイアログ ボックスへのアクセスは、ガジェット上のオプション グリフを使って提供します。



オプション ダイアログ ボックスには、ガジェットの右上隅にあるオプション グリフからアクセスできます。

- オプション ダイアログ ボックスの外観は、タブのない[プロパティ ウィンドウ](#)のようにします。オプションは、スクロールバーを使用せず1ページで表示できるように制限します。ダイアログ ボックスにコントロールを配置する際は、一般的な[レイアウト](#)のガイドラインに従います。

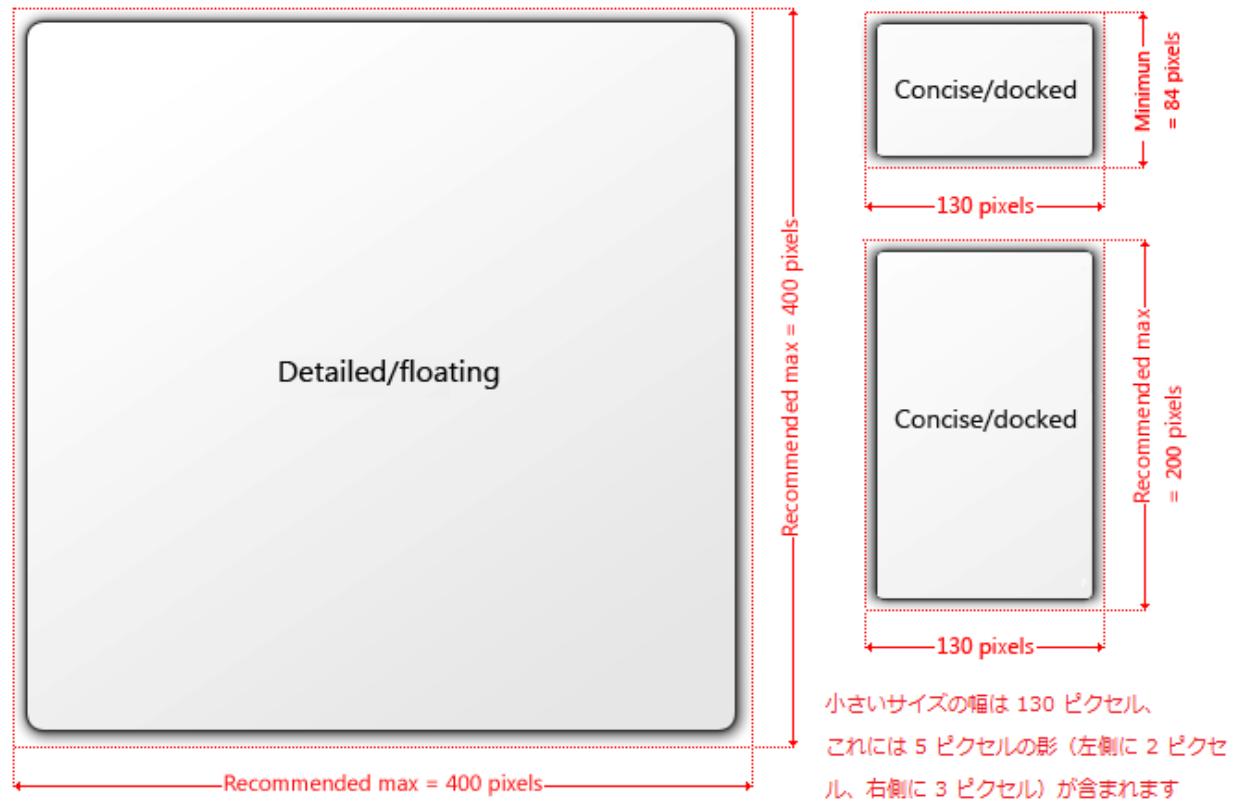
## Windows との統合

- ガジェットのインストールを行う Windows 7 プログラムの場合:
  - プログラムのセットアップ時に、ガジェットをインストールするオプションを提供します。
  - このオプションは、既定ではオフの状態で提示します。

- ユーザーがこのオプションを選択した場合は、ガジェットを 1 つだけインストールします。
- ユーザーがプログラムを実行した後でのみ役立つガジェットの場合は、セットアップ時ではなくプログラムの初回実行時にガジェットをインストールします。たとえば、最新のメールを表示するガジェットは、ユーザーがメール アカウントを設定するまでは意味がありません。
- ガジェットを [Windows SideShow](#) に拡張することを検討します。

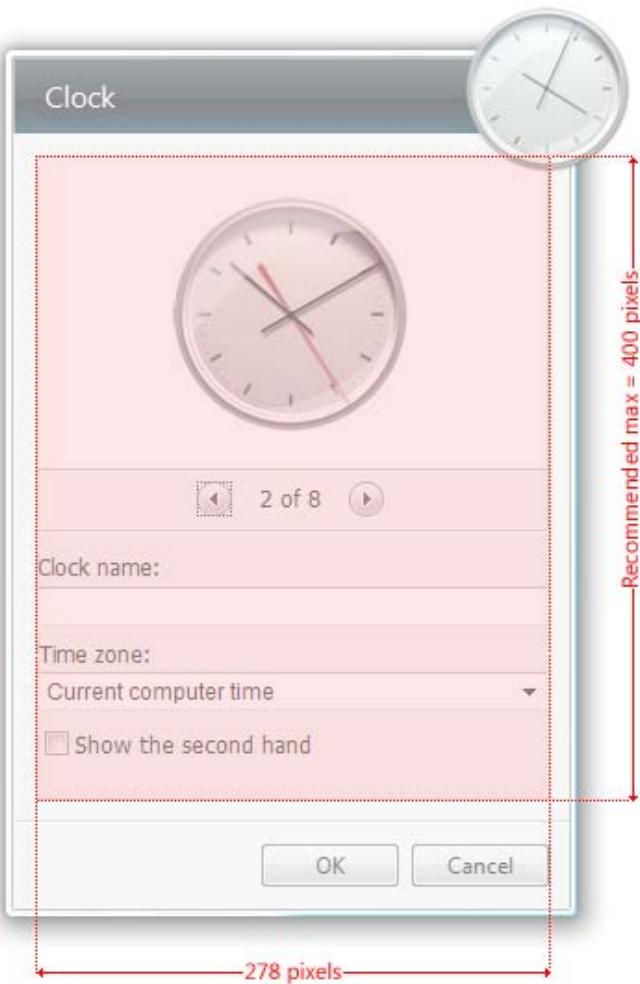
## 推奨されるサイズと間隔

- 簡易表示(ドッキング)状態のガジェットの幅は 130 ピクセルです。影の 5 ピクセル、左側の 2 ピクセル、および右側の 3 ピクセルを含みます。
- 簡易表示(ドッキング)状態のガジェットの高さは最小で 84 ピクセル、推奨される最大値は 200 ピクセルです。スペースを効率よく使用すると、ガジェットはより使いやすくなります。
- 詳細表示(フローティング)状態のガジェットは、400 × 400 ピクセル以内にする必要があります。



## 詳細表示(フローティング)および簡易表示(ドッキング)状態のガジェットの推奨サイズ

- オプションダイアログ ボックスは幅 278 ピクセルのクライアント エリアを持ち、高さは 400 ピクセル以下です。高さは、400 ピクセル以内でコントロールを収納できるように調整する必要があります。
- オプションダイアログ ボックスのテキストは、96 dpi の場合、12 ピクセルの Segoe UI で、行間は 14 ピクセルです。



オプション ダイアログ ボックスの推奨サイズ

## ドキュメント

Windows デスクトップ ガジェットに言及するときは、以下のこと留意します。

- ガジェットの機能を示す場合は、"Windows デスクトップ ガジェット"と呼びます。
- 特定のガジェットを示す場合は、ガジェット名の後に"ガジェット"を付けます。このとき、"ガジェット"の先頭は大文字にします(英語の場合)。  
例: 天気ガジェット、CPU メーター ガジェット
- 総称的にガジェットを示す場合は、"ガジェット"の先頭は大文字にしません(英語の場合)。  
例: 既定では、ガジェットをドッキングします。
- "サイドバー"という名称は、Windows Vista で実行するガジェットについてのみ使用します。"サイドバー"に言及するときは、常に先頭を大文字にします(英語の場合)。
- "ガジェット ギャラリー"に言及するときは、常に先頭を大文字にします(英語の場合)。

## コントロール パネル

適切なユーザーインターフェイスかどうかの判断基準

デザイン コンセプト

使用パターン

ガイドライン

プロパティ シートのコントロール パネル アイテム

タスク フロー の コントロール パネル アイテム

全般

タスク リンク と ボタン

ダイアログ ボックス

ハブ ページ

全般

オブジェクト リスト

対話操作

作業 ウィンドウ

[関連項目] リンク

スパーク ページ

全般

対話操作

対話操作 (中間スパーク ページ)

対話操作 (最終スパーク ページ)

コミット ボタン

[プレビュー] ボタン

リアルタイム の プレビュー

[適用] ボタン

コントロール パネル の 統合

カテゴリー ページ

タスク リンク

検索語

標準ユーザーおよび保護された管理者

デザイン と テーマ

その他

既定値

テキスト

ドキュメント

Microsoft® Windows® の "コントロール パネル" で、ユーザーはシステム レベルの機能を構成し、関連するタスクを実行できます。システム レベルの機能構成の例として、ハードウェアのセットアップと構成、ソフトウェアのセットアップと構成、セキュリティ、システム メンテナンス、ユーザー アカウント管理などがあります。

"コントロール パネル" という用語は、Windows のコントロール パネルに関する機能全体を指します。個々のコントロール パネルは、"コントロール パネル アイテム" と呼ばれます。コントロール パネルのホーム ページ または カテゴリー ページ から直接アクセスできるコントロール パネル アイテムは "最上位" のアイテムと見なされます。最上位のコントロール パネル アイテムに必要な基準については、「[カテゴリー ページ](#)」を参照してください。



典型的なコントロールパネルアイテム。

"コントロールパネルのホーム ページ" は、すべてのコントロールパネルアイテムのメインエントリポイントです。カテゴリー別にアイテムが表示され、最も一般的なタスクも示されます。このページは、[スタート] メニューの [コントロールパネル] をクリックすると表示されます。

"コントロールパネルのカテゴリー ページ" には、1 つのカテゴリーに含まれるアイテムが表示され、最も一般的なタスクも示されます。このページは、ホームページのカテゴリー名をクリックすると表示されます。

コントロールパネルアイテムの実装には、タスク フローまたはプロパティ シートを使用します。Windows Vista® 以降では、ユーザー インターフェイス (UI) としてタスク フローを使用することを推奨します。

開発者向け情報: コントロールパネルアイテムの作成方法については、「[コントロールパネルアイテム](#)」を参照してください。

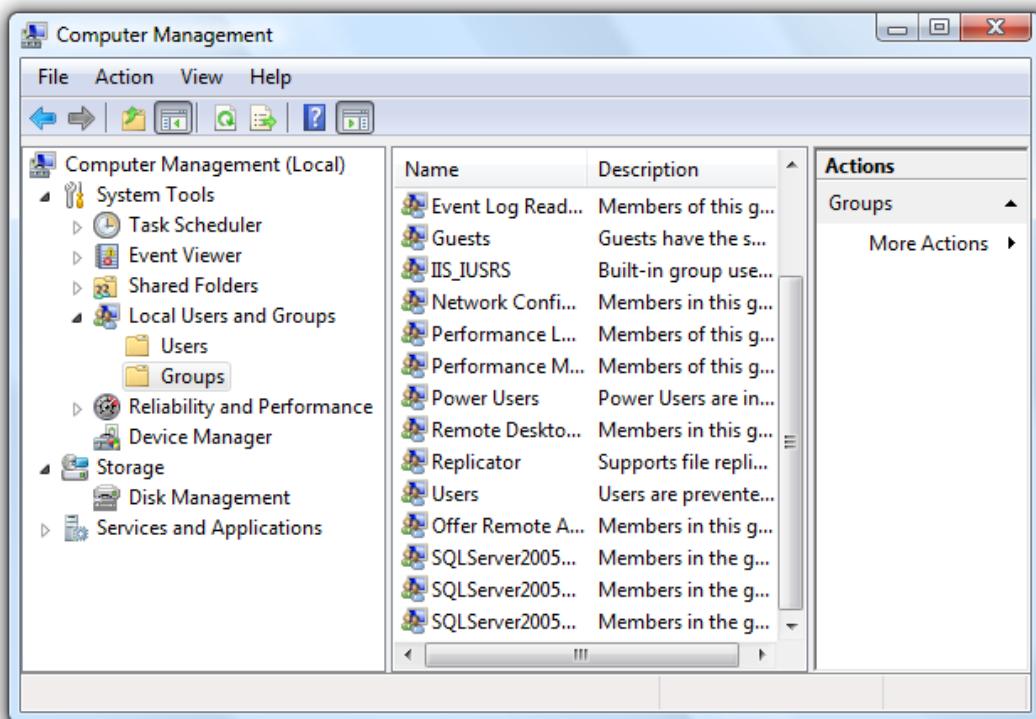
注: [プロパティ シート](#)に関するガイドラインは、別の項目として記載しています。

## 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- システムレベルの機能を構成することが目的かどうか。該当しない場合は、別の統合ポイントを使用します。コントロールパネルを使用する代わりにオプション ダイアログ ボックスを使用して、アプリケーションの機能を UI から直接設定できるようにします。セットアップ、構成、または関連するタスク (トラブルシューティングなど) に使用しないユーティリティは、統合ポイントとして [スタート] メニューを使用します。
- そのシステムレベルの機能に固有の UI があるかどうか。該当する場合は、変更を行うためにユーザーが移動すべき場所にその UI を配置します。たとえば、システムバックアップユーティリティは、コントロールパネルからではなく、ユーティリティのプログラム オプションから構成する必要があります。
- 構成を頻繁に変更する必要があるかどうか。該当する (たとえば、週に数回必要になる) 場合は、コントロールパネルの使用だけでなく、別の方法も検討します。たとえば、Windows のマスター音量設定は、通知領域内の音量アイコンから直接構成できます。設定によっては、自動的に構成できるものもあります。たとえば、Windows エクスプローラーで、アプリケーションのプロパティの [互換性] タブを使用すると、ビデオ モードを手動で変更する代わりに、アプリケーションを 256 色モードで実行できます。
- 対象ユーザーが IT プロフェッショナルであるかどうか。該当する場合は、システム管理タスク専用に設計された [Microsoft 管理コンソール \(MMC\)](#) スナップインを使用してください。一般ユーザー向けのコントロールパネルアイテムと、IT プロフェッショナル

向けの MMC スナップインの両方を用意するのが最適な場合もあります。



この例では、IT プロフェッショナル向けのユーザー管理機能が、[ローカルユーザーとグループ] MMC スナップインによって提供されています。それ以外のユーザーは、ほとんどの場合、コントロールパネルの [ユーザー アカウント] アイテムを使用します。

- その機能が、初期システム構成時にのみ使用される OEM 機能かどうか。該当する場合は、統合ポイントとして Windows ウェルカム センターを使用してください。

ほとんどのシステムレベル機能には、それほど明確な、または直接的な統合ポイントがないため、コントロールパネルアイテムが必要となります。ただし、コントロールパネルを、すべての構成設定を "1 か所" で調整できる場所と見なさないようにします。ユーザー インターフェイスを持つアプリケーションでは、コントロールパネルアイテムを使用する代わりに、UI から直接構成する必要があります。

間違った例:



この例では Windows Internet Explorer® をコントロールパネル内に表示していますが、このようにしないことをお勧めします。Windows Internet Explorer 固有の UI の方が、統合ポイントとして適しています。

コントロールパネルアイテムを新しく作成するか、既存のものを拡張するかを選択する際の判断基準

以下の点に基づいて判断します。

- その機能を、既存の拡張可能なコントロールパネルアイテムにプラグインとして組み込み可能なタスクとして示すことができるかどうか。拡張可能なコントロールパネルアイテムには、Bluetooth デバイス、ディスプレイ、インターネット、キーボード、マウス、ネットワーク、電源、システム、ワイヤレス(赤外線)があります。
- 既存の拡張可能なコントロールパネルアイテムの機能を、プロパティとタスクで代用できるかどうか。該当する場合は、ユーザー エクスペリエンスを単純化できるので、既存のコントロールパネルアイテムを拡張します。該当しない場合は、新しいコントロールパネルアイテムを作成してください。

## デザイン コンセプト

コントロールパネルのコンセプトは、現実世界を参考にして作成されています。現実の世界でデバイスの監視と制御に使用されるコントロールパネルは、つまみ、スイッチ、計器、ディスプレイといった複数の制御装置から成っています。ほとんどの場合、ユーザーがこれらのコントロールパネルを使用するには、使い方を習得するためのトレーニングが必要です。

現実世界の制御装置とは異なり、Windows のコントロールパネルのデザインは、使い慣れていないユーザー向けに最適化されています。ほとんどのコントロールパネルタスクは実行頻度が低いため、通常、実行方法を記憶していることは少なく、毎回効率的に実行方法を思い出すことができる必要があります。

使いやすく便利なコントロールパネルアイテムをデザインするには、次の点に注意します。

- プロパティの必要性を確認します。
- テクノロジーではなく、ユーザーの目的という観点からプロパティを表示します。
- 適切なレベルのプロパティを表示します。
- 特定のタスク向けのページをデザインします。
- 標準ユーザーと保護された管理者向けのページをデザインします。

コントロールパネルに表示するアイテムをデザインおよび評価する際には、ユーザーが実行する一般的なタスクを特定し、それらのタスクを実行するための明確なプロセスが用意されていることを確認してください。コントロールパネルアイテムを使用して一般的に実行されるタスクには、次の種類があります。

- 初期構成
- 低頻度の変更 (ほとんどの設定)
- 高頻度の変更 (少数の重要な設定)
- 初期状態または以前の状態への設定のロールバック
- トラブルシューティング

#### 最も重要な点

コントロールパネルのページを具体的なタスク向けにデザインし、テクノロジーではなくユーザーの目的という観点から提示します。

### 使用パターン

コントロールパネルアイテムに対しては、タスクフローまたはプロパティシートを使用できます。これらの使用パターンを以下に示します。

#### タスクフローのパターン

タスクフロー アイテムの場合、高度なレベルの選択肢を提示するには "ハブページ" を使用し、実際に構成を実行するには "スポークページ" を使用します。

#### ハブページ

- **タスクベースのハブページ:** 最も一般的に使用されるタスクを表示するハブページです。少数の一般的に使用されるタスクや重要なタスクに対して、比較的詳細な手順および説明が必要な場合は、ハブページを使用するのが最も適しています。ハブページには、コミットボタンを配置しません。また、"ハイブリッドのタスクベースハブページ" は、ハブページ上にいくつかのプロパティまたはコマンドが直接配置されています。ユーザーがコントロールパネルを使用してプロパティやコマンドにアクセスする可能性が高い場合は、ハイブリッドハブページを使用することを強くお勧めします。
- **オブジェクトベースのハブページ:** 使用可能なオブジェクトを表示するハブページです。オブジェクトが複数ある場合に最も適しています。ハブページには、コミットボタンを配置しません。このパターンには 2 つのバリエーションがあります。
  - **標準バージョンの実装:** 標準リストビュー コントロールが使用されます。
  - **"ハイブリッド" バージョン:** オブジェクトベースのハブページで、ハブページ上にいくつかのプロパティまたはコマンドが直接配置されています。ユーザーがコントロールパネルを使用してプロパティやコマンドにアクセスする可能性が高い場合は、ハイブリッドハブページを使用することを強くお勧めします。

#### スポークページ

- **タスクページ:** タスクまたは手順と、具体的なタスクベースのメイン指示テキストを提示するスポークページです。詳細な手順や説明を追加することでメリットの得られるようなタスクに最も適しています。
- **フォームページ:** 全般的なメイン指示テキストと、これに関連するプロパティやタスクの集まりを提示するスポークページです。フォームページは、多数のプロパティを持ち、これらを 1 ページに直接表示すると便利な機能に最も適しています。たとえば、詳細プロパティなどがあります。

#### プロパティシートのパターン

- **プロパティシート:** 詳しい知識のあるユーザーを対象とした、設定が多く、従来から存在するアイテムに最も適しています。新しいアイテムの場合は、フォームページのパターンを使ったタスクフローを使用すると同様の効果が得られます。

詳細と例については、「[コントロールパネルの使用パターン](#)」を参照してください。

### ガイドライン

#### プロパティシートのコントロールパネルアイテム

- 新規のコントロールパネルアイテムに対しては、プロパティシートを使用しません。代わりに、タスクフローを使用してシームレスなエクスペリエンスを実現し、コントロールパネルのホームページのカテゴリー分類および検索機能を十分に活用します。

#### タスクフローのコントロールパネルアイテム

#### 全般

- 最も重要なコンテンツやコントロールは、スクロールしなくても表示されるようにします。ユーザーは理由がない限り、ページコンテンツを表示するためにスクロール操作をしません。コミットボタンは、[コンテンツ領域](#)ではなく[コマンド領域](#)に配置することで、常に表示しておくことができます。スクロールを回避するためだけに、ページを分割しないでください。
  - 長いページは上下にスクロールできます。ただし、スクロールしなくとも最も重要なコンテンツが表示できる場合に限ります。
  - 水平スクロールは使用しません。代わりに、ページコンテンツをデザインし直し、垂直スクロールを使用してください。ページの幅が非常に狭い場合に限り、水平スクロールバーを表示するようにします。
- ページ間の移動には次の方法を使用します。
  - タスクを開始するには、[タスク リンク](#)を使用する。
  - 複数のステップから成るタスクで次のページに移動するには、タスク リンクまたは[\[次へ\]](#)ボタンを使用する。
  - タスクを完了するには、[コミット ボタン](#)を使用する。
  - 前に表示したページに戻るには、メニュー バーの[\[戻る\]](#)ボタンを使用する。[\[戻る\]](#)ボタンをコマンド エリアに追加しないでください。
  - コントロール パネルのホーム ページに直接戻るには、アドレス バーを使用する。
  - 他のコントロール パネル アイテムのページに移動するには、作業 ウィンドウの[\[関連項目\]](#)リンクを使用する。これ以外の方法の移動可能範囲は、1つのコントロール パネル アイテム内に限定する必要があります。
- アドレス バーには、コントロール パネルのホーム ページのみを配置します。このリンクをクリックすると、コントロール パネルのホーム ページに戻り、進行中の作業は[確認](#)されずに破棄されます。
- コントロール パネル ページには、[\[閉じる\]](#)コマンド ボタンを配置しないでください。コントロール パネルのウィンドウを閉じるには、タイトル バーの[\[閉じる\]](#)ボタンを使用します。

## タスク リンクとボタン

- ページ内の固定オプションが少ない場合は、ラジオ ボタンと[\[次へ\]](#)ボタンの組み合わせの代わりに、タスク リンクを使用します。これにより、ユーザーは1回のクリックで応答を選択できます。
- タスク リンクおよびボタンは、(見つけやすいように)次の場所に配置できます。
  - [コマンド領域](#)(スポーツ ページ上のコマンド ボタンのみ)。
  - [コンテンツ領域](#)(次のものを配置できます)。
    - コマンド ボタン
    - タスク リンク
    - その他のリンク
  - [作業 ウィンドウ](#)内のリンク(ハブ ページのみ)。
- タスク リンクおよびボタンは、見つけやすくするために、重要性や必要性に基づいて位置を決めます。
  - コマンド エリアには、コミット ボタンのみを配置します。
  - 重要なタスクはコンテンツ エリアに配置します。コマンド ボタンには最も注意が向けられます。したがって、ユーザーに提示する必要のあるコマンドのために使用します。タスク リンクにも注意が向けられますが、コマンド ボタンほどではありません。
  - (重要性の低い)サブタスクに対しては、作業 ウィンドウおよびプレーン リンクを使用します。作業 ウィンドウは、タスク ページ内で最も注目されない領域です。プレーン リンクは、コマンド ボタンやタスク リンクほど目立ちません。
- コンテンツ エリア内のタスク リンクについては、次のようにします。
  - リンクが7つを超える場合は、リンクをいくつかのカテゴリーに分類します。グループごとに見出しを付けてください。
  - リンクが7つ未満の場合は、見出しを付けず、1つのグループ内にリンクを提示します。
- タスク リンクおよびボタンは、論理的な順序で提示します。タスク リンクは上下、コマンド ボタンは水平に並べて配置します。
- カテゴリー内では、コマンドを関連するグループに分けます。最も一般的に使用されるタスク グループを先頭に配置し、各グループ内では最も一般的に使用されるタスクを先頭に配置します。主に使用される頻度順に従って並べますが、論理的にも自然になるようにします。
  - 例外: "すべて実行する"ためのタスク リンクは先頭に置きます。
- タスク リンクが多い場合は、最も重要なタスクを目立つ外観にします。24×24ピクセルのアイコンと2行のテキストを使用すると目立ちます。重要性の低いタスクについては、16×16ピクセルのアイコンを使用するかアイコンを使用せず、1行のリンク テキストを使用します。

#### Parental controls:

On, enforce current settings

Off

#### Activity reporting:

On, collect information about computer usage

Off

#### Windows settings



##### Windows Vista Web Filter

Control allowed websites, downloads, and other use



##### Time limits

Control when Jonathan uses the computer



##### Games

Control games by rating, content, or title



##### Allow and block specific programs

Allow and block any programs on your computer

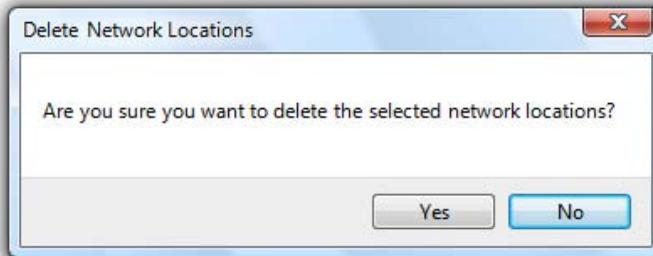
この例では、重要なコマンドの外観を特に目立つようにしています。

- 頻繁に使用されるコマンドとリスクのあるコマンドを、物理的にはっきり見分けられるようにします。そうでない場合、リスクのあるコマンドが誤ってクリックされてしまうおそれがあります。リスクのあるコマンドは、何らかの方法でまとめて整列し直す必要があります。
- コマンドの処理を元に戻すメカニズムをページ上に用意します。ユーザーが誤りを元に戻すために、別の場所に移動する必要がないようにします。
- タスク リンクのアイコンは、すべて既定のタスク リンクアイコンを使用するか、すべてカスタム アイコンを使用します。これらを混在させないでください。次の場合にのみ、カスタム アイコンの使用を検討します。
  - タスクの把握に役立つ。
  - Aero アイコン標準に準拠している。
  - 目立たない外観である。

#### ダイアログ ボックス

タスク フローを使用する場合、一般的に 1 つのウィンドウ内でページからページへタスクが進行するようになりますが、次のような場合は例外です。

- 次のような状況では、ダイアログ ボックスを使用します。
  - 管理者のユーザー名およびパスワードの入力をユーザーに求めるとき。この目的では、常に、資格情報マネージャー ダイアログ ボックスを使用します(必ずモーダルにします)。
  - タスク ダイアログ ボックスまたはメッセージ ボックスを使用して、インプレース コマンドを確認するとき(必ずモーダルにします)。
  - インプレース コマンド([新規]、[追加]、[名前を付けて保存]、[名前の変更]、[印刷]などのコマンド)の入力を取得するとき。



この例では、[削除] コマンドを、別のページではなくダイアログ ボックスで実行しています。

- 独立したサブタスクを実行するとき。モードレスのサブ ウィンドウを使用することにより、これらのタスクを、主要なタスク フローの外部で個別に実行できます。

#### ハブ ページ

#### 全般

- 次の場合は、タスクベースのハブ ページを使用します。
  - 一般的に使用されるタスクまたは重要なタスクが少数の場合。
  - 関連するオブジェクトが 1 つまたは 2 つの構成である場合(モニター、キーボード、マウス、ゲーム コントローラーなど)。

- ・システム全体に適用される構成の場合(日時、セキュリティ、電源オプションなど)。
- ・次の場合は、オブジェクトベースのハブページを使用します。
  - ・関連するオブジェクトが複数の構成である場合(ユーザー アカウント、ネットワーク接続、プリンターなど)。
  - ・選択されたオブジェクトのみに適用される構成の場合。
- ・コントロール パネル アイテムが1ページで構成される場合は、ハブページを使用しません(つまり、1ページにすべての関連タスクおよびプロパティが含まれる場合)。

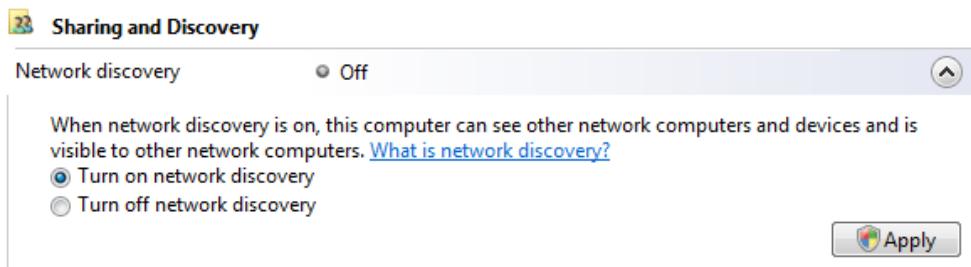
## オブジェクトリスト

- ・アイテムを論理的な順序で一覧表示します。名前付きオブジェクトはアルファベットまたは五十音順、数字は番号順、日付は時系列に基づいて並べ替えます。
- ・オブジェクトベースのハブの場合、表示を変更する機能が重要なタスクでは、作業ウィンドウ内でオブジェクト表示コマンドを提供します。オブジェクトが多数あり、既定の表示がすべてのシナリオで効果的とはいえない場合は、表示を変更する機能が重要となります。リストビューのコンテキストメニューを使用すれば、明示的なコマンドがなくてもリストビューの変更は可能ですが、これらのメニューは見つけやすいとはいません。

オブジェクトリストの表示に関する詳細なガイドラインについては、「[リストビュー](#)」を参照してください。

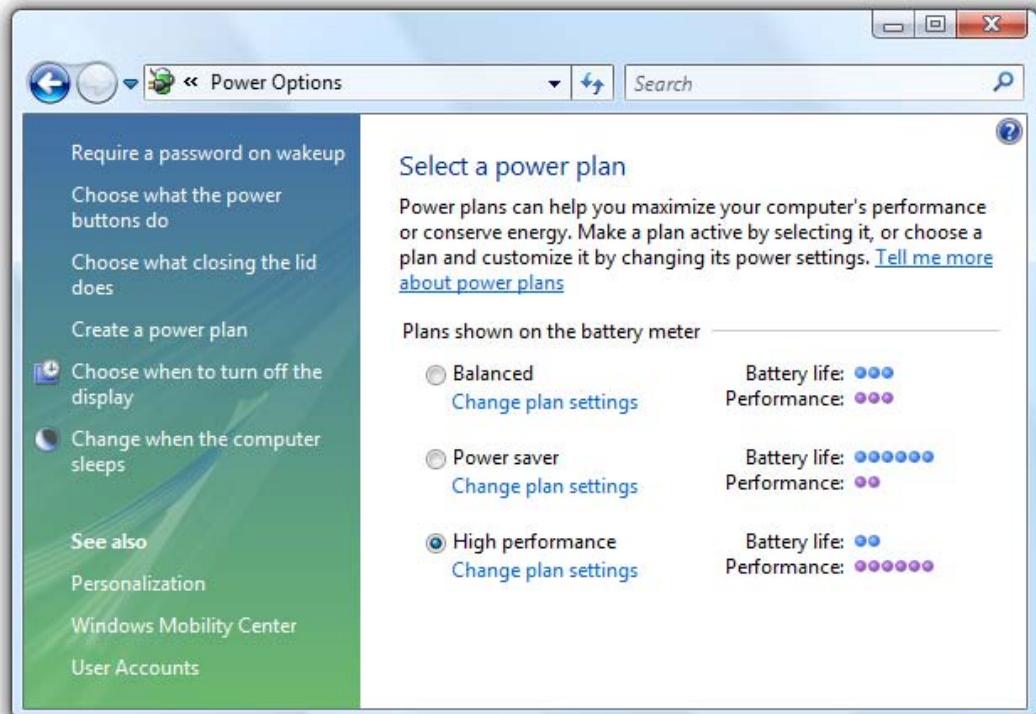
## 対話操作

- ・コミット ボタンは、ハブページには配置しません。ハブページは基本的に開始ポイントです。ユーザーがハブページで操作を完了することはないため、"コミット"することはあり得ません。ハブページにコミット ボタンを配置すると、ハブから開始されたタスクと紛らわしくなります(ユーザーは、これらのタスクをコミットする必要があるかどうかの判断に迷います)。
  - ・例外: 設定の変更に昇格が必要な場合は、[適用] ボタンとセキュリティ シールド アイコンを使用します。変更が適用された場合は、コミット ボタンを無効にしてください。



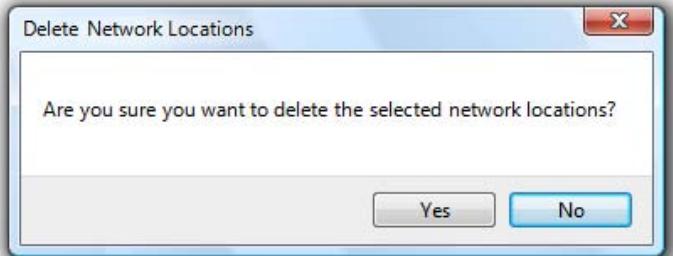
この例では、設定の変更に昇格が必要なため、[適用] ボタンが使用されています。

- ・最もよく使用されるプロパティをハブページ上に直接配置することを検討します。ユーザーがコントロールパネルを使用してこれらのプロパティにアクセスする可能性が高い場合は、ハイブリッドハブページを使用することを強くお勧めします。



この例の [電源オプション] コントロール パネル アイテムでは、最もよく使用される設定がハブページ上に直接配置されています。

- ハイブリッド ハブ ページ上のどの設定も変更が即座に実行されるように、[即時型のコミット モデル](#)を使用します。繰り返しになりますが、ユーザーがハブ ページをコミットすることはあり得ません。コミット ボタンを必要とする設定は、ハブ ページに配置しないでください。
- ナビゲーション リンクを使用する代わりに、シンプルな "ワンステップ" コマンドをハブ ページ上に直接配置することを検討してください。
- 簡単に元に戻すことができないインプレース コマンドについては確認を行います。[タスク ダイアログ ボックス](#)または[メッセージ ボックス](#)を使用してください。



この例では、ダイアログ ボックスで [削除] コマンドを確認しています。

- タスクベースのハブ ページでは、タスク リンクとアイコンを使用して各タスクを特定します。また、オプションで各リンクに説明を追加することもできます。ただし、タスク リンクだけで内容がわかるように心掛け、本当に必要なリンクに限り説明を追加するようしてください。

#### Windows settings

- [Windows Vista Web Filter](#)  
Control allowed websites, downloads, and other use
- [Time limits](#)  
Control when cc uses the computer
- [Games](#)  
Control games by rating, content, or title
- [Allow and block specific programs](#)  
Allow and block any programs on your computer

この例では、各タスクに対応するタスク リンクおよびアイコンがあります。

- オブジェクトベースのハブ ページでは、オブジェクトをシングルクリックすると選択できます。また、ダブルクリックするとオブジェクトが選択され、さらに既定のページに移動できます。既定のページは、通常、プロパティ ページまたはタスクベースのハブ ページです。
- オブジェクトベースのハブ ページでは、選択されたオブジェクト用のタスクベースのハブに移動することもできます。ただし、コントロール パネル アイテムが間接的であるような印象が強くなるため、このような補助的なハブは避ける必要があります。

#### 作業 ウィンドウ

コマンド、表示、および関連するコントロール パネル アイテムへのリンクは、作業 ウィンドウに表示します。

- タスクベース ハブの作業 ウィンドウでは、次の順序でリンクを表示します。
  - 副コマンド。主要なタスクは必ずコンテンツ エリアに配置します。作業 ウィンドウには、副次的な、オプションタスクを表示します。重要なシナリオにおいてユーザーが見つける必要があるタスクは主要なタスクです。一方、ユーザーが見つけられなくても問題のないタスクは副次的なタスクです。
  - 関連項目。関連するコントロール パネル アイテムに移動するオプションのリンク。
- オブジェクトベース ハブの作業 ウィンドウでは、次の順序でリンクを表示します。
  - オブジェクト ビュー。オブジェクトの表示を制御するオプションのリンク。
  - 固定コマンド。現在選択されているオブジェクトに依存しないコマンド。
  - コンテキスト依存コマンド。現在選択されているオブジェクトに依存し、常に表示されるとは限らないコマンド。
  - 関連項目。関連するコントロール パネル アイテムに移動するオプションのリンク。
- スパーク ページでは、作業 ウィンドウを使用しません。ハブ ページと異なり、スパーク ページはタスクの完了に重点を置く必要があります。完了する前に、ユーザーに移動を促さないようにしてください。

#### [関連項目] リンク

- 作業 ウィンドウ内に [関連項目] リンクを設定して、関連するコントロール パネル アイテムをユーザーが簡単に見つけられるようにします。また、ユーザーが不適切なコントロール パネル アイテムを選択した場合に、適切なコントロール パネル アイテムを見つけることも容易になります。ユーザーがコントロール パネル アイテムと関連付けて考える可能性の高いアイテムのリンクを表示します。



この例では、[パフォーマンス] コントロール パネル アイテムに、[セキュリティ センター] および [問題のレポートと解決策]へのリンクが表示されています。ユーザーがこれらのコントロール パネル アイテムからタスクを実行する可能性が高いです。

- 特定のタスク ページをユーザーが利用する可能性が高い場合は、そのページへのリンクを表示します。それ以外の場合は、すべてをコントロール パネル アイテムにします。“コントロール パネル”という語句を使用せず、コントロール パネル名のみを使用してください。

## スポーク ページ

### 全般

- 一般的に使用されるタスクや重要なタスクに対して、比較的詳細な手順および説明が必要な場合は、タスク ページを使用します。
- 多数の設定があり、これらを 1 ページに直接表示すると便利な機能には、フォーム ページを使用します。一般的に、少数のシンプルなプロパティを明確に変更するようなタスクが、フォーム ページに最適なタスクです。
- スポーク ページでは、作業 ウィンドウを使用しません。

### 対話操作

- 主要タスクは 1 ページに収めるようにします。必要なページ数が 2 ページ以上に及ぶ場合は、次の対処を行ってください。
  - 補助的な手順またはオプションの手順に対して、中間スポーク ページを使用します。中間スポーク ページは、最終スポーク ページによってコミットされます。
  - 独立した補助タスクに対しては、独立したウィンドウを使用します。独立したウィンドウは、主要タスクとは無関係に単独でコミットされます。

上記の対処を行うと、主要タスクに対してコミット ボタンを使用する意味が明確になります。ユーザーがコミット 対象を必ず理解している状態にする必要があります。

- タスク フローでは、[関連項目] リンクを使用しません。これらのリンクは、関連する別のコントロール パネル アイテムへのリンクです。ハブ ページから別のアイテムへ移動することはできますが、スポーク ページから別のアイテムへ移動するとタスクが中断されてしまうため、不可能です。
- スポーク ページは、単純な入力または確認に使用しないでください。代わりに、モーダル ダイアログ ボックスを使用します。

### 対話操作 (中間スポーク ページ)

- 次のページに移動するには、タスク リンクまたは [次へ] ボタンを使用します。次の手順に進む方法は常に明確にする必要があります。
- オプションのタスク ステップへのナビゲーション リンクを用意することもできます。タスクをコミットする必要があるかどうかの判断にユーザーが迷うことがないように、これらのページは、同一コントロール パネル アイテム内の中間スポーク ページにする必要があります。中間スポーク ページにはコミット ボタンを配置せず、主要タスクがコミットされたときにコミットする必要があります。

### 対話操作 (最終スポーク ページ)

- タスクを完了するには、コミット ボタンを使用します。スポーク ページでは、明示的にコミットされるまで変更が実行されないように、[遅延型のコミット モデル](#)を使用します(ユーザーが [前へ]、[閉じる]、またはアドレスバーを使用してページから移動した場合は、変更が破棄されます)。コミット ボタンは、タスクを完了しようとしていることを示す、視覚的な手掛かりとなります。この用途でリンクを使用しないでください。
- コミット ボタンでは、[キャンセル] を含め、確認を行いません。確認すると、わずらわしくなります。次の場合は例外です。
  - 操作が重大な結果をもたらし、誤りがあつても簡単に修正できない。
  - 操作によって、時間または労力に関する重大な損失をユーザーに与える可能性がある。
  - その操作が明らかに他の操作と矛盾している。
- 変更を破棄してよいかどうかをユーザーに確認しないでください([戻る]、[閉じる]、またはアドレスバーを使って移動した場合)。ただし、意図せずに移動した場合に時間または労力に関する重大な損失をユーザーに与える可能性があるときには、確認してもかまいません。
- コマンドやナビゲーション リンクは使用しません ([関連項目] リンクを含む)。最終スポーク ページでは、ユーザーは明示的にタスクを完了するかキャンセルする必要があります。他の場所への移動をユーザーに促さないようになります。移動すると、タスクが暗黙のうちにキャンセルされるおそれがあります。
- ユーザーがタスクを完了またはキャンセルした場合は、タスクが開始されたハブ ページに戻る必要があります。該当するページがない場合は、代わりにコントロール パネル ウィンドウを閉じます。スポーク ページが常に別のページから開始されるとは限りません。

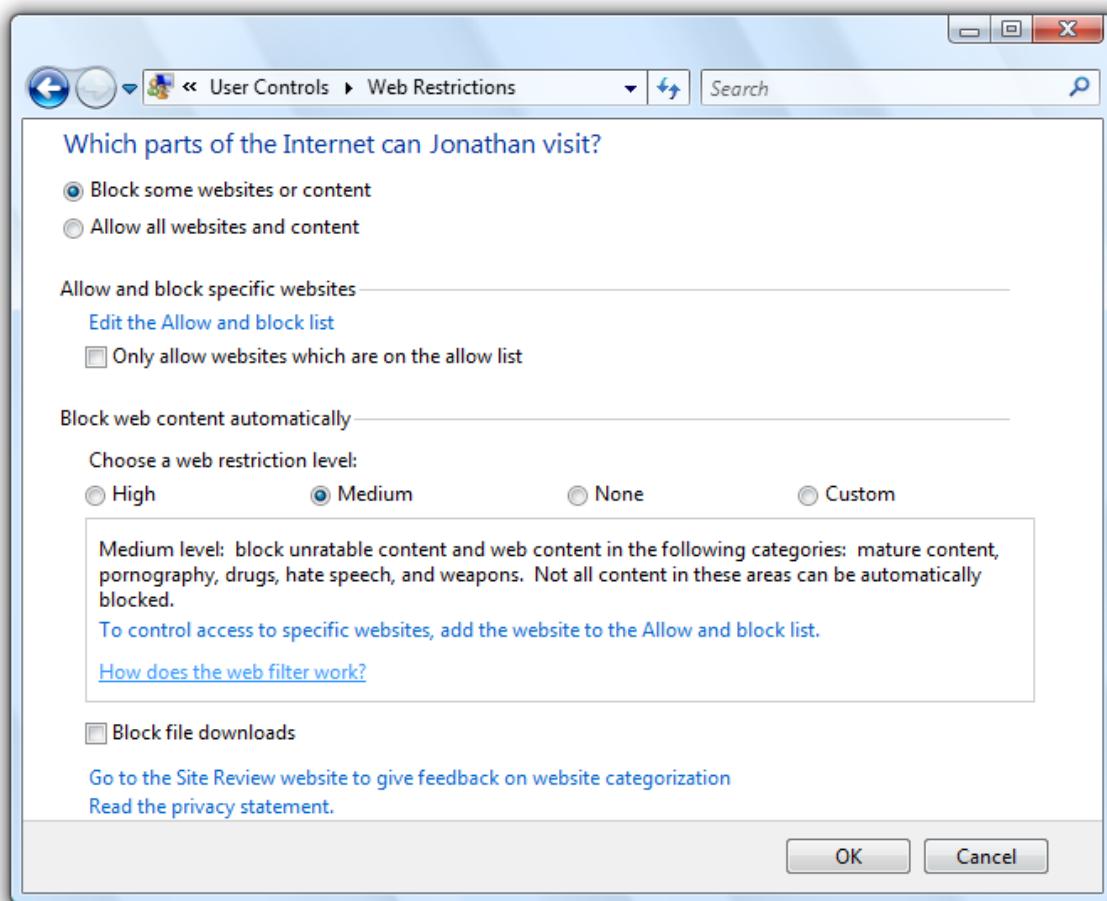
せん。

- 不要になった "コミット済み" ページは Windows エクスプローラーのバック スタックから削除します (タスクが開始されたページに戻る場合)。[戻る] ボタンがクリックされた場合に、ユーザーが既にコミットしたページは表示しません。追加の変更を加える場合は、[戻る] をクリックして以前のページを修正するのではなく、タスクを完全にやり直す必要があります。
  - 開発者向け情報: これらの不要なページを削除するには、API の `ITravelLog::FindTravelEntry()` および `ITravelLogEx::DeleteEntry()` を使用します。

## コミット ボタン

注: [キャンセル] ボタンはコミット ボタンと見なされます。

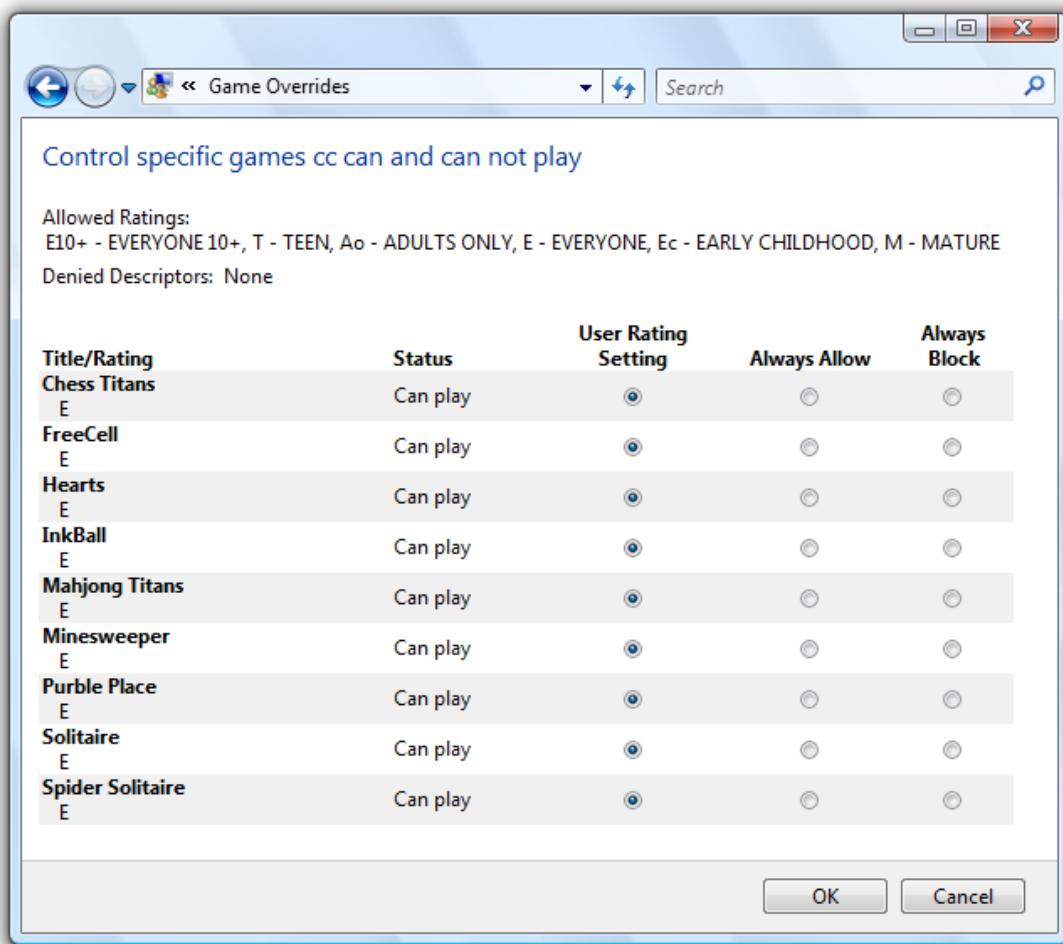
- タスクを確認するには、[OK] などの汎用的なラベルではなく、メイン指示テキストに対する具体的な応答を示したコミット ボタンを使用します。コミット ボタンのラベルは、単独で意味のわかる名前にする必要があります。[OK] を使用しないようしてください。[OK] はメイン指示テキストに対する具体的な応答でないため、別の意味に解釈されるおそれがあります。また、[OK] は一般的にモーダル ダイアログ ボックスで使用されるため、コントロール パネル アイテムのウィンドウを閉じるという誤った意味で理解される可能性もあります。
  - 次の場合は例外です。
    - 設定を行わないページでは、[OK] を使用します。
    - 具体的な設定/設定群を変更する場合で、[保存] や [選択] など、具体的な応答が汎用的な場合は、[OK] を使用します。
    - メイン指示テキストに対する応答がラジオ ボタンで表示されているページでは、[OK] を使用します。遅延型のコミット モデルを使用するため、最終スポーツ ページ上ではタスク リンクを使用できません。



この例では、コミット ボタンではなくラジオ ボタンが、メイン指示テキストに対する応答に使用されています。

- ユーザーが明示的に変更を破棄できるように [キャンセル] ボタンを用意します。ユーザーは変更を確認しないことによって暗黙的にタスクを破棄できますが、[キャンセル] ボタンを使えば確実に変更を破棄できます。
  - 例外: ユーザーが変更できないタスクでは、[キャンセル] ボタンは使用しません。この場合、[OK] ボタンをクリックすると、[キャンセル] と同じ結果になります。
- [閉じる]、[完了]、または [終了] コミット ボタンを使用しないでください。これらのボタンは一般的にモーダル ダイアログ ボックスで使用されるため、コントロール パネル アイテムのウィンドウを閉じるという誤った意味で理解される可能性があります。ウィンドウを閉じるには、タイトルバーの [閉じる] ボタンを使用します。また、[完了] と [終了] は誤解を招く可能性があります。ユーザーはタスクが開始されたページに戻るので、実際には完了できません。
- コミット ボタンは無効にしません。無効にすると、ユーザーはコミット ボタンが無効化されている理由を推測しなければならなくなります。コミット ボタンを有効にしておき、問題が発生したときに有用なエラー メッセージを表示することをお勧めします。

- スクロールしなくとも、ページ上にコミット ボタンが必ず表示されるようにします。ページが長い場合は、コミット ボタンをコンテンツ領域ではなくコマンド領域に配置することで、常に表示しておくことができます。



この例のコンテンツ エリアのサイズは無制限です。そのため、コミット ボタンはコマンド エリアに配置されています。

- コミット ボタンは右揃えにし、左から肯定的なコミット ボタン、[キャンセル]、[適用] の順に配置します。

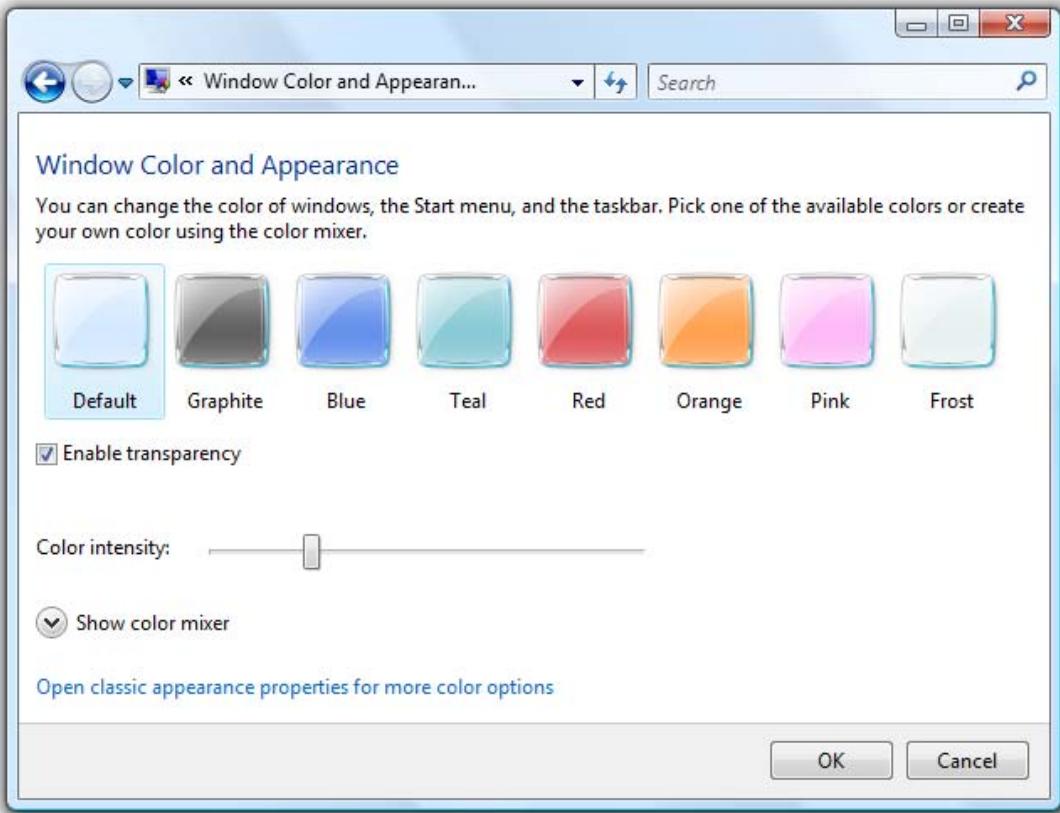
#### [プレビュー] ボタン

- [プレビュー] ボタンにより、保留中の変更を即座に確認できます。これにより、変更をコミットせずにページから移動することで現在の設定を復元できます。
- [プレビュー] ボタンは、どのスパーク ページ上でも使用できます。ハブ ページでは、[即時型のコミット モデル](#)が使用されているため、[プレビュー] ボタンは不要です。
- コントロール パネル ページには、[適用] ボタンではなく [プレビュー] ボタンを使用することを検討してください。[プレビュー] ボタンは理解しやすいと、どのスパーク ページ上でも使用できます。
- [プレビュー] ボタンは、ユーザーが効果を確認できる設定がページに(1つ以上)ある場合にのみ配置します。ユーザーが変更をプレビューし、その変更を評価して、その評価を基にさらに変更を行えるようにしてください。
- [プレビュー] ボタンは常に有効にしておきます。

#### リアルタイムのプレビュー

スパーク ページに対する変更の影響を即座に確認できる場合は、コントロール パネル アイテムにリアルタイムのプレビューを表示します。

- 次のような場合は、表示設定にリアルタイムのプレビューの使用を検討してください。
  - 通常はモニター全体に適用されるので、効果が明確である。
  - 大幅に遅延することなく効果を適用できる。
  - 結果が安全で、簡単に元に戻すことができる。



この例では、[ウィンドウの色とデザイン]設定の効果を直ちに確認できます。これにより、ユーザーが変更のために実行する作業を最小限に抑えることができます。

- ・コミットボタンには、[変更の保存]および[キャンセル]を使用します。[変更の保存]では現在の設定が保存され、[キャンセル]では元の設定に戻ります。[OK]の代わりに[変更の保存]を使用すると、プレビューした変更が適用されていないことを明確に示すことができます。
- ・[適用]ボタンは使用しません。リアルタイムのプレビューでは[適用]は不要です。
- ・ユーザーがページから移動した場合は変更を元に戻します([戻る]、[閉じる]、またはアドレスバーを使って移動した場合)。ユーザーが変更を保存するには、変更を明示的にコミットする必要があります。

#### [適用]ボタン

- ・[適用]ボタンは、保留中の変更(タスクが開始された後や、前回[適用]ボタンがクリックされた後に行われた変更)を適用し、現在のページを表示したままにする手段です。こうすることで、ユーザーが他のタスクへ移動する前に変更内容を評価できるようになります。
- ・最終スパークページ上でのみ[適用]ボタンを使用してください。中間スパークページは即時型のコミットモデルを使用するため、中間スパークページ上で[適用]ボタンは使用できません。
  - ・例外:ハイブリッドハブページで、設定の変更に昇格が必要な場合は、[適用]ボタンを使用できます。詳細については、ハブページの「対話操作」を参照してください。
- ・[適用]ボタンは、ユーザーが意味のある方法で効果を評価できる設定がページに(1つ以上)ある場合にのみ配置します。通常、[適用]ボタンは、設定によって目に見える変化が生じる場合に使用します。ユーザーが変更を適用し、その変更を評価して、その評価を基にさらに変更を行えるようにしてください。
- ・保留中の変更がある場合にのみ、[適用]ボタンを有効にします。保留中の変更がない場合は無効にします。
- ・アクセスキーとして"A"を割り当てます。

#### コントロールパネルの統合

コントロールパネルアイテムをWindowsに統合するには、次のことを行います。

- ・コントロールパネルアイテム(アイテムの名前、説明、アイコンを含む)を登録します。これにより、Windowsにコントロールパネルアイテムが認識されます。
- ・コントロールパネルアイテムが最上位(後述)の場合は、
  - ・コントロールパネルアイテムを適切な"カテゴリー ページ"に関連付けます。
  - ・タスクリンク(リンクの名前、説明、キーワード、コマンドラインを含む)を提供します。これにより、主要タスクを指定し、ユーザーがタスクに直接移動できるようになります。
- ・検索語を提供します。これにより、コントロールパネルの検索機能を使用してタスクリンクを検索できるようになります。

この情報は、個々のコントロールパネルアイテムに対してのみ提供できます。既存のコントロールパネルアイテムを拡張する場合、この情報は追加または変更できません。

## カテゴリー ページ

- コントロール パネル アイテムは、次の場合にのみカテゴリー ページに追加します。
  - ほとんどのユーザーが必要とする。例: ネットワークと共有センターなど。
  - 何回も使用される。例: システムなど。
  - 他の場所には表示されない重要な機能を提供している。例: プリンターなど。これらの基準を満たすコントロール パネル アイテムは、"最上位" のアイテムです。
- 次の場合は、コントロール パネル アイテムをカテゴリー ページに追加しません。
  - ほとんど使用されないか、1回限りのセットアップに使用される。例: ウェルカム センターなど。
  - 詳しい知識のあるユーザーまたは IT プロフェッショナルを対象とする。例: カラー調整など。
  - 現在のハードウェアまたはソフトウェア構成に該当しない。例: Windows SideShow (現在のハードウェアでサポートされない場合) など。

これらのコントロール パネル アイテムをカテゴリー ページから削除すると、最上位アイテムを簡単に見つけられるようになります。使用方法を考慮することで、検索またはコンテキストに沿ったエントリ ポイントから確実にこれらのコントロール パネル アイテムを見つけることができます。

- ユーザーが探す可能性の高いカテゴリーに、最上位のコントロール パネル アイテムを関連付けます。必ずユーザー テストに基づいて判断してください。
- ユーザーが 2 番目に探す可能性が高いカテゴリーに、最上位のコントロール パネル アイテムを関連付けることも検討します。ユーザーが主要タスクを探す可能性の高いカテゴリーが複数ある場合は、コントロール パネル アイテムを 2 つのカテゴリーに関連付ける必要があります。
- コントロール パネル アイテムは、3 つ以上のカテゴリーに関連付けしません。コントロール パネル アイテムが複数のカテゴリーに表示されると、カテゴリーに分類する意味がなくなります。

## タスク リンク

- コントロール パネル アイテムを主要タスクに関連付けます。カテゴリー ページに表示できるタスク数は最大で 5 つですが、コントロール パネル の検索にはタスクを追加できます。タスク リンクと同じ表現を使用します。単語をいくつか削除してタスク リンクを簡潔にする場合もあります。
- コントロール パネル アイテムの個々の場所へのタスク リンクを作成することを推奨します。同じ場所に複数のリンクを関連付けると、混乱を招くおそれがあります。

## 検索語

- ユーザーがコントロール パネル アイテムを検索するときに使用する可能性の高い検索語を登録します。次のような検索語が考えられます。
  - 構成対象の機能やオブジェクト。
  - 主要タスク。必ずユーザー テストに基づいて、検索語を設定してください。
- これらの検索語には、一般的な類義語も登録します。たとえば、"モニター" と "ディスプレイ" は類義語なので、両方の単語を登録する必要があります。
- つづり字異形や単語区切りの有無を考慮します。たとえば、ユーザーは "Web サイト" と "Web サイト" のいずれかを検索する可能性があります。一般的なスペル ミスも考慮する必要があります。
- 名詞の単数形および複数形を考慮します。コントロール パネル の検索機能では、単数形と複数形の両方を自動的に検索できません。ユーザーが検索に使用する可能性が高い形を指定してください。
- 単純現在時制の動詞を使用します。"connect (接続する)" を検索語として登録した場合、検索を実行しても "connects (接続する)"、"connecting (接続している)"、および "connected (接続されている)" は自動的には検索されません。
- 大文字と小文字の区別を考慮する必要はありません。この検索機能では、大文字と小文字は区別されません。

## 標準ユーザーおよび保護された管理者

変更するうえで管理者特権が必要な設定は、数多くあります。処理に管理者特権が必要な場合、Windows Vista 以降では、[標準ユーザー](#) と [保護された管理者](#) は各自の特権を明示的に昇格する必要があります。これにより、危険なコードが管理者特権で実行されるのを防ぐことができます。

詳細と例については、「[ユーザー アカウント制御](#)」を参照してください。

## デザインとテーマ

"デザイン (scheme)" は、名前が付けられた、表示に関する設定のコレクションです。"テーマ" は、名前が付けられた、システム全体の設定のコレクションです。デザインやテーマの例としては、ディスプレイ、マウス、電話とモデム、電源オプション、サウンドとオーディオのオプションなどがあります。

- 次のような場合は、ユーザーにデザインの作成を許可します。
  - ユーザーが設定を変更する可能性が高い場合。
  - ユーザーが設定をコレクションとして変更する可能性がきわめて高い場合。

デザインは、物理的な場所(国または地域、タイムゾーン)、コンピューターの使用状況(バッテリーで使用しているか、ドッキングしているか、ドッキング解除しているか)、コンピューターの使用目的(プレゼンテーション、ビデオ再生)などが異なる、さまざまな環境にユーザーがいる場合に便利です。

- 既定のデザインを少なくとも1つ用意します。既定のデザインには適切な名前を指定し、ほとんどの状況で大多数のユーザーに適用されるものにします。ユーザーは独自のデザインを作成する必要がありません。
- プレビューできるようにします。これにより、ユーザーがデザインの状況を確認できるようにします(プレビュー以外のメカニズムでもかまいません)。



この例の「テーマの設定」コントロールパネルアイテムでは、デスクトップやさまざまな表示要素の設定のプレビューが表示されています。

- [名前を付けて保存]および[削除]コマンドを用意します。名前を変更するコマンドは不要です。デザインの名前の変更は、別の名前でデザインを保存し、元のデザインを削除することで実行できます。
- デザインがないと適用できない設定がある場合は、すべてのデザインを削除できるようにしないでください。ユーザーは独自のデザインを作成する必要がありません。
- デザインが完全に独立したものでなく、他の影響を受ける場合(たとえば、電源設定はポータブルコンピューターの現在の動作モードに応じて変化します)、すべてのデザインに共通する設定について簡単に変更できる方法を用意します。たとえば、電源設定と同じ場所で、ポータブルコンピューターの蓋を閉じた場合の処理を設定できるようにします。

## その他

- Windowsの既存の機能を変更または拡張する場合は、コントロールパネルの拡機能張を使用します。拡張可能なコントロールパネルアイテムには、Bluetoothデバイス、ディスプレイ、インターネット、キーボード、マウス、ネットワーク、電源、システム、ワイヤレス(赤外線)があります。

## 既定値

- コントロールパネルアイテム内の設定は、機能の現在の状態を反映している必要があります。そうしないと、誤解を招き、好ましくない結果が生じる可能性があります。たとえば、設定が現在の状態ではなく推奨事項を反映している場合、ユーザーは変更不要と見なし、変更作業を行わずに[キャンセル]をクリックする可能性があります。
- 最も安全(データの消失やシステムアクセスが失われることを防ぐため)で、最もセキュリティの高い初期状態を選択します。ほとんどのユーザーが設定を変更しないことを前提としてください。
- 安全性とセキュリティを判断材料として考える必要がない場合は、最もよく使用される初期状態か、最も便利な初期状態を選択します。

## テキスト

### アイテム名

- コントロールパネルアイテムの処理の内容と他のアイテムとの違いがはっきりわかる、説明的な名前にします。ほとんどの場合、構成対象のWindows機能またはオブジェクトの名前にします。この名前は、コントロールパネルのホームページのクラシックビューに表示されます。

- 名前には "設定"、"オプション"、"プロパティ"、"構成"などの語を使用しないでください。これらの語を明示せず、省略することで、ユーザーの視認性が向上します。

間違った例:

ユーザー補助のオプション  
モデム設定  
電源オプション  
地域と言語のオプション

正しい例:

ユーザー補助  
モデム  
電源  
地域の形式と言語

正しい例では、不要な語が削除されています。

- コントロールパネルアイテムが関連する複数の機能で構成される場合は、アイテムの識別に必要な機能のみを示すか、最もよく使用される機能またはユーザーが最初に使用する機能を示します。

間違った例:

フォルダー オプション  
電話とモデムのオプション

正しい例:

ファイルとフォルダー  
モデム

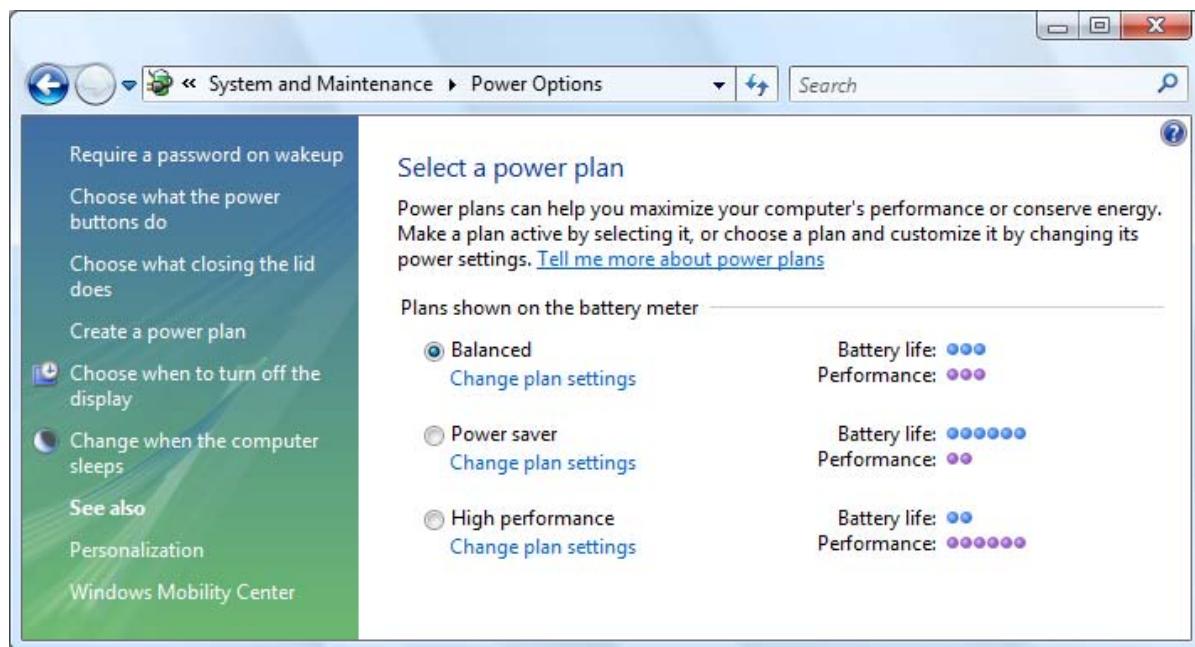
正しい例では、コントロールパネルアイテムの主要な機能が強調されています。

- タイトルスタイルの大文字化**を使用します。

#### ページ タイトル

注: すべてのエクスプローラー ウィンドウと同様、コントロールパネルのページ タイトルは[アドレスバー](#)に表示され、タイトルバーには表示されません。

- ハブ ページには、コントロールパネルアイテム名を使用します。
- スポーク ページには、ページの目的の要約を使用します。ページのメイン指示テキストが簡潔な場合はそのまま使用し、簡潔でない場合はメイン指示テキストを簡潔に修正したものを使用します。



この例では、メイン指示テキストでなく "電源オプション" がページ タイトルに使用されています。

- タイトルスタイルの大文字化**を使用します。

#### タスク リンク テキスト

タスク ページへのリンク (カテゴリーページのタスク リンクや [関連項目] リンク) には、次のガイドラインが適用されます。

- タスクの意味を明確に表し、判別できる簡潔なタスク名を選択します。タスクの本来の意味や他のタスクとの違いをユーザーが考えこむようなものにはしないでください。

間違った例:

## 表示設定の調整

### 正しい例:

画面の解像度

正しい例の表現は、内容をより正確に伝えています。

- タスク リンクとその参照先ページで同じ表現を使用します。リンクをクリックして表示されたページにユーザーが驚くことがないようにしてください。
- タスク ページの場合、メイン指示テキスト、コミット ボタン、タスク リンクのテキストを、関連性の高いものにします。

例:

タスク リンク: ワイヤレス ネットワークへの接続

メイン指示テキスト: 接続するネットワークを選択します

コミット ボタン: 接続

タスク リンク: 保護者による制限のセットアップ

メイン指示テキスト: ユーザーを選択して保護者による制限を設定します

コミット ボタン: 保護者による制限の適用

タスク リンク: 同期の競合の解決

メイン指示テキスト: この競合を解決する方法を指定してください

コミット ボタン: 解決

これらは、タスク リンク テキスト、メイン指示テキスト、コミット ボタン テキストの間の関係を示した例です。

- ほとんどのタスクには動詞が含まれていますが、汎用的な構成に関連する動詞で、タスクの内容がわかりにくいものは、カテゴリーページから除外することを検討してください。

具体的で、わかりやすい語の例は次のとおりです。

Add (追加)

Check (チェック)

Connect (接続)

Copy (コピー)

Create (作成)

Delete (削除)

Disconnect (切断)

Install (インストール)

Remove (削除)

Set up (セットアップ)

Start (開始)

Stop (停止)

Troubleshoot (トラブルシューティング)

汎用的で、わかりにくい語の例は次のとおりです。

Adjust (調整)

Change (変更)

Choose (選択)

Configure (構成)

Edit (編集)

Manage (管理)

Open (開く)

Pick (選択)

Set (設定)

Select (選択)

Show (表示)

View (表示)

- タスクが関連する複数の機能で構成される場合は、代表的な機能のみを示します。簡単に推論できる詳細情報は省略します。

間違った例:

音量、ミュート、音量アイコン

スピーカー、マイク、MIDI、サウンド設定

正しい例:

音量

スピーカーとマイク

正しい例では、代表的な機能のみをタスク リンクで使用しています。

- 対象ユーザーがタスクに対してテクノロジーに関する用語を使用することが一般的である場合にのみ、そのような用語を使用します。

間違った例:

Connectoid  
ピクセル深度  
dpi

正しい例:

プリンター  
スキャナー  
マウス

正しい例は、対象ユーザーが使用する可能性の高いテクノロジーベースの用語です。これに対し、間違った例は、対象ユーザーが使用する可能性が低いものです。

- システムで複数の機能やオブジェクトをサポートする場合のみ、名詞の複数形を使用します(英語の場合)。
- センテンススタイルの大文字化**を使用します。
- 末尾に句読点は付けません。

#### メイン指示テキスト

- ハブページのメイン指示テキストでは、コントロールパネルアイテムの目的を説明します。メイン指示テキストによって、ユーザーは、適切なコントロールパネルアイテムを選択したかどうかを確認できます。ユーザーがコントロールパネルアイテムを間違って選択し、実際に探している設定が別のコントロールパネルアイテムに属している場合もあることを考慮してください。

例:

コンピューターをセキュリティで保護し、最新の状態に保ちます  
表示、再生、制御が簡単になるようにコンピューターを最適化します

- スパークページのメイン指示テキストでは、ページで実行する処理を説明します。指示テキストは、具体的な文、必須の指示、または質問である必要があります。優れた指示テキストとは、処理のしくみにのみ重点を置くものではなく、ページの目的をユーザーに伝えるテキストです。

間違った例:

通知タスクの選択

正しい例:

着信情報の処理方法の指定

正しい例では、ページで実行する内容がより明確に伝わります。

- 可能な限り、具体的な動詞を使用します。ユーザーにとって、具体的な動詞は、汎用的な動詞よりも有益です。
- センテンススタイルの大文字化**を使用します。
- 指示テキストが文の場合、文末の句点は含めません。指示テキストが疑問文の場合は、疑問符を付けます。

#### 補足指示テキスト

- ハブページの補足指示テキストでは、コントロールパネルアイテムの目的を詳細に説明します。
- スパークページの補足指示テキストでは、ページの理解や使用に役立つ追加情報を表示します。詳細情報や用語の定義を提示できます。
- 完全な文にし、**センテンススタイルの大文字化**を使用します。

#### ページ テキスト

- コンテンツエリアで、メイン指示テキストと同じ内容を繰り返さないようにします。
- ページ自体を指す場合は "ページ" という語を使用します。
- ユーザーに指示を伝える場合は、二人称(あなた/あなたの)を使用します(英語の場合)。これは、メイン指示テキストおよびコンテンツエリアで使用します。これ以外のほとんどの場合は、二人称を明示しません。

例:

Choose the pictures you want to print. (印刷する画像を選択します。)

- ユーザーがコントロールパネルアイテムに指示を伝える場合は、一人称("私は/私に/私の")を使用します(英語の場合)。これは、メイン指示テキストに対する応答を示すコンテンツエリアで使用します。

例:

Print the photos on my camera. (カメラの写真を印刷します。)

#### タスク リンク

- タスクリンクの処理の内容と他のタスクリンクとの違いがはつきりわかる、簡潔なリンクテキストにします。これだけで内容が理解できる、メイン指示テキストに対応した表現である必要があります。リンクの本来の意味や、他のリンクとの違

いを、ユーザーが考えこむようなものにはしないでください。

- タスク リンクは常に動詞から始めます(英語の場合)。
- センテンス スタイルの大文字化を使用します。
- 末尾に句読点は付けません。
- タスク リンクに詳細な説明が必要な場合は、別のテキスト コントロールで説明を提示します。このテキストは完全な文とし、末尾に句点を付けます。ただし、この説明は必要な場合にのみ追加します。1つのタスク リンクに説明が必要な場合でも、他のすべてのタスク リンクに説明を付ける必要はありません。

詳細と例については、「[リンク](#)」を参照してください。

#### コミット ボタン

- 単独で意味をなし、メイン指示テキストに対応した、具体的なコミット ボタンラベルを使用します。ラベル以外のものを読まなくてもユーザーが意味を理解できることが理想です。一般にユーザーは、静的テキストよりも、コマンド ボタンのラベルを優先的に読む傾向があります。
- コミット ボタンのラベルは常に動詞から始めます(英語の場合)。
- [閉じる]、[完了]、または[終了]は使用しないでください。これらのボタンラベルは、他の種類のウィンドウでの使用に適しています。
- センテンス スタイルの大文字化を使用します。
- 末尾に句読点は付けません。
- 一意な[アクセスキー](#)を割り当てます。
  - 例外: [キャンセル] ボタンでは、Esc キーが既定のアクセスキーになっているため、アクセスキーを割り当てません。これらのボタンにアクセスキーを割り当てないことで、他のアクセスキーの割り当てが容易になります。

#### ドキュメント

コントロール パネルのホーム ページまたはカテゴリー ページに言及するときは、以下のことに留意します。

- ユーザー向けドキュメントでは、"コントロール パネル"と表現します。また、タイトルスタイルの大文字化を使用し、定冠詞 "the" は付けません(英語の場合)。
- 例:
- コントロール パネルの [セキュリティ センター] を開きます。
- プログラミングおよびその他の技術文書では、"コントロール パネルのホーム ページ" および "コントロール パネルのカテゴリー ページ" と表現します。単語の先頭は大文字にしません(英語の場合)。定冠詞は省略可能です(英語の場合)。

コントロール パネル アイテムは、次のように表現します。

- ユーザー向けドキュメントで、個々のコントロール パネル アイテムを示す場合は、"コントロール パネルの [コントロール パネル アイテム名]" と表現します。また、一般的には、"コントロール パネル アイテム" とします。個々のコントロール パネル アイテムを示す場合に、"アプレット"、"プログラム"、または"コントロール パネル" という語を使用しないでください。
- コントロール パネル アイテムのハブ ページを示す場合は、"[コントロール パネル アイテム名] のメイン ページ" と表現します。
- 名前を半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。

例:

- コントロール パネルの [保護者による制限] を開きます。
- [保護者による制限] のメイン ページに戻ります。

# コントロール パネルの使用パターン

## コントロールパネル

コントロールパネル アイテムに対しては、タスク フローまたはプロパティ シートを使用できます。これらの使用パターンを以下に示します。

### タスク フローのパターン

タスク フロー アイテムの場合、高度なレベルの選択肢を提示するには "ハブ ページ" を使用し、実際に構成を実行するには "スパーク ページ" を使用します。ハブとスパークの組み合わせにはいくつかのパターンがあります。

#### タスクベースのスパーク ページを使用したハブ

この場合、アイテム全体が個々のタスクから構成されています。

#### タスクベースおよびフォームベースのスパーク ページを使用したハブ

この場合、タスクベースのスパーク ページは最も一般的で重要なタスクに使用され、フォームベースのスパーク ページはあまり一般的でなく重要度の低い詳細プロパティの設定に使用されます。

**注:** タスクベースのスパーク ページのみを使用すると、アイテム数が膨大になり、使用が困難になります。タスクベースのスパーク ページがあまり適していないプロパティがある場合は、フォームベースのスパーク ページを使用することを検討してください。

#### スパークを使用しないオブジェクトベースのハブ

この場合、オブジェクトベースのハブ ページにすべてのプロパティおよびタスクが配置されるので(ダイアログ ボックスを使用して簡単に入力できるなど)、スパーク ページは不要です。

#### ハブを使用しないフォームベースのスパーク ページ

この場合、関連するすべてのタスクおよびプロパティがフォームに含まれているので、ハブ ページは不要です。

### タスクベースのハブ ページ

最も一般的に使用されるタスクを表示するハブ ページです。タスク リンクをクリックするとスパーク ページに移動し、そのページでシステム機能を構成したり、関連タスクを実行したりできます。少数の一般的なタスクや重要なタスクに対して、比較的詳細な手順や説明が必要な場合は、ハブ ページを使用するのが最も適しています。構成は、システム全体(日時、セキュリティ、電源オプションなど)に対して設定します。また、一般的に、1つまたは2つのオブジェクトに関係があります(モニター、キーボード、マウス、ゲーム コントローラーなど)。ハブ ページには、コミット ボタンを配置しません。

"ハイブリッド" バージョンはタスクベースのハブ ページで、ハブ ページ上にいくつかのプロパティまたはコマンドが直接配置されています。ユーザーがアイテムを使用してプロパティまたはコマンドにアクセスする可能性が高い場合は、ハイブリッド ハブ ページを使用することを強くお勧めします。



典型的なハイブリッドのタスクベース ハブ ページ。

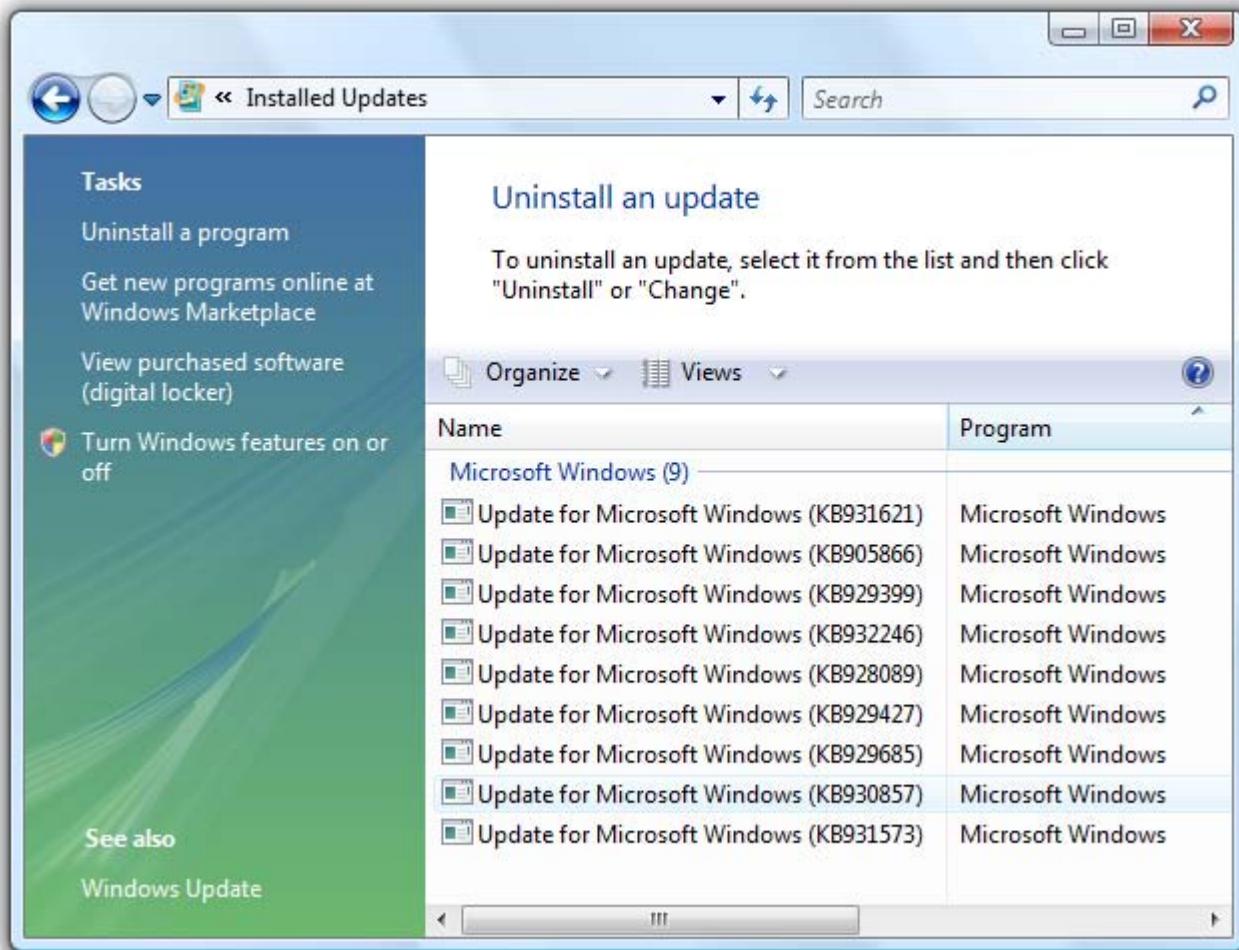
ハブ ページに関する詳細なガイドラインについては、「[コントロール パネル](#)」の関連するセクションを参照してください。

#### オブジェクトベースのハブ ページ

使用可能なオブジェクトを表示するハブ ページです。オブジェクトをシングルクリックすると選択できます。オブジェクトをダブルクリックすると、そのオブジェクトが選択され、さらにそのオブジェクトのプロパティが表示されたスポーク ページに移動できます。オブジェクトが複数ある場合に最も適しています。システム全体に対する設定が含まれる場合もありますが、ほとんどの構成タスクは、選択されたオブジェクト(たとえば、ユーザー アカウント、ネットワーク接続、プリンター)にのみ適用されます。ハブ ページには、コミット ボタンを配置しません。

#### 標準バージョン

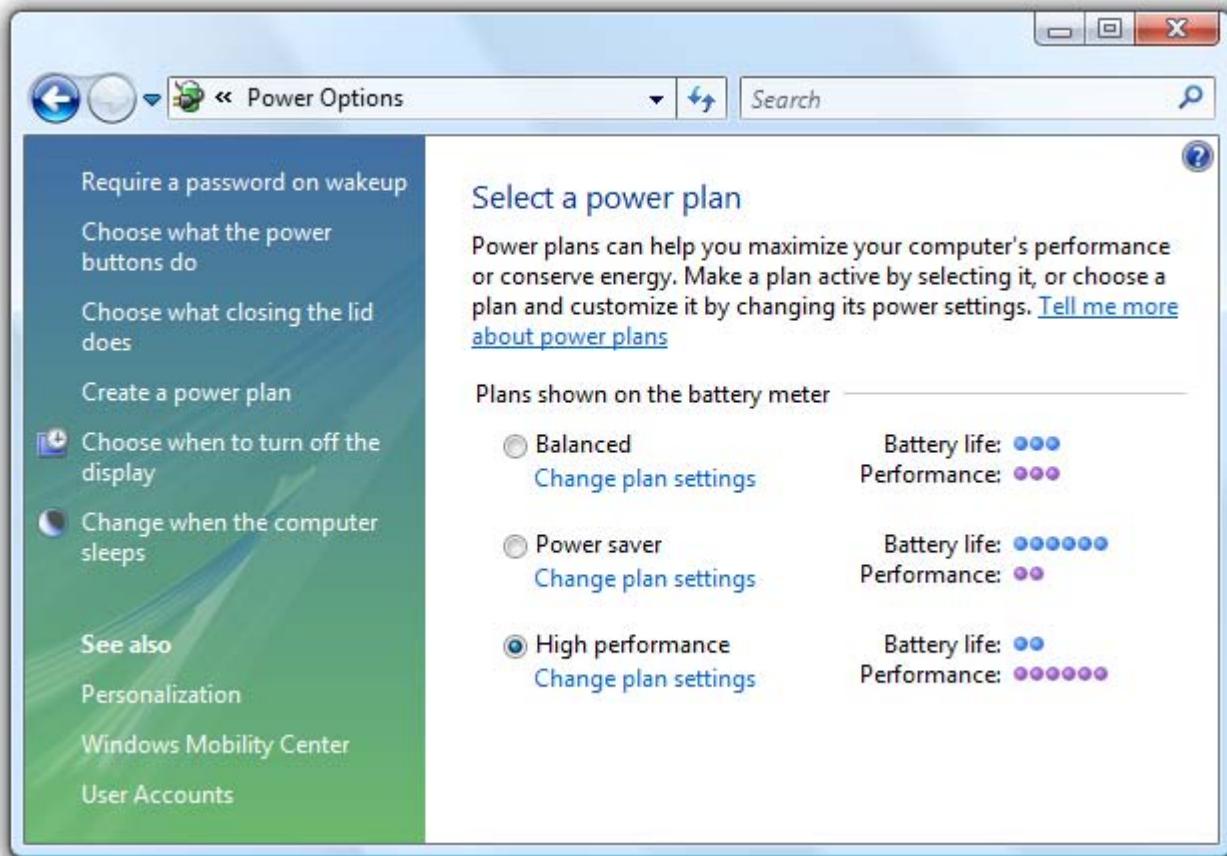
このパターンには 2 つのバリエーションがあります。“標準”バージョンの実装には標準 [リスト ビュー](#) コントロールが使用されます。



典型的なオブジェクトベースのハブ ページの実装には標準リスト ビュー コントロールが使用されます。

### ハイブリッド バージョン

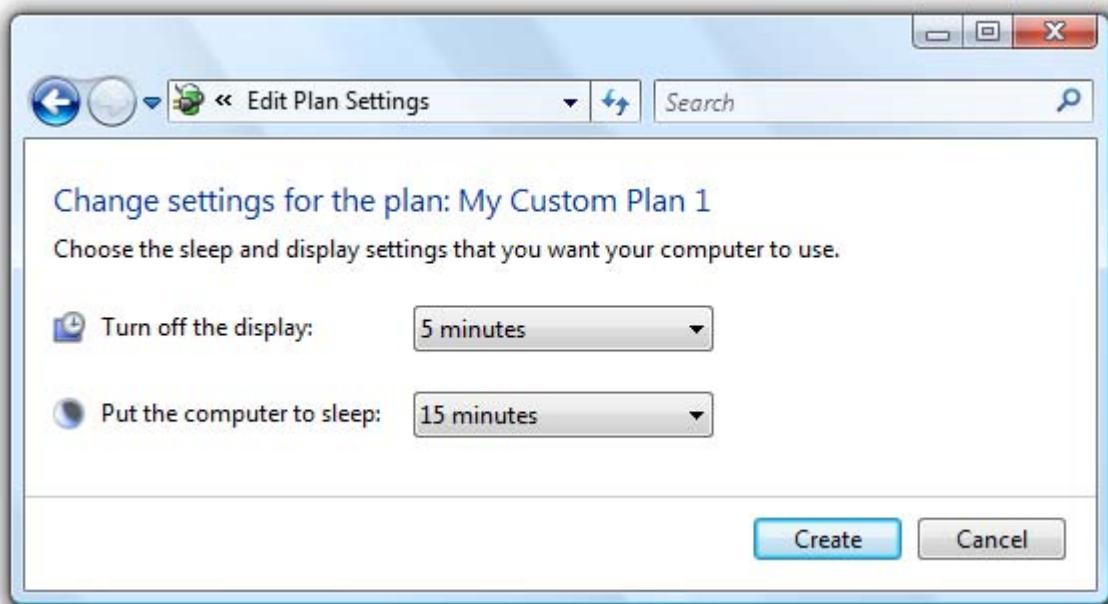
"ハイブリッド" バージョンはオブジェクトベースのハブ ページで、ハブ ページ上にいくつかのプロパティまたはコマンドが直接配置されています。ユーザーがアイテムを使用してプロパティまたはコマンドにアクセスする可能性が高い場合は、ハイブリッド ハブ ページを使用することを強くお勧めします。



この例では、ハブ ページ上に主要な設定が直接配置されています。多くの場合、ユーザーはこれらの設定にアクセスする目的のみでアイテムを使用するため、エクスペリエンスが単純化されます。

### タスク ページ

タスクまたは手順と、具体的なタスクベースのメイン指示テキストを提示するスポーク ページです。詳細な手順や説明を追加することでメリットの得られるようなタスクに最も適しています。タスクの最終ページには、タスクを完了またはキャンセルできるコミット ボタンを配置します。

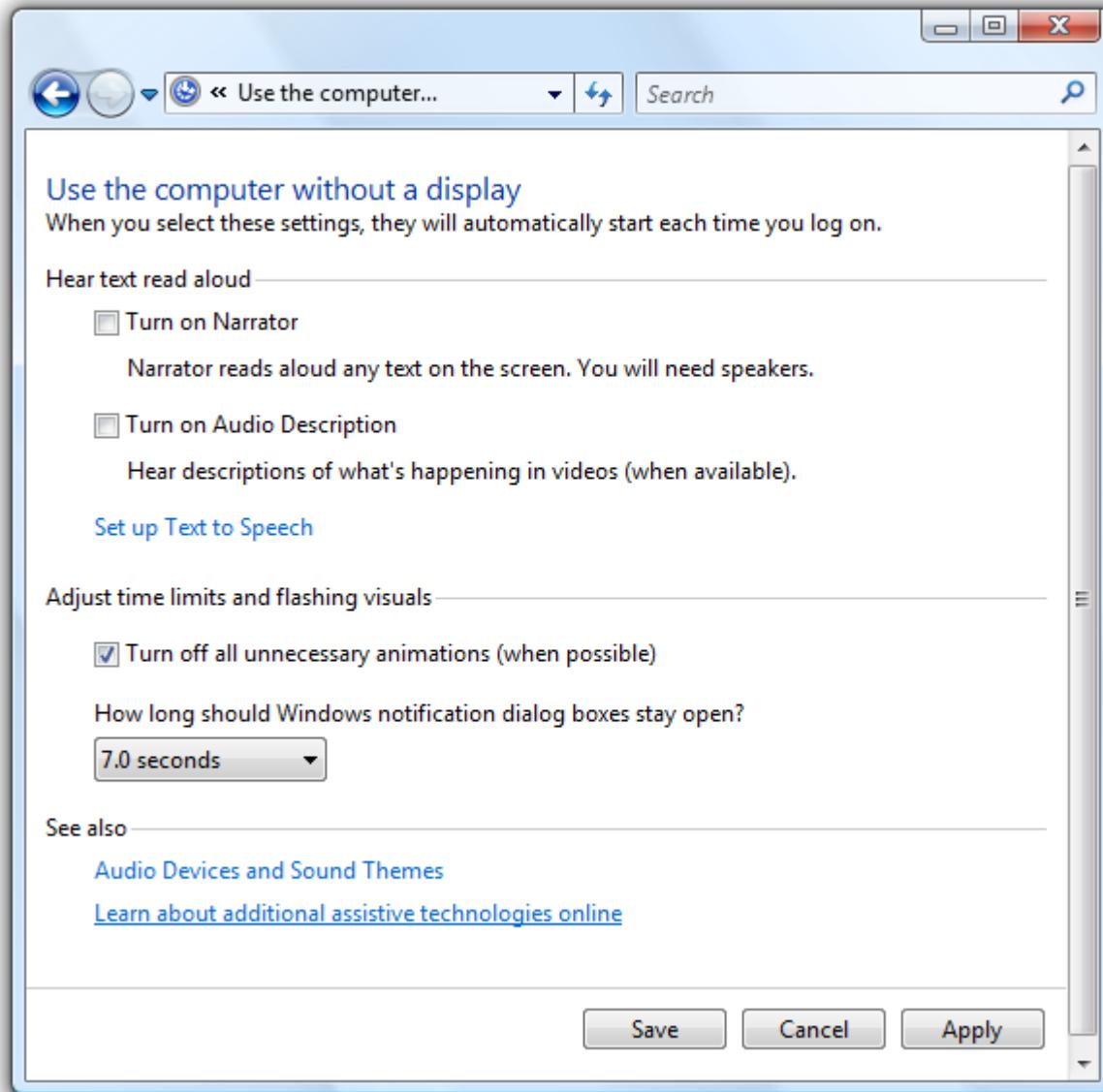


典型的なタスク ページ。

スポーク ページに関する詳細なガイドラインについては、「[コントロール パネル](#)」の関連するセクションを参照してください。

### フォーム ページ

全般的なメイン指示テキストと、これに関連するプロパティやタスクの集まりを提示するスパークページです。フォームページは、多数のプロパティを持ち、これらを1ページに直接表示すると便利な機能に最も適しています。たとえば、詳細プロパティなどがあります。タスクの最終ページには、タスクを完了またはキャンセルできるコミットボタンを配置します。

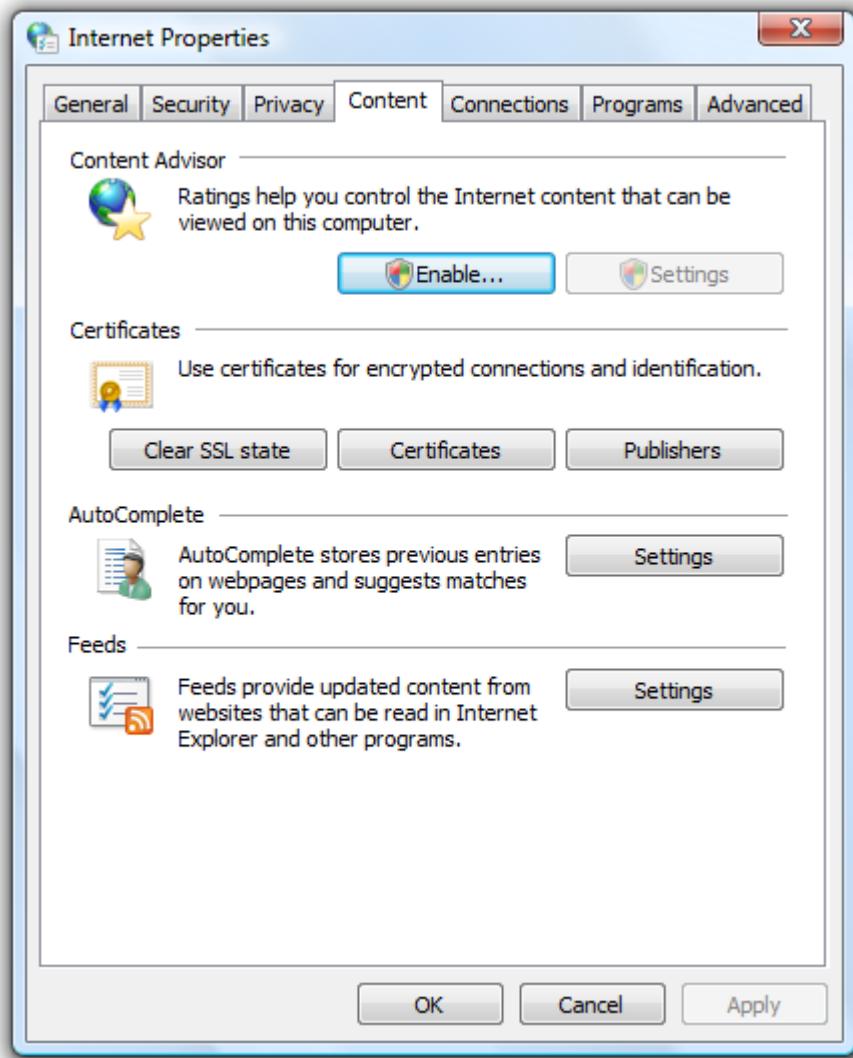


典型的なフォームページ。

スパークページに関する詳細なガイドラインについては、「[コントロールパネル](#)」の関連するセクションを参照してください。

### プロパティシートのパターン

プロパティシートは、詳しい知識のあるユーザーを対象とした、設定が多く、従来から存在するアイテムに最も適しています。新しいアイテムの場合は、フォームページのパターンを使ったタスクフローを使用すると同様の効果が得られます。



このアイテムの実装には、プロパティ シートが使用されています。

## ヘルプ

適切なユーザーインターフェイスかどうかの判断基準

デザインコンセプト

使用パターン

ガイドライン

エントリーポイント

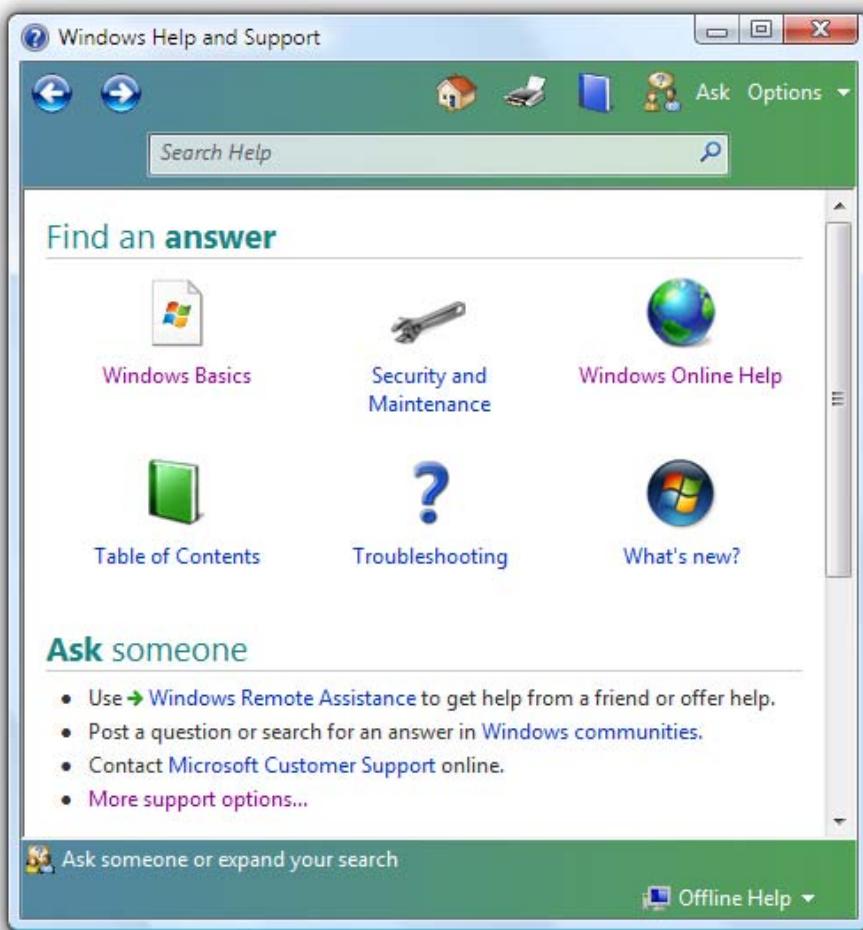
コンテンツ

アイコン

テキスト

"ヘルプシステム" は、ユーザーがタスクを完了できないときや、概念の詳細を理解したいとき、または UI から入手できるよりも詳細な技術情報が必要なときの支援を行うために設計された、さまざまなタイプのコンテンツで構成されています。

ここでは、UI を補助するものとしてのヘルプについて考えていきます。ユーザーが最初に問題を解決しようとする場所は UI ので、UI が第一です。ユーザーは、UI でタスクを達成できない場合にのみヘルプシステムを閲覧します。



Windows® のヘルプとサポート ホーム ページ (スタート メニューからアクセス可能)

注: [スタイルとトーン](#)に関するガイドラインは、別の項目として記載しています。

### 適切なユーザーインターフェイスかどうかの判断基準

以下の点に基づいて判断します。

- 対象ユーザーの動機の強さ。ユーザーは、プログラムの機能を知りたいと思ったり、そのプログラムの中級または上級ユーザーになりたいと思えば思うほど、ヘルプトピックを閲覧して、疑問の回答を進んで探します。
- 質の悪い UI を修正する手段としてヘルプを使用していないかどうか。UI が優れていれば、追加のヘルプを探すユーザーは少なくなります。プログラムのプライマリ UI が非常に明確で役立つものであれば(専門用語のないエラー メッセージ、効果的に記述されたウィザード、わかりやすいダイアログ ボックスなど)、補助としてのヘルプシステムはまったく必要ありません。
- 比較的シンプルなプログラムであるかどうか。該当する場合は、必要な支援コンテンツをすべてプライマリ UI 画面に組み入れることを検討します。ユーザーは複雑なタスクを実行するプログラムにおいて、より追加のヘルプを探す傾向があります。

- ・ アプリケーションの対象が、開発者、IT プロフェッショナル、またはその他のソフトウェア専門家であるかどうか。これらのユーザーは、機能をマスターするために、プログラミング言語の規則に関するリファレンス ヘルプや、概念に関する詳細なヘルプに期待しています。

## デザイン コンセプト

プログラムにヘルプを組み込むことにした場合は、ヘルプをデザイン全体に統合します。ヘルプのインターフェイスはシンプル、効率的、かつ適切なものにする必要があります。また、ユーザーが簡単にヘルプを参照した後に作業に戻れるようにする必要があります。ユーザーの時間の観点からヘルプシステムについて考えてみます。まず、プログラム内でユーザーが問題に遭遇する場所を予想し、次に、UI に直接基本的な支援を組み入れて問題を解決し、より詳細なヘルプへの明確で一貫性のあるエントリ ポイントを作成することで、混乱を最小限に抑えることができます。

Windows® アシスタンスは、この原則に従ってデザインされています。以下は、Windows ヘルプのユーザー エクスペリエンスに対するデザインの変更の一部です。

- ・ 特に、ダイアログ ボックス、エラーメッセージ、ウィザードといった UI 画面からヘルプへのリンクの追加など、プライマリ UI からヘルプへのエントリ ポイントが見つけやすくなりました。ヘルプ リンクを選択すると、ユーザーはヘルプ内の適切なトピックに直接移動します。
- ・ コントロール パネルのほとんどのハブ ページおよびシェル フォルダーで、右上隅にあるヘルプ ボタン アイコンが使用できます。
- ・ ユーザーは、オンラインの場合は Windows のオンライン ヘルプとサポートから最新のヘルプ コンテンツを取得することを選択できます。
- ・ ヘルプ トピックが機能中心ではなくタスク ベースになったため、ユーザーはすばやく効率的にタスクを完了できます。
- ・ ヘルプ トピックは、主として既知の主要なユーザー シナリオに基づくようになりました。
- ・ ヘルプ トピックの **トーン** が、現実世界の言葉を使った、より碎けたものになりました。
- ・ ユーザーが一語一句コンテンツを読むことはあまりないので、ヘルプ トピックは効果的に流し読みできるデザインになっています。

## ヘルプを日常生活にたとえると

ヘルプシステムのデザインについてより深く考えるために、日常生活にたとえて検討してみます。見慣れぬ町で、迷子になったとします。どうしますか。多くの人は、次のように反応します。

1. 自分の位置を確認する、ランドマークや道路標識 (場所の名前や案内) を探す。
2. 地図で探す。
3. 最後の手段として、道をたずねる、または友達に電話する。

この町の "インターフェイス" のデザインによって、必要な支援は変わります。通りに適切な標識があり、わかりやすい案内 (病院、空港、博物館、郵便局などへの道順) があり、目立つ地理的な特徴やビルがあれば、道を探すのに役立ちます。

助けを求めるのは最後の手段です。それは、町の "インターフェイス" のデザインが悪く、わかりにくいために機能しなかったことを意味します。助けになることを示す特定の標識がある場所であれば、より助けを求めやすくなります。たとえば、"案内" や "インフォメーション" などの標識がある場所では、市役所などの一般的な場所よりも助けを求めやすくなります。市役所にいるほとんどの人が道を教えてくれるとしても、この事実は変わりません。

助けを求めるときというのは、ただ目的地に到着したいと思っているのにうまくいかないときです。おそらく、町を見学したり、町の歴史を学んだりすることに時間をかけたくない気分のときです。また、この積極性はタスクの重要性によっても異なります。もしホテルの部屋を探そうとしているなら、なんとしても見つけようとするでしょう。しかし、あまり重要でない場所を見つけるのが目的の場合は、ある程度努力したところで諦めがちです。

現実世界で道を見つけるときのこれらの態度は、ユーザーがプログラム内の仮想世界で方法を見つけるときのやり方に対応しています。プライマリ UI ではなくヘルプを探すということは、UI が本質的にわかりにくいためです。優れた UI デザインを採用し、気の効いた "道路標識" を使用することで、このような経験ができる限り軽減し、ユーザーを答えに導いてください。

## ヘルプの必要がなくなるような UI のデザイン

まず、次のようにしてヘルプの必要がなくなるよう努めます。

- ・ 一般的なタスクは見つけやすく、実行しやすくします。
- ・ 明確な **メイン指示テキスト** を提供します。
- ・ 目的やタスクを重視した、明確で簡潔なコントロール ラベルを提供します。

- 必要に応じて、補足指示テキストや説明を提供します。
- 有効な選択肢のみ選択可能なコントロールを使用し、適切な既定値を提供し、すべての入力形式に対応して、エラーを回避することにより、回避可能な問題に事前に対応します。
- 解決方法またはユーザーが取るべきアクションを明確に示すエラー メッセージを出力します。
- フローが不十分なタスクや、明確な理由なく無効になるコントロールの使用など、混乱を招く UI デザインを避けます。
- 開発の初期サイクルからライターやエディターと連携し、プログラム全体を通して高品質で一貫した [UI テキスト](#) を作成します。

UI の使用方法を知るために、ユーザーが別の場所に移動する必要がないようにします。最も重要な情報は、ユーザーが表示中のコンテキストからヘルプ ウィンドウに移動しなくてもよいよう、直接プライマリ UI に追加します。重要な情報がヘルプ トピックにしかないと、その情報はユーザーの目に入らない可能性が高くなります。オプションの情報や、より詳しい情報については、プライマリ UI からヘルプへのリンクを使用した、関連するヘルプ トピックによる補完的な支援を提供します。

#### ユーザーの積極性を考慮する

ほとんどのユーザーにとって、速度と効率は優れたプログラムの最も重要なメリットです。ユーザーは自分の作業を遂行しようとします。一般に、ユーザーはプログラムやそのテクノロジー自体を学ぶことには関心がありません。プログラムが自分の関心事に貢献し、近いうちに問題を解決してくれる限りにおいてのみ、ユーザーの忍耐は続きます。

ユーザーの積極性に合致するようにヘルプ システムを設計します。たとえば、博物館の売店に向かって歩いてくるユーザーがいるとします。タスクをすばやく実行する方法がわからなければ、彼女はすぐに諦めて立ち去ってしまいます。彼女は、ヘルプの使用に時間をかけたくないようです。一方、動機の強いユーザーは、答えを求めるために、より長時間ヘルプ システムを調べることに耐えられます。たとえば、会計簿の決算をするビジネス ユーザーは、新しい会計アプリケーションを最大限に活用するために、進んでヘルプ コンテンツを調べるでしょう。

#### 流し読みのためのコンテンツ記述

ヘルプ トピックは、一語一句読まれるのではなく、特定の情報を探すために流し読みされることを前提に記述します。簡潔に記述し、早く要点に触れ、ユーザーがそれに従って行動できるような情報を提供します。

- "How-to" トピックは、手続き型のヘルプであることをユーザーが認識できるよう、一貫した形式の番号付きの手順を使用して記述します。
- リファレンストピックは、流し読みしやすいことを念頭に置いて記述します。たとえば、UI オプションや言語の構文は表を使用して説明します。
- 概念に関するトピックは、小見出しを付けて論理的に整理し、ユーザーが関心の低いセクションは丸ごとスキップできるようにします。

すべてのヘルプ コンテンツにおいて、標準のテキスト段落ブロックよりも黒丸付きの箇条書きを流し読みする方が簡単です。ただし、黒丸付きの箇条書きは、慎重に使用してください。整理されていない資料の体裁を整えるためには使用しません。

#### 意味のあるコンテンツを作成する

ヘルプ システムであらゆるユーザーのあらゆる疑問を見込んで対応することはできないため、対象ユーザーの主要なシナリオにおける主な疑問に回答することに集中したコンテンツを作成します。たとえば、効果的な検索方法や、ネットワーク接続の確立方法などは、他のタスクよりも需要の高いトピックです。また、機能やテクノロジーそのものを網羅的に説明するのではなく、主要なユーザー シナリオにおけるタスクを中心に取り扱います。

ヒント: テクニカル サポートは、ヘルプ コンテンツの有益な情報源です。ヘルプ デスクでは、通常、ユーザーが実行しようとしている(および失敗している)特定のプログラムやタスクに関して、よくある質問を記録しています。

UI にあるすべての機能にヘルプを提供する必要はありません。何にでもヘルプを作成しようとして、その結果ヘルプが役に立たなくなるということも少なくありません。UI のデザインが優れていれば、このようなヘルプ トピックのほとんどはあまり役に立たず、ただわかりきったことを言い直しているだけになります。

あるタスクを実行する方法が複数ある場合、多くの倍、経験が浅いユーザーが使用する最も一般的な方法を説明するだけではいません。アクセシビリティに関する考慮事項(マウス操作に相当するキーボードショートカットの説明など)、およびプラットフォームに関する考慮事項(タブレットのフォーム ファクターの説明、グラフィカル ユーザー インターフェイスの代わりにコマンド ラインを使用できるサーバー 環境の説明など)は例外です。

ユーザーは、通常、遭遇した問題について、開発者とまったく同じ用語では考えない点に注意してください。たとえば、ユーザーは自分自身のことを "アカウント" と考えるのに違和感を覚える場合があります。したがって、検索や索引の機能は、用語について考えられるバリエーションや類義語に対応できるように設計します。

ただし、プライマリ UI とヘルプシステムの間では、用語が同一でないにしても非常に類似している必要があります。ヘルプシステムの言葉が、画面に表示されている用語と緊密に関連し合っていないと、ユーザーを混乱させる可能性があります。

#### 説得力のあるヘルプ リンク テキストを記述する

プライマリ UI からヘルプ トピックにリンクするときは、説得力のあるヘルプ リンク テキストを使用します。明確で具体的な言葉を使用すると、ユーザーに自信を与えることができます。ユーザーは、汎用的なヘルプ リンク ("ヘルプ (Help)" や "詳細情報 (Learn more)" と書かれたボタン) を使っても、相当時間をかけなければ必要な情報にたどり着けないと考える傾向があります。

#### 5つの重要な点

1. ヘルプが必要ないように UI をデザインします。
2. 対象ユーザーの主要なシナリオにおける主な質問事項を中心としたコンテンツを作成し、ヘルプの品質を向上させます。
3. 流し読みしやすいヘルプ コンテンツを作成します。
4. UI にあるすべての機能にヘルプを提供する必要はないことを理解します。
5. ヘルプのエントリ ポイントは、見つけやすく、説得力のあるものにします。

#### 使用パターン

コンテンツの種類は目的によって異なります。

**手続き型ヘルプ** 手続き型ヘルプは、"何" や "なぜ" ではなく "どのように" といった情報に重点を置く必要があります。

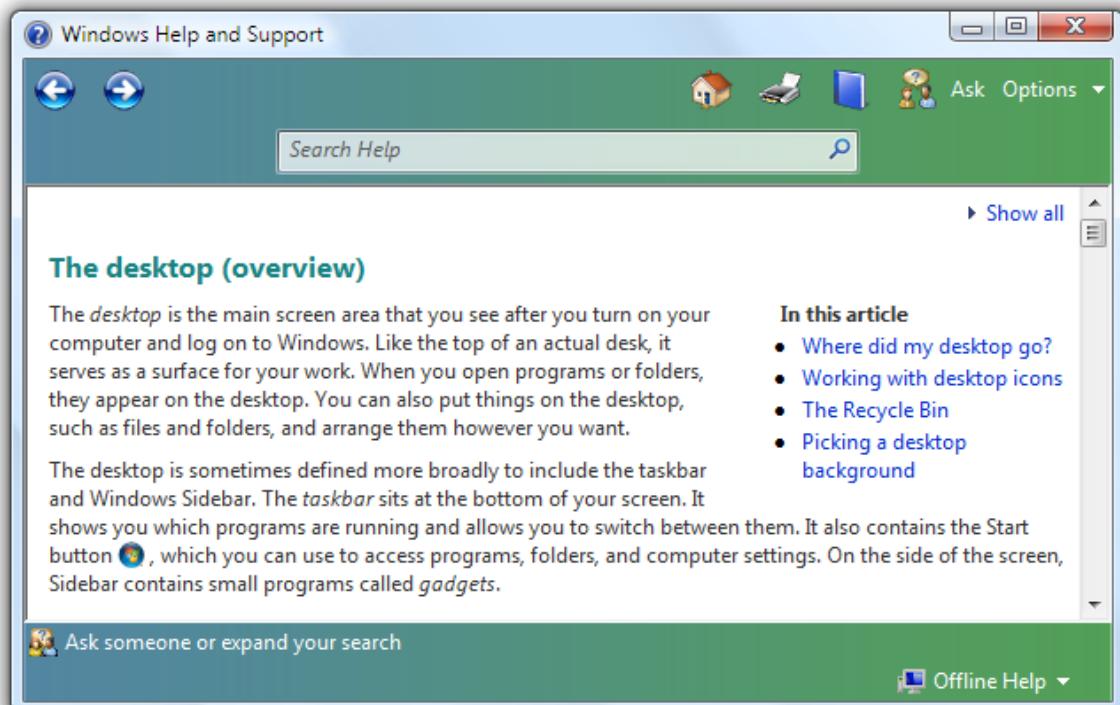
タスクを遂行するための手順を提供します。



このヘルプ トピックの例では、順に追って行う手順を提示することで、ディスクのクリーンアップ ユーティリティの機能の使用方法を説明しています。

**概念に関するヘルプ** 概念に関するヘルプでは、タスクを完了するために必要な情報ではなく、"何" または "なぜ" といった情報を提供する必要があります。

背景情報、機能の概要、しくみなどを説明します。



このヘルプトピックの例では、デスクトップとは何かを明らかにし、デスクトップに含まれる要素やユーザーがデスクトップを操作する理由などの詳細情報を提供しています。

- リファレンスヘルプ リファレンスヘルプを使用すると、プログラミング言語やプログラミングインターフェイスを記述できます。  
オンラインの参考文献として役立ちます。

### Notational Conventions

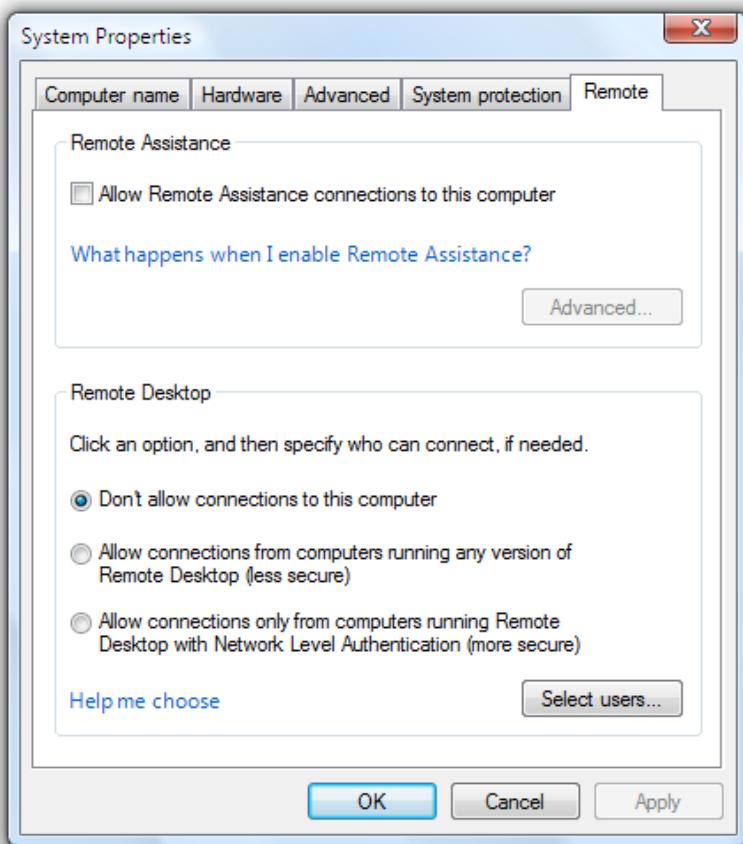
Convention	Meaning
<b>bold</b>	In syntax, characters that you type exactly as shown, including commands and parameters. In text, menu names and menu commands are also bold.
<b>bold monospace</b>	Commands that you must type exactly as shown to get the results being discussed.
<i>italic</i>	Variables for which you supply a specific value. For example, <i>Filename.ext</i> represents any valid file name.
<b>Initial Capitals</b> ( <i>Filename.ext</i> )	Names of files should begin with an initial capital letter, for example, <i>Filename.ext</i> . Paths and folders can be uppercase, lowercase, or mixed, according to how they actually appear in a standard installation of the application or the operating system.
<b>ALL CAPITALS</b>	Used for acronyms.

このヘルプトピックの例では、この特定の言語またはアプリケーションで使用されている体裁の規則を一覧にし、流し読みしやすい表形式で情報を提供しています。

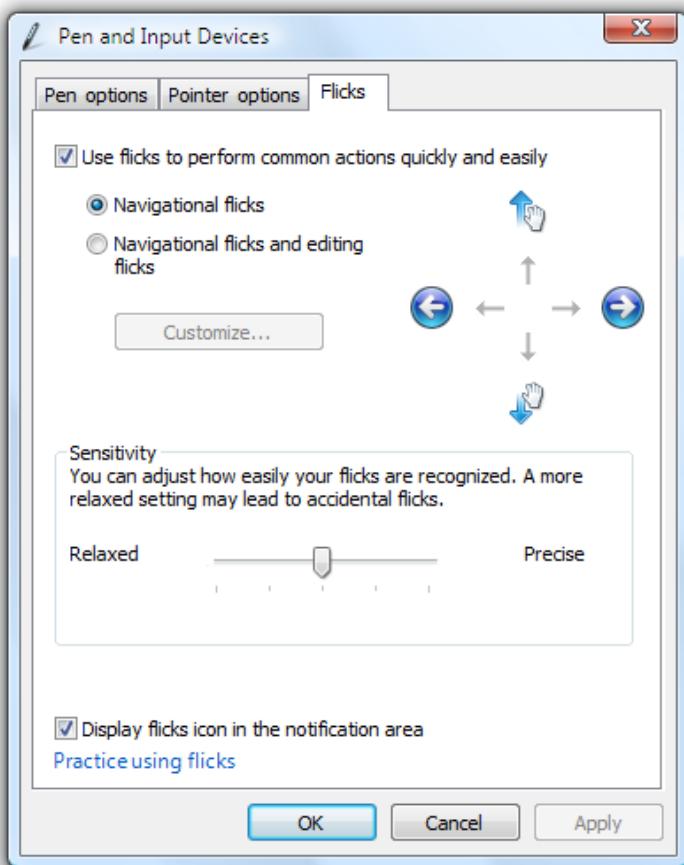
## ガイドライン

### エントリー ポイント

- 具体的で意味のあるヘルプトピックにリンクします。ヘルプのホームページ、目次、検索結果の一覧、または他のページにリンクしているだけのページにはリンクしません。よく寄せられる質問の膨大な一覧で構成されたページにはリンクさせないでください。こうすると、ユーザーはクリックしたリンクに合致する項目を探さなければならなくなります。具体的であっても、作業中のタスクに関連がなく、役に立たないヘルプトピックにはリンクしません。空のページにはリンクしないでください。
- 一貫性を持たせる目的で、すべてのウィンドウやページにヘルプリンクを配置することは避けます。1か所にヘルプリンクを配置したからといって、すべての場所に配置しなければならないわけではありません。
- ヘルプリンクは、ダイアログボックス、エラーメッセージ、ウィザード、プロパティシートで使用します。特定のコントロールに関するヘルプリンクは、コントロールの下に左揃えで配置します。ウィンドウ全体に関するヘルプリンクの場合には、ウィンドウのコンテンツ領域の左下隅に配置します。



この例では、2番目のヘルプリンクはコントロールグループに関連しています。



この例では、ヘルプリンクはウィンドウ全体に関連しています。

- 技術的に可能であれば、汎用的なテキストによるヘルプへの参照ではなく、ヘルプリンクを使用します。

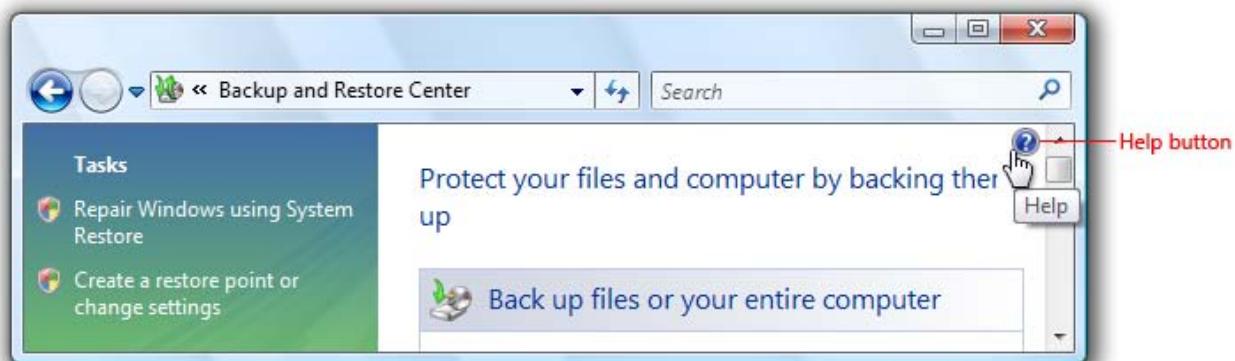
正しい例:

[ディスク エラーの修復方法](#)

間違った例:

ディスク エラーの修復方法については、ヘルプとサポートを参照してください。

- コントロールパネル アイテムのハブ ページでは、ヘルプアイコン付きのヘルプ ボタンを使用します。ヘルプ ボタンは右上隅に配置します。これらのボタンにはラベルがありませんが、ツールヒントに "ヘルプ" と表示されます。



ヘルプ ボタン付きのコントロールパネル アイテム

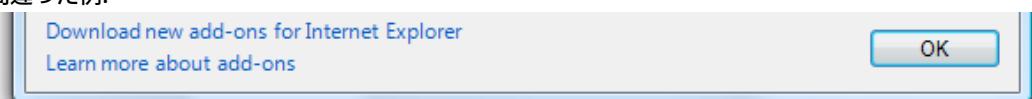
- F1 キーによるヘルプはオプションです。ユーザーは、標準的なキーボードで "Help" とラベルが付いている F1 キーを押して、画面上の UI の現在のコンテキストに関連するヘルプ情報を見つけることに慣れています。たとえば、ユーザビリティに関する調査においてユーザーが F1 キーによるヘルプを期待していることがわかっている場合や、プログラム UI が複雑すぎてコンテキストヘルプが有効でない場合に、F1 キーによるヘルプを組み込みます。
- メニューバー付きのプログラムには、[ヘルプ] メニュー カテゴリを追加できます。[ヘルプ] メニューのガイドラインについては、「[メニュー](#)」を参照してください。



この例では、Windows のペイントアクセサリに [ヘルプ] メニュー カテゴリがあります。

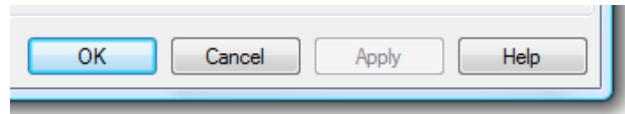
- キーボードによるアクセシビリティのために、ヘルプ ボタンとリンクにもタブストップを適用します。
- ヘルプ ボタンとリンクの動作は、次のようにする必要があります。まず、[ヘルプ] ウィンドウが開いて、専用のヘルプトピックが表示されます。[ヘルプ] ウィンドウを起動した UI は開いたままにし、コンテキストに応じたエクスペリエンスを維持します。
- 以下に示す旧式のヘルプエントリ ポイントのスタイルは使用しないでください。過去には使用されていましたが、ユーザビリティに関する調査で、ユーザーはこのようなエントリ ポイントを無視する傾向があることがわかりました。代わりに、特定のヘルプトピックへのリンクを使用します。

間違った例:



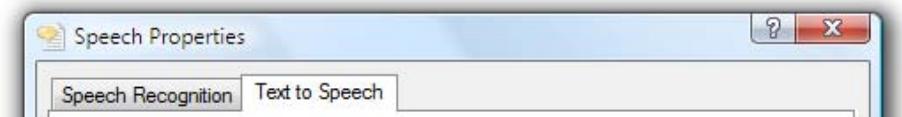
"Learn more (詳細情報)" や "Learn more about... (...に関する詳細情報の表示)" といったリンクは使いません。

間違った例:



汎用的な [ヘルプ] ボタンは使用しません。

間違った例:



タイトルバーでは、コンテキストヘルプのボタンを使用しません。

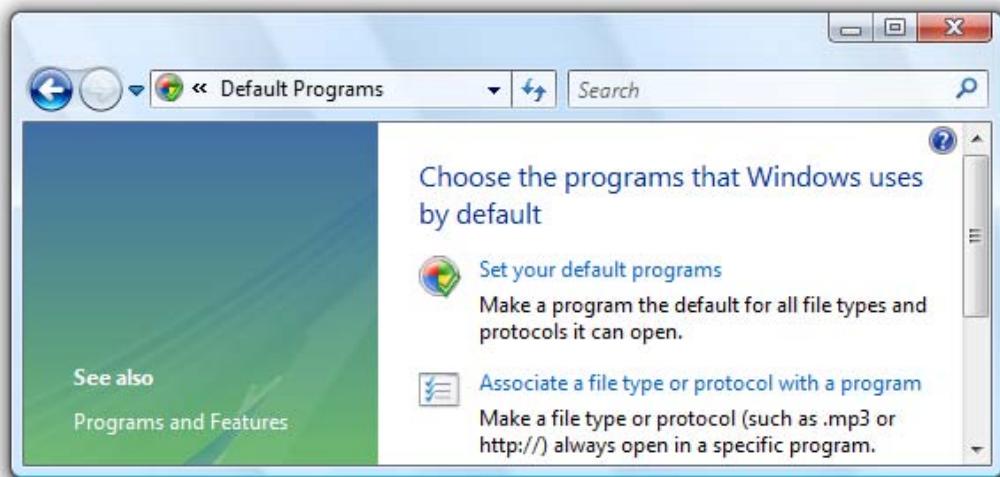
## コンテンツ

- わかりきったコンテンツは作成しないでください。プライマリ UI でわかるなどを繰り返すヘルプトピックに価値はありません。
- ユーザーがそれに従って行動できないコンテンツは作成しないでください。
  - 例外: 概念に関する一部のコンテンツでは、必ずしもユーザーの行動につながらない、重要な背景情報を提供します。
- 問題の解決方法があいまいにならないようにしてください。たとえば、"システム管理者に問い合わせてください" や "アプリケーションを再インストールしてください" という説明は、ユーザーの期待に反します。
  - 例外: 他に現実的な解決方法がない場合は、システム管理者への問い合わせを推奨します。システム管理者は問題についての問い合わせを受けることになります。
- まったくありそうにないユーザーシナリオを解決するコンテンツは作成しないでください。想定している通常の使用に合わせてメインのヘルプコンテンツを開発します。予測される使用方法の重要な例外には言及しますが、このコンテンツは優先度が低いものとして扱います。
- ヘルプトピックがどの程度役に立ったか、ユーザーからのフィードバックを収集します。個々のトピックについて、ユーザーが評価できるようにします。ドキュメンテーションに対して [ユーザビリティ調査](#)を行い、コンテンツの品質や見つけやすさに関する問題を確認します。
  - ヒント: また、ユーザーのフィードバックは、単にテクノロジーの説明に集中している機能ベースのコンテンツとは逆に、ユーザーがプログラムを使用して実際に行っている操作を中心とした、タスクベースのコンテンツを作成する場合に非常に有効な方法となります。
- コンテンツへのアクセス方法を複数用意してください。目次、索引、および [検索メカニズム](#)は、見つけやすさを向上するための最も一般的な手段です。
- あるタスクを実行する方法が複数ある場合、多くの倍、経験が浅いユーザーが使用する最も一般的な方法を説明するだけにまいません。

## アイコン

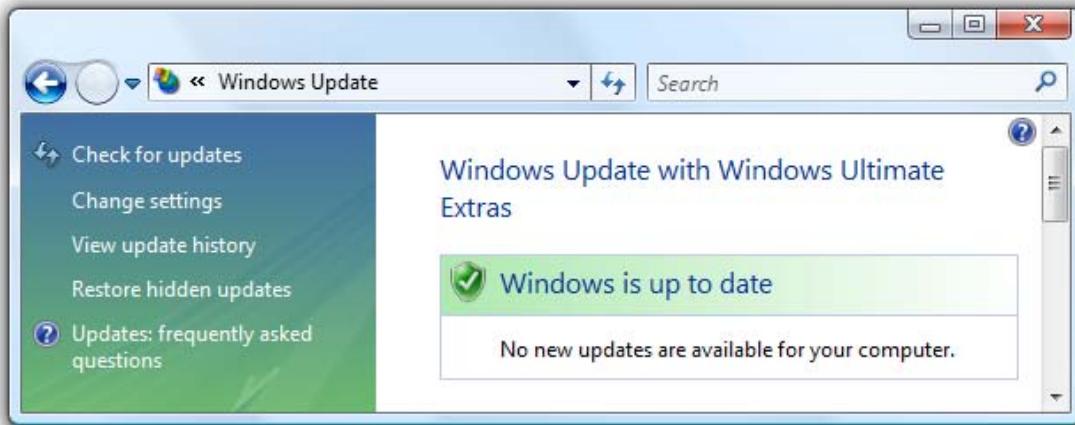
- ヘルプアイコンは、エクスプローラー ウィンドウおよびコントロールパネルアイテムのハブ ページでのみ使用します。ヘルプアイコンをヘルプリンクと一緒に使用しないでください。

正しい例:



この例では、Windows エクスプローラー ウィンドウでヘルプアイコンを使用して、ヘルプへのアクセスを提供しています。

間違った例:



この例では、左下のヘルプ アイコンが誤ってヘルプ リンクと共に使用されています。

## テキスト

### ヘルプ リンク

- 必要に応じて、できる限り関連性のある簡潔なテキストを使用し、ヘルプトピックのコンテンツに関する明確な情報を提供します。ユーザーは、汎用的なヘルプリンクを無視する傾向があります。リンクの結果が予測できる(リンク先でユーザーが驚かない)ようにします。
  - 例外: "詳細情報 (More information)" を使って、UIに直接示されている指示テキストを補足することができます。特に、ヘルプリンクで特定の情報を提供すると不必要に繰り返しが起こる場合やリンクの説得力が損なわれる場合に有効です。

#### 間違った例:

強力なパスワードとは、英数字の大文字と小文字、および記号を組み合わせた、6 文字以上の文字列です。[強力なパスワードとは](#)

#### 正しい例:

強力なパスワードとは、英数字の大文字と小文字、および記号を組み合わせた、6 文字以上の文字列です。[詳細情報](#)

間違った例では、ヘルプリンクが繰り返しになっています(一度回答済みの質問をもう一度たずねています)。

- 可能であれば常に、ヘルプリンク テキストは、メインの質問に対してヘルプコンテンツが回答を提供するような表現にします。"詳細情報の表示 (Learn more about)" や "ヘルプの表示 (Get help with this)" という表現は使用しません。

#### 間違った例:

[例外の追加に関する詳細情報の表示](#)

#### 正しい例:

[例外を許可することのリスクとは](#)  
[例外の追加方法](#)

正しい例のリンクでは、メインの質問に対してヘルプトピックが回答するような表現になっています。

- 最も関連性の高い情報を簡潔にまとめることができる場合は、まとめた情報を直接 UI 上に配置し、ヘルプリンクは使用しません。ただし、ヘルプリンクを使用して補足情報を提供することは可能です。

#### 間違った例:

Enter a strong password:

[What is a strong password?](#)

#### 正しい例:

Enter a strong password:

A strong password has at least six mixed-case letters, numbers, and symbols.

#### より良い例:

Enter a strong password:

A strong password has at least six mixed-case letters, numbers, and symbols. [More information](#)

正しい例では、ヘルプ情報を簡潔にまとめることにより、ユーザーに読んでもらえる可能性が大幅に向 上しています。より良い例では、この複雑なテーマに関する詳細情報を提供するためにヘルプ リンク を配置しています。

- ヘルプ リンクは、ヘルプであることを明確に示す表現にします。ヘルプ リンクがアクション リンクとして解釈されないようになります。
- キーワードだけでなく、ヘルプ リンク全体をリンク テキストに使用します。

正しい例:

[例外を許可することのリスクとは](#)

間違った例:

[例外を許可することのリスクとは](#)

正しい例では、ヘルプ リンクの文全体がリンク テキストに使用されています。

- 例外: 外部 Web サイトへのヘルプ リンクでは、単純にサイト名またはページ名をリンクとして使用する必要があります。サイト名を紹介するテキストを、リンク自体に含める必要はありません。
- ヘルプ リンクは、ヘルプ トピックの見出しに厳密に一致する必要はありませんが、2 つの間には緊密で明確な関連性が存在する必要があります。そこで、リンクと見出しあは 2 つ 1 組でデザインします。

正しい例:

[この機能のパフォーマンスを改善するには \(リンク テキスト\)](#)

最適なパフォーマンスのためにこの機能を構成する (トピックの見出し)

間違った例:

[この機能のパフォーマンスを改善するには \(リンク テキスト\)](#)

この機能の概要 (トピックの見出し)

間違った例では、ヘルプ トピックの見出しがヘルプ リンク テキストと大幅に異なっているため、ユーザーを混乱させる可能性があります。

- ヘルプ コンテンツがオンライン上にある場合は、そのことをリンク テキストに明記します。そうすることによって、リンクの結果が予測しやすくなります。

正しい例:

その他のフォーマットおよびツールについては、[Microsoft の Web サイト](#)を参照してください。

間違った例:

[その他のフォーマットとツールの入手方法](#)

- 完全な文を使用します。
- 疑問符を除いて、末尾に句読点は使用しません。
- ヘルプ リンクやコマンドには省略記号を使用しません。

## ヘルプ コンテンツ

- UI であることがわかりやすいように、UI 要素は半角の角かっこ ([ ]) で囲み、可能な場合は太字にします。これは、手続き型のヘルプ トピックで特に役立ちます。これにより、ユーザーは手続きを流し読みして、該当する UI 要素をすばやく見つけることができます。
- キャプションは斜体にします (英語の場合)。これは、表、アート、スクリーンショット、およびテキストによる簡単な説明が有効なその他のグラフィック要素に当てはまります。
- ヘルプのこととは、単に "ヘルプ" と呼びます。"オンラインヘルプ" という表現は、実際に Web サイト上のコンテンツのことを探している場合を除いて、通常は使用しません。

## ユーザー アカウント制御

ユーザー アカウント制御の設計が優れていると、望ましくないシステム全体の変更が行われることを、予測可能な方法で、最小限の労力により防ぐことができます。

[デザインコンセプト](#)  
[使用パターン](#)  
[ガイドライン](#)  
[UAC シールドアイコン](#)  
[昇格](#)  
[昇格 UI](#)  
[ウィザード](#)  
[テキスト](#)  
[ドキュメント](#)

"ユーザー アカウント制御" (UAC) を完全に有効にすると、現場の管理者は、通常は最小限のユーザー権限で操作を行いながら、承認 UI を使用して明確な承認を行うことにより、自分自身を昇格させて、管理タスクを行うことができます。このような管理タスクには、ソフトウェアやドライバーのインストール、システム全体の設定の変更、他のユーザー アカウントの表示/変更、および管理ツールの実行があります。

最小限の特権を持つ状態の管理者のことを、"保護された管理者" と呼びます。昇格した状態の管理者のことは、"システム特権を持つ管理者" と呼びます。一方、"標準ユーザー" は自分自身を昇格させることができますか、資格情報 UI を使用して昇格できるように管理者に依頼することができます。ビルトイン Administrator アカウントの場合は、昇格は必要ありません。



保護された管理者を昇格して管理者特権を持たせるために使用する承認 UI



標準ユーザーを昇格させるために使用する資格情報 UI

UAC を使用するメリットは次のとおりです。

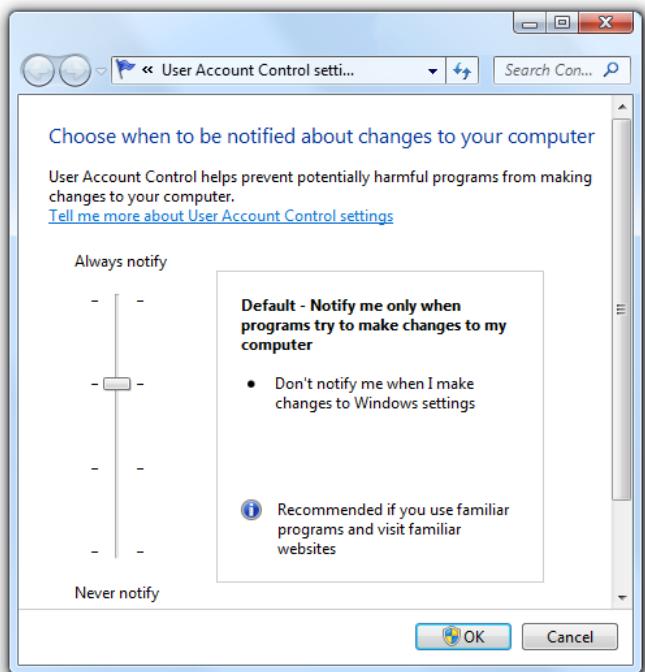
- UAC を使用すると、昇格した権限で動作するプログラムの数が減るので、ユーザーが誤ってシステム設定を変更したり、"マルウェア" にシステム全体のアクセス権を取得されることがありますを防ぐのに役立ちます。昇格が拒否された場合、マルウェアが影響を及ぼすことができる原因是現在のユーザーのデータのみです。昇格なしに、マルウェアはシステム全体に対する変更や、他のユーザーのデータを侵害することはできません。
- 管理環境においては、優れた設計の UAC を使用すると、不要な制限がなくなるので、標準ユーザーとして実行する際のユーザーの生産性を向上させることができます。
- 標準ユーザーは、現在のセッション内で管理タスクを実行するため、管理者に権限の付与を依頼することができます。
- 家庭環境においては、インストールされるソフトウェアなど、システム全体の変更に対する保護者による制限が強化されます。

開発者向け情報: 実装に関する情報については、[UAC との互換性のための UI のデザイン変更に関するページ](#)を参照してください。

Windows Vista® では、保護された管理者は、すべてのシステム変更について通知を受けるか、一切通知を受けないかを選択できます。UAC の既定の設定では、元が何であれすべての変更について通知されます。通知があると、デスクトップが暗くなり、UAC ダイアログ ボックスで要求を承認または拒否するまでは、コンピューター上で他の操作を行うことができません。デスクトップが暗くなっている間は他のプログラムが動作できないため、このようなデスクトップのことをセキュリティで保護されたデスクトップと呼びます。

Windows® 7 では、Windows Vista で追加された 2 つの UAC 設定に加え、保護された管理者用に 2 つの中間的な UAC 設定が導入されています。1 つはプログラムが変更を実行しているときにだけユーザーに通知する設定です。管理者が自分自身で変更を行うときには自動的に昇格します。Windows 7 ではこれが既定の UAC 設定です。この設定でもセキュリティで保護されたデスクトップが使用されます。

Windows 7の中間設定の2つ目は、セキュリティで保護されたデスクトップを使用しない以外は1つ目と同じです。



Windows 7では2つの中間的なUAC設定が導入されています。

注: [ユーザー アカウント制御をサポートするためのコードの記述に関するガイドライン](#)は、別の項目として記載しています。

## デザインコンセプト

### 目的

優れた設計のユーザー アカウント制御の目的は、以下のとおりです。

- 不必要的昇格を省きます。ユーザーが昇格するのは、"管理者特権を必要とするタスク"を実行する場合に限定します。その他のすべてのタスクを昇格なしで行えるように設計する必要があります。従来のソフトウェアの多くでは、レジストリ セクションの HKLM や HKCR、または Program Files フォルダーや Windows システム フォルダーへの書き込みに対して、不必要的管理者特権が頻繁に求められます。
- 予測可能な状態にします。標準ユーザーは、実行するために管理者である必要があるタスク、または管理環境では一切実行できないタスクがどれかを知る必要があります。管理者は、昇格が必要なタスクがどれかを知る必要があります。昇格の必要性を正確に予測できなければ、ユーザーは不適切に管理者タスクに承認しやすくなります。
- 必要な労力を最小限に抑えます。管理特権が必要なタスクは、一度の昇格で実行できるように設計します。複数回の昇格が必要なタスクは、すぐに面倒になります。
- 最小限の権限に戻ります。管理者特権が必要なタスクが完了したら、プログラムは最小限の権限の状態に戻る必要があります。

### 昇格タスクのフロー

タスクに昇格が必要な場合の手順は、以下のとおりです。

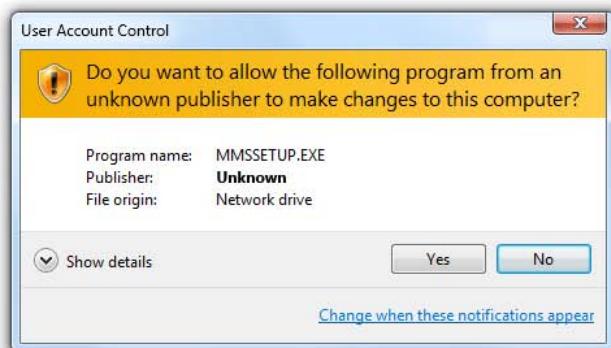
- エントリ ポイント。UACが完全に有効になっている場合、即時の昇格が必要なタスクのエントリ ポイントには、UAC シールドのマークが付きます。この場合、ユーザーがそのようなコマンドをクリックした後すぐに昇格 UI が表示されます。シールドの付いていないタスクから昇格 UI が表示された場合は、特に注意が必要です。



この例では、保護者による制限とユーザー アカウントのコントロールパネル アイテムで昇格が必要です。

UACが部分的に有効になっている場合、または完全に無効になっている場合、昇格 UI が表示されなくとも、タスクがシステム レベルの変更を伴うものであり、昇格が必要であることを示すために、UAC シールドは表示されます。昇格が必要なタスクに対して常に UAC シールドを表示することによって、UI を常にシンプルで予測可能な状態にできます。

- 昇格。保護された管理者の場合、タスクによって承認 UI が表示され、承認が求められます。標準ユーザーの場合、タスクによって資格情報 UI が表示され、管理者資格情報が求められます。



これらは、資格情報 UI と承認 UI の例です。

3. 昇格したプロセスを分ける。タスクを実行するために、昇格した新しいプロセスが内部的に作成されます。
4. 最小限の権限に戻す。必要に応じて最小限の権限に戻し、昇格が不要なその他の手順を完了します。

タスクは昇格した状態を "記憶" しません。たとえば、ウィザードにおいて昇格のエントリ ポイントを行き来する場合、ユーザーは毎回昇格する必要があります。

#### 使用パターン

ユーザー アカウント制御には、以下に示す複数の使用パターンがあります (優先度順)。

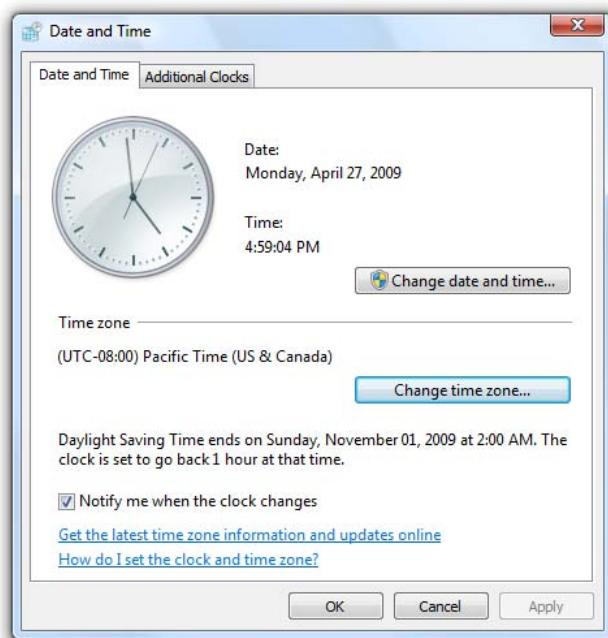
1. 標準ユーザーの作業。機能の範囲を現在のユーザーに限定することによって、すべてのユーザー用に機能を設計します。設定を現在のユーザー (システム全体の逆) に限定することで、昇格 UI を一切不要にし、ユーザーがタスクを完了できるようにします。

間違った例:



この例が示すように、Windows XP ユーザーは現在のタイム ゾーンを確認または変更するために管理者特權を持つ必要がありました。

正しい例:



この例が示すように、Windows 7 および Windows Vista® ではタイム ゾーン機能の設計が変わり、すべてのユーザーが使用できるようになっています。

2. 標準ユーザーと管理者に対して個別の UI 要素を用意する。標準ユーザーのタスクと管理タスクを明確に分離します。有用な読み取り専用の情報へのアクセスをすべてのユーザーに付与します。UAC シールドを使用して、管理タスクを明確に特定します。

Computer name, domain, and workgroup settings

Computer name:	Jonathan-PC	<a href="#">Change settings</a>
Full computer name:	Jonathan-PC.corp.fabrikam.com	
Computer description:		
Domain:	corp.fabrikam.com	

この例では、[システム] コントロールパネルアイテムではすべてのユーザーに対してシステム状態が表示されていますが、システム全体の設定を変更するには昇格が必要です。

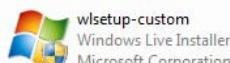
3. 標準ユーザーがタスクを試行し、失敗した場合に昇格を行うことを許可する。標準ユーザーが情報を見ることができ、昇格なしで変更を行うことができる場合は UI へのアクセスを許可し、タスクが失敗した場合にのみ昇格させます。このアプローチは、Windows エクスプローラー内にある自分のファイルのプロパティなど、標準ユーザーのアクセス権が限定されている場合に適しています。また、コントロールパネルのハイブリッドハブページの設定にも適しています。



この例では、ユーザーがプログラムファイルのプロパティを変更しようとして、権限が足りなかった場合を示しています。ユーザーは昇格して再試行できます。

4. 管理者専用の作業。このアプローチは、管理者用の機能やプログラムに対してのみ使用します。管理者だけを対象とした機能の場合 (かつ標準ユーザーのナビゲーションパスや標準ユーザーにとって有用な読み取り専用の情報がない場合)、UI を表示する前にエントリ ポイントで管理者の資格情報を求めることができます。このアプローチは、長いウィザードやページ フローで、すべてのバスで管理特権が必要な場合に使用します。

プログラム全体が管理者専用の場合は、管理者の資格情報を求めてから起動するようにします。Windows では、このようなプログラムのアイコンには、UAC シールドが重ねられて表示されます。



この例では、このプログラムを起動するには管理者特権が必要です。

## ガイドライン

### UAC シールド アイコン

- UAC が完全に有効になっているときに "即時" の昇格が必要タスクであることを示すために、現在 UAC が完全に有効になっていない場合でも、コントロールに UAC シールドを付けて表示します。ウィザードおよびページ フローにおけるすべてのバスで昇格が必要な場合は、タスクのエントリ ポイントに UAC シールドを表示します。UAC シールドを適切に使用すると、ユーザーは昇格が必要なタイミングを予測しやすくなります。
- 複数バージョンの Windows をサポートするプログラムの場合、少なくとも 1 つのバージョンで昇格が必要であれば UAC シールドを表示します。Windows XP では昇格が不要なため、Windows XP については、パフォーマンスに影響を及ぼすことなく一貫して UAC シールドを削除できる場合は、削除を検討します。
- ほとんどのコンテキストで昇格が不要なタスクには UAC シールドを表示しません。このアプローチは誤解を招く可能性があるので、適切にシールドを表示したコンテキスト依存コマンドを使用するアプローチを採用することを推奨します。



新しいフォルダーの作成コマンドは、システム フォルダーで使用するときのみ昇格が必要なので、ここで UAC シールドなしで表示されています。

- UAC シールドを表示できるコントロールは以下のとおりです。

コマンド ボタン:



即時の昇格が必要なコマンド ボタン。

コマンド リンク:

## Uninstall using recommended settings

即時の昇格が必要なコマンド リンク。

リンク:

### Change account type

即時の昇格が必要なリンク。

メニュー:

### Run as administrator

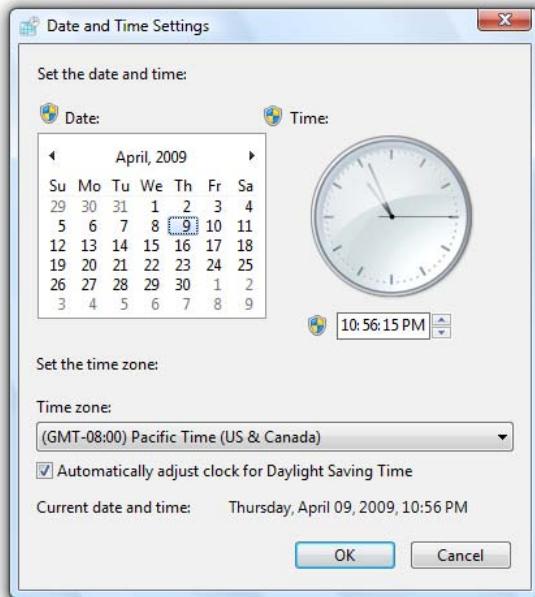
即時の昇格が必要なドロップ ダウン メニュー。

- タスクは昇格した状態を記憶しないので、状態を反映するために UAC シールドを変更しないでください。
- ユーザー アカウント制御がオフになっている場合や、ユーザーがビルトイン Administrator アカウントを使用している場合でも、UAC シールドを表示します。UAC シールドを一貫して表示する方が、プログラミングが簡単です。また、タスクの本質をユーザーに知らせることができます。

## 昇格

- 可能であれば常に、タスクは標準ユーザーが昇格せずに実行するように設計します。有用な読み取り専用の情報へのアクセスをすべてのユーザーに付与します。
- 設定単位ではなく、タスク単位で昇格を行います。標準ユーザーの設定と管理設定を1つのページやダイアログ ボックスに混在させないでください。たとえば、標準ユーザーが変更できる設定とできない設定がある場合は、これらの設定を別の UI 画面に分離します。

間違った例:



この例では、標準ユーザーの設定が誤って管理設定と混在しています。

正しい例:



この例では、日時を変更する設定が別のダイアログ ボックスにあり、管理者だけが使用できるようになっています。タイム ゾーンの設定は標準ユーザーも使用できるので、管理設定とは別にしてあります。

- コントロールを表示するか無効にするかを決める際には、昇格の必要性を考慮しないでください。理由は次のとおりです。
  - 管理されていない環境の場合、標準ユーザーは管理者に依頼することで昇格できます。昇格が必要なコントロールを無効になると、ユーザーは管理者に昇格を求めることができなくなります。

- 管理環境の場合、標準ユーザーは一切昇格できないことが想定されています。昇格が必要なコントロールを削除してしまうと、ユーザーはあきらめるきっかけを失うことになります。
- 不必要的昇格を避けるためには、以下の方針に従います。
  - タスクで昇格が必要な "可能性がある" 場合は、昇格のタイミングをできるだけ遅くします。確認が必要なタスクの場合は、ユーザーが確認を行った場合のみ昇格 UI を表示します。常に昇格が必要なタスクの場合は、エントリ ポイントで昇格を行います。
  - 一度昇格したら、昇格した権限が不要になるまでその状態を維持します。1つのタスクを実行するために複数回の昇格が必要にならないようにします。
  - 変更を行うには昇格が必要な場合で、ユーザーが変更を行わないことを選択した場合は、肯定的なコミット ボタンを有効なままにして、コミットをキャンセル扱いにします。こうすると、ユーザーはウィンドウを閉じるためだけに昇格する必要がなくなります。

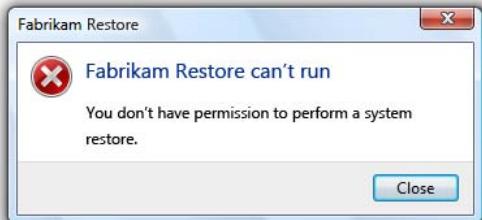
間違った例:



この例では、不必要的昇格が行われないように [変更の保存] ボタンが無効になっていますが、このボタンはユーザーが選択肢を変更すると有効になります。しかし、コミット ボタンが無効になっていると、ユーザーには選択肢がないように見えます。

- ユーザーが昇格しないことを選択した場合は、タスクが失敗してもエラー メッセージを出力しないようにします。ユーザーは操作を続行しないことを意図的に選択したので、この状況をエラーとは考えません。

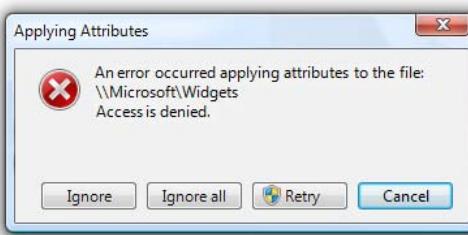
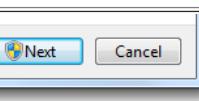
間違った例:



この例では、Fabrikam のリストアでユーザーが昇格しないことを決めたときに、エラー メッセージが表示されています。

- タスクを実行するために権限を昇格させる必要があることを説明するための警告は表示しません。この事実はユーザーが自分で見つけられるようにします。
- UAC シールドと昇格 UI を表示する基準を、次の表に示します。

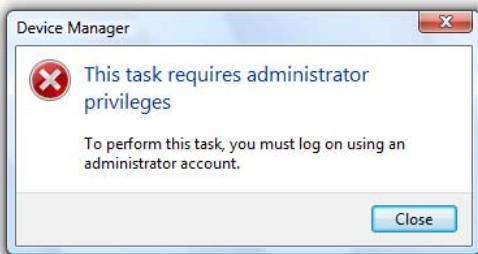
オブジェクト	状況	UAC シールドの配置場所	昇格のタイミング
プログラム	プログラム全体が管理者専用である。	wlsetup-custom Windows Live Installer Microsoft Corporation UAC シールドをプログラム アイコンに重ねて表示します。	起動時に昇格 UI を表示します。
コマンド	コマンド全体が管理者専用である。	Change account type コマンド ボタンまたはコマンド リンク上に UAC シールドを配置します。	コマンド ボタンまたはコマンド リンクがクリックされたとき(ただし、確認操作がある場合)

			合は その 後) に昇 格 UI を表 示し ま す。
コマ ンド	コマ ンド で は、 す べ ての ユ ー ザ に適 した 有 用 な 読 み取 り專 用の 情 報 が表 示 さ れ る が、 変 更 を行 うた めに は管 理特 権が 必 要 と な る。	Computer name, domain, and workgroup settings  Computer name: Jonathan-PC <a href="#">Change settings</a> Full computer name: Jonathan-PC.corp.fabrikam.com Computer description: Domain: corp.fabrikam.com  変更を行うためのコマンド ボタンまたはコマンド リンク上に UAC シールドを配置します。	コマ ンド ボタ ンが ク リッ クさ れた とき (た だ し、 確 認 操 作 が あ る 場 合は そ の 後) に昇 格 UI を表 示し ま す。
コマ ンド	標準 ユ ー ザ は、 昇 格 す る こ と な く 情 報 の 表 示 と 一 部 の 変 更 を 行 う こ と が で き る。 標準 ユ ー ザ がコ マン ドを 試 行 し、 失 敗 し た 場 合 に 昇 格 を 行 う こ と を許 可 す る。	  コマンドには UAC シールドを表示せず、コマンドが失敗した場合に昇格のエントリ ポイントに UAC シールドを表示します。	ユ ー ザ がコ マン ドを 再試 行 し た と き に 昇 格 UI を表 示し ま す。
タス ク手 順	後続 のす べて の手 順で 昇 格 が必 要。	  [次へ] ボタン(またはそれに相当するもの)に UAC シールドを配置します。	[次 へ] ま た はそ の他 のコ ミッ ト ボタ ンが

		クリックされたときに昇格 UI を表示します。
タスク手順	一部の分岐で昇格が必要。	<p> <a href="#">Uninstall using recommended settings</a></p> <p>→ <a href="#">This program uninstalled correctly</a></p> <p>昇格が必要なコマンドリンク上に UAC シールドを配置します。</p>

#### 昇格 UI

- ユーザーが(名前またはパスワードの)有効でない、または管理者特権を持たないアカウントの場合は、単に資格情報 UI を再表示します。エラーメッセージは表示しません。
- ユーザーが資格情報 UI を取り消した場合は、元の UI に戻ります。エラーメッセージは表示しません。
- ユーザー アカウント制御がオフになっているときに、標準ユーザーが昇格の必要なタスクを実行しようとした場合は、"このタスクには管理者特権が必要です。このタスクを実行するには、管理者アカウントを使用してログインする必要があります。" というエラーメッセージを表示します。



この例では、ユーザー アカウント制御がオフになっているので、ユーザーが管理者アカウントを使用する必要があることを説明するエラーメッセージが表示されています。

#### ウィザード

- 複数回の昇格を行わないようにします。一度昇格したウィザードは、昇格した状態を維持する必要があります。
- ウィザード内で実行するタスクの場合、コミットページの[次へ]ボタン(このボタンには、より具体的なラベルを付ける必要があります)に UAC シールドを配置します。ユーザーがコミットを行った場合は、次のようにします。
  - 次のページが進行状況ページの場合は、そのページに進み、昇格 UI をモーダルに表示します。昇格が成功したらタスクを実行します。
  - 次のページが完了ページの場合は、そのページに進み(ただし一時的に"アクセス許可を設定しています。しばらくお待ちください..."というコンテンツに置き換え)、昇格 UI をモーダルに表示します。昇格が成功したら、タスクを実行し、次に完了ページのコンテンツを表示します。
  - ユーザーが昇格 UI を取り消した場合は、コミットページに戻ります。これによって、ユーザーは再試行できます。
- ウィザードが完了した後で実行するタスクの場合は、コミットページの[完了]ボタン(このボタンには、より具体的なラベルを付ける必要があります)に UAC シールドを配置します。ユーザーがコミットを行った場合は、次のようにします。
  - コミットページを表示したままにして、昇格 UI をモーダルに表示します。昇格が成功したらウィザードを閉じます。
  - ユーザーが昇格 UI を取り消した場合は、コミットページに戻ります。これによって、ユーザーは再試行できます。
- 管理者専用の長いウィザードの場合は、UI を表示する前のエントリ ポイントで管理者の資格情報を求めることがあります。

#### テキスト

- 昇格が必要なコマンドだという理由で、省略記号を使用しないでください。昇格の必要性は UAC シールドによって示します。

#### ドキュメント

ユーザー アカウント制御に言及するときは、以下のことに留意します。

- この機能のことは、"ユーザー アカウント制御"(初出時、または"UAC"(初出時以外)と呼びます。"最小限の権限を持つユーザー アカウント"や" LUA"は使用しません。
- 管理者以外のユーザーは"標準ユーザー"と呼びます。
- 組み込みのコンピューター管理者のことは"ビルトイン Administrator"と呼びます。

ユーザー ドキュメントにおいては、以下のことに留意します。

- 管理タスクを実行するために承認することを、"権限を付与する"と呼びます。

プログラミングおよびその他の技術文書においては、以下のこと留意します。

- 管理タスクを実行するために承認することを、"昇格" と呼びます。
- UAC のコンテキストでは、昇格していない管理者のことを "保護された管理者"、昇格した後の管理者のことを "システム特権を持つ管理者" と呼びます。
- パスワードを入力するためのダイアログ ボックスのことを "資格情報 UI" と呼びます。承認するためのダイアログ ボックスのことを "承認 UI" と呼びます。両方のダイアログ ボックスのことを、一般に "昇格 UI" と呼びます。

## ビジュアル索引

コントロール

コマンド

ポインター

ウィンドウ

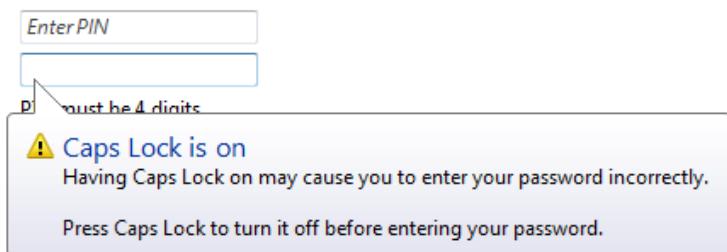
Windows 環境

外観

ここでは、UXガイドで扱われている数多くのユーザーインターフェイス要素について、ビジュアルによるサンプルを示します。イメージをクリックすると、それぞれの要素に関するガイドラインに移動します。

### コントロール

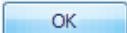
コモン コントロール



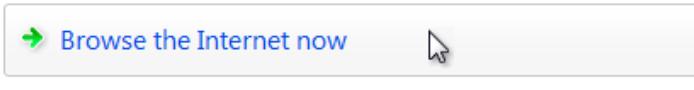
パルーン



チェック ボックス



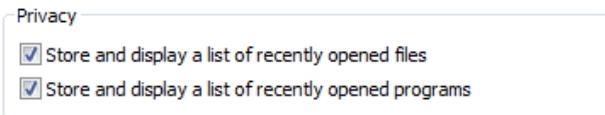
コマンド ボタン



コマンド リンク



ドロップダウン リストとコンボ ボックス



グループ ボックス



## リンク

List of valid modes:

1680 by 1050, True Color (32 bit), 60 Hertz
1680 by 1050, High Color (16 bit), 60 Hertz
1680 by 1050, 256 Colors, 60 Hertz
1280 by 1024, True Color (32 bit), 60 Hertz
1280 by 1024, High Color (16 bit), 60 Hertz
1280 by 1024, 256 Colors, 60 Hertz
1280 by 800, True Color (32 bit), 60 Hertz
1280 by 800, High Color (16 bit), 60 Hertz
1280 by 800, 256 Colors, 60 Hertz

## リスト ボックス

Name	Date taken	Tags	Size	>>
Autumn Leaves				
Creek				
Desert Landscape				
Dock				
Forest				
Forest Flowers				
Frangipani Flowers				

## リスト ビュー



## 通知

from Pictures (D:\User...\Pictures) to Pictures (D:\User...\Pictures)  
Calculating time remaining...



## 進行状況/バー



## 段階的表示コントロール

- Display as a link
- Display as a menu
- Don't display this item

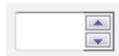
## ラジオ ボタン



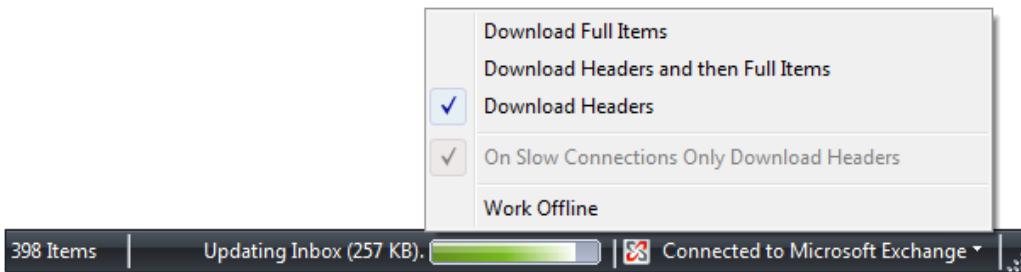
## 検索ボックス



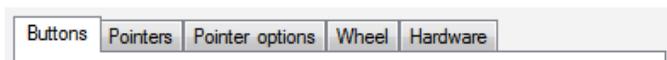
## スライダー



## スピントロール



## ステータスバー

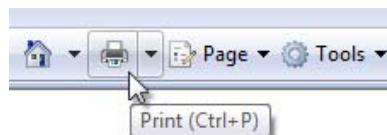


## タブ

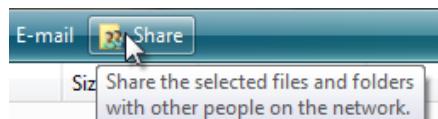
## Display name:



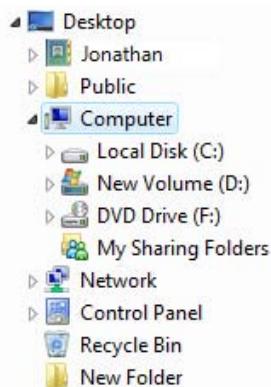
## テキストボックス



## ツールヒント

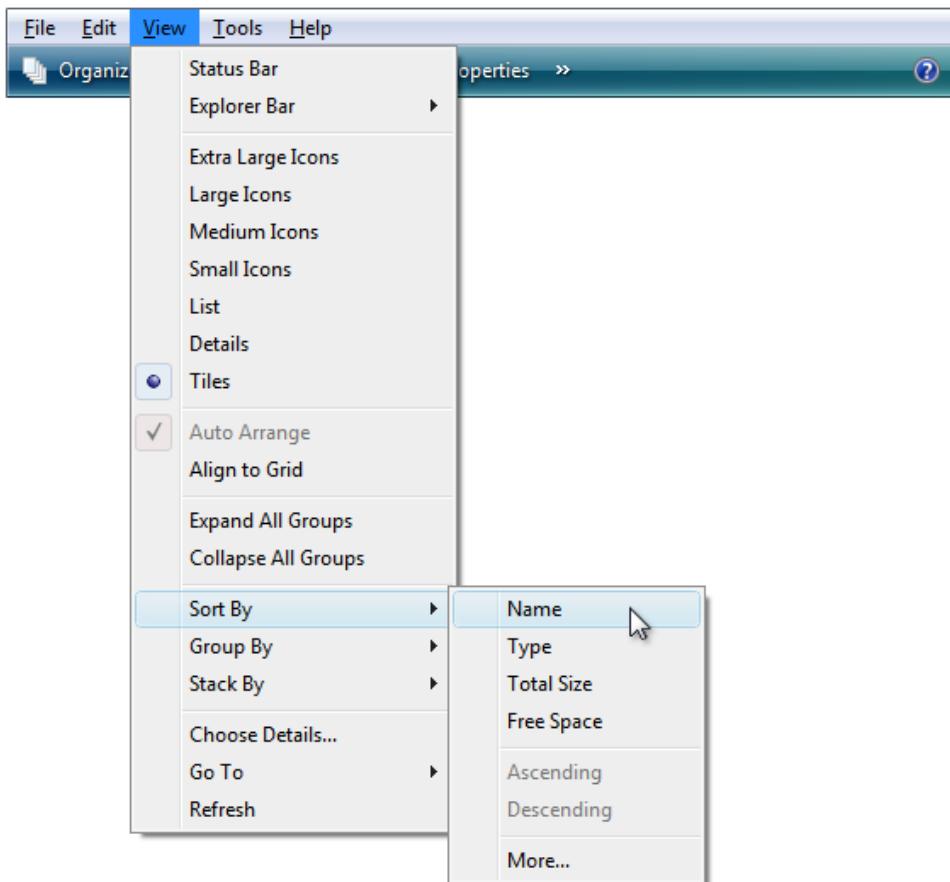


## 情報ヒント



## ツリービュー

## コマンド



メニュー



ツールバー



リボン

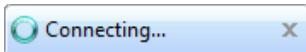
ポインター



バックグラウンドで作業中

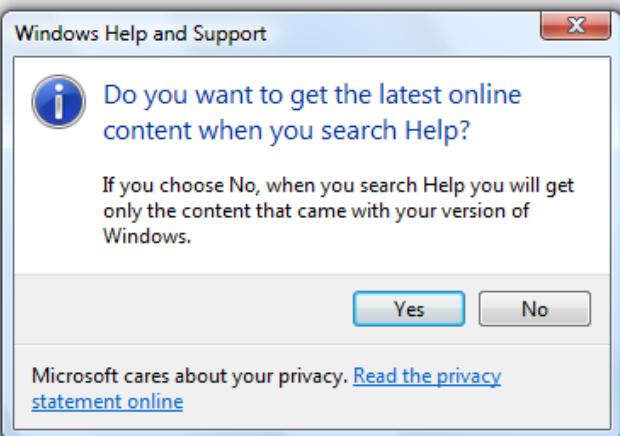


ビジー



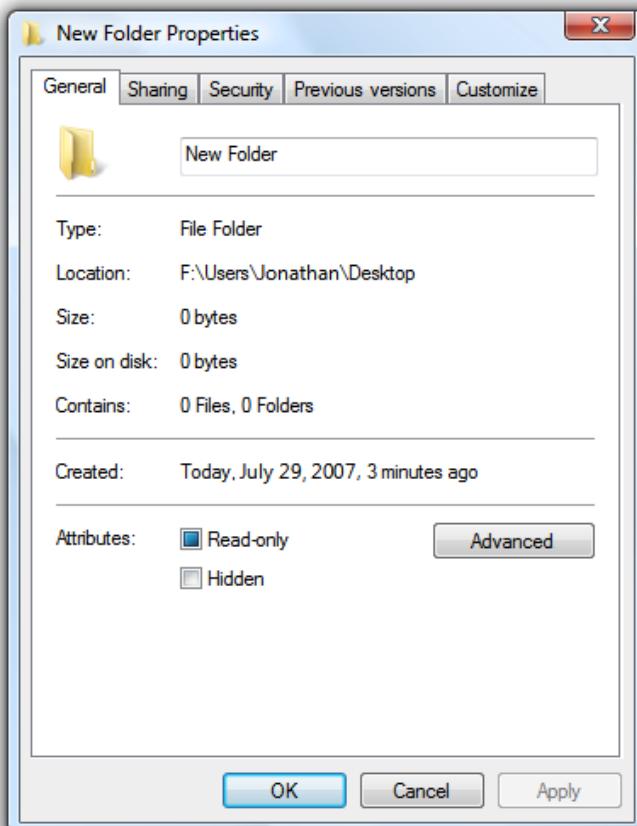
アクティビティ インジケーター

ウィンドウ



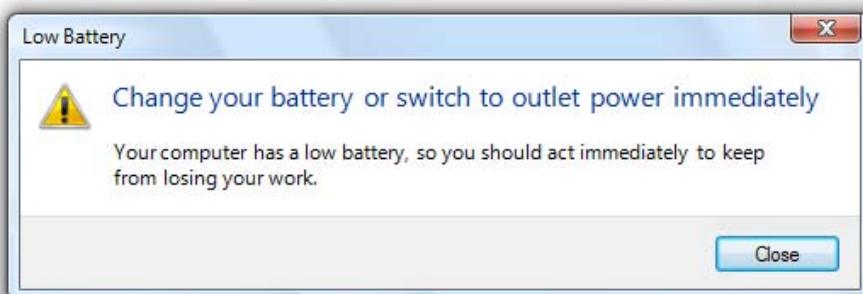
ダイアログ ボックス

---



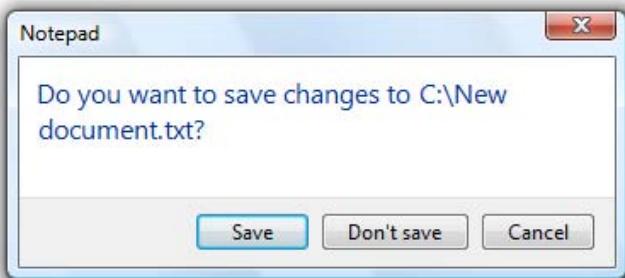
プロパティ ウィンドウ

---



警告メッセージ

---



確認

## Windows 環境



ガジェット (詳細表示/フローティング)



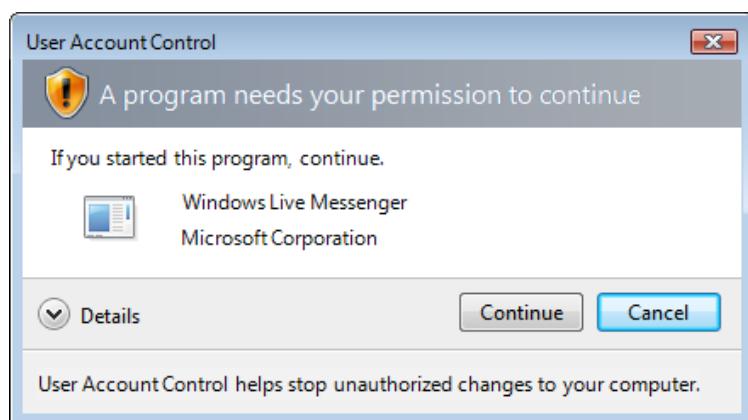
ガジェット (簡易表示/ドッキング)



タスク バー

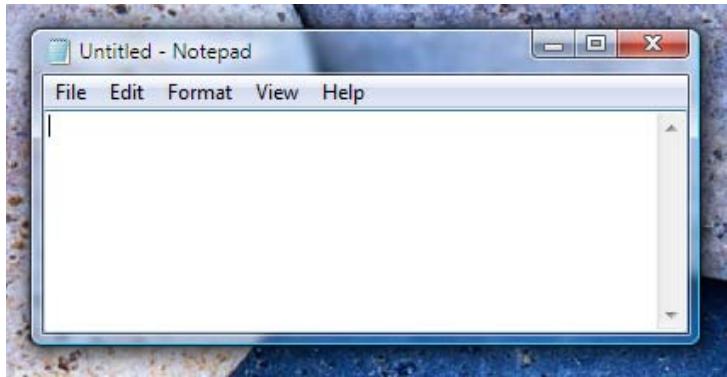


ユーザー アカウント制御のエントリ ポイント



ユーザー アカウント制御の承認 UI

## 外観



ウィンドウ枠

---

abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ  
あいうえおかきくけこさしすせそアイウエオカキクケサシスセソ  
亜哩娃阿哀愛挨始逢葵茜庵惡握渥旭葦芦鯵梓庄斡扱宛姐虻飴綾鮎  
0123456789 () 「」！？、。、。

フォント (メイリオ)

---



標準アイコン

# 用語集

A | B | C | D | E | F | G | H | I | J | K | L | M  
N | O | P | Q | R | S | T | U | V | W | X | Y | Z

A

## [バージョン情報] ボックス (About box)

バージョン ID、著作権情報、使用許諾契約、テクニカルサポートへの問い合わせ方法などの一般的なプログラム情報を表示するダイアログボックス。

## 1面トップ (above the fold)

新聞ジャーナリズムから引用した画面レイアウトの隠喩。新聞の1面トップの内容は、特に売上を伸ばすことを念頭に置く必要があります。同様に、画面レイアウトにおいては、最も重要なコンテンツはスクロールなしで表示できる必要があります。ユーザーが、最初に目にする"1面トップ"のコンテンツをスクロールしてみよう、という気になるようにしなければなりません。

## アクセスキー (access key)

Altキーと組み合わせて、コントロールをアクティビ化する英数字キー。アクセスキーは、コントロールのラベルの文字の1つに下線を引いて示します。たとえば、Altキーを押しながらOキーを押すと、ラベルが"開く(O)"で、割り当てられたアクセスキーが"O"であるコントロールがアクティビ化されます。アクセスキーでは、大文字と小文字は区別されません。コントロールをアクティビ化したときの結果は、コントロールの種類によって異なります。

主に詳しい知識のあるユーザー向けであるショートカットキーとは対照的に、アクセスキーはアクセシビリティの向上を目的としています。アクセスキーは、ユーザーインターフェイス(UI)自身に直接含まれているため、常に一貫して割り当てるることはできず、記憶するためのものではありません。

## アクティブモニター (active monitor)

アクティブなプログラムが実行しているモニター。

## アドレスバー (address bar)

通常、ウィンドウの上部に表示され、ユーザーの現在の場所を表示したり、変更したりできるナビゲーション要素。「階層リンクバー」も参照してください。

## アフォーダンス (affordance)

オブジェクトの使用方法や動作を示す視覚的特性。たとえば、コマンドボタンの外観は、押したりクリックすることを示唆する実際のボタンのようになっています。

## アプリケーション (application)

関連するユーザー タスクを実行するためのプログラム。多くの場合、比較的複雑で高機能なプログラムです。「プログラム」も参照してください。

## アプリケーションキー (application key)

## アプリケーションメニュー (application menu)

ファイル関連のコマンドなど、ドキュメントやワークスペースに対して何らかの操作を行うコマンドのメニューを表示するコントロール。

現在の選択要素のコンテキストメニュー(Shiftキーを押しながらF10キーを押したときと同じメニュー)を表示します。

## アプリケーションテーマ (application theming)

アプリケーション独自の外観やブランドを作成するために、カスタマイズされたコントロールなどの関連する表示技術を使用すること。

## アスペクト比 (aspect ratio)

オブジェクトの幅と高さの関係。たとえば、高精細テレビでは16:9のアスペクト比を使用します。

## オートコンプリート (auto-complete)

以前の入力内容から自動的に入力候補を推定して入力できる、テキスト ボックスや編集可能なドロップダウンリストで使用されるリストの種類。最小限のテキストを入力すると、オートコンプリートリストが表示されます。

## 自動終了 (auto-exit)

ユーザーが最後の文字を入力すると、すぐに入力フォーカスが自動的に次の関連するテキスト ボックスに移動するテキスト ボックス。

B

## バルーン (balloon)

ユーザーに重大でない問題や特殊な状態を通知する一般的な Windows コントロール。

## 階層リンク バー (breadcrumb bar)

通常、ウィンドウの上部に表示され、ユーザーの現在の場所を表示したり、変更したりできるナビゲーション要素。“階層リンク”とは、一連のリンクを矢印で区切り、現在の場所を分割して表示したものをいいます。ユーザーはこのリンクを直接操作できます。階層リンク バーの代わりにはアドレスバーを使用します。「アドレスバー」も参照してください。

C

## チェック ボックス (check box)

オプションのオンとオフを切り替えるなど、ユーザーが明らかに異なる選択肢から指定できる一般的な Windows コントロール。

## シェブロン (chevron)

割り当てられたスペースに表示できない項目があることを示す小さなコントロールまたはボタン（山かっこ）。シェブロンをクリックすると、さらに項目が表示されます。

## 子ウィンドウ (child window)

親ウィンドウと呼ばれる別のウィンドウ内に完全に含まれるコントロールやウィンドウ枠などのウィンドウ。「親ウィンドウ」、「所有されるウィンドウ」も参照してください。

## 選択解除 (cleared)

チェック ボックスで、オプションが設定されていないことを示す。「選択済み」、「混在状態」も参照してください。

## コンボ ボックス (combo box)

ドロップダウンリストまたは標準的なリスト ボックスの特徴と、編集可能なテキスト ボックスを組み合わせた一般的な Windows コントロール。「リスト ボックス」、「ドロップダウンリスト」も参照してください。

## コマンド領域 (command area)

コミット ボタンが配置されたウィンドウ内の領域。通常、ダイアログ ボックスとウィザードにはコマンド領域があります。「コミット ボタン」も参照してください。

## コマンド ボタン (command button)

ユーザーがすぐに操作を開始できる一般的な Windows コントロール。

## コマンド リンク (command link)

関連する相互に排他的な一連の選択肢から選択するためのコントロール。通常の状態では、コマンド リンクの外観はハイパーリンクに似た軽量なものですですが、動作はコマンド ボタンに似ています。

## コミット ボタン (commit button)

作業をコミットする、複数手順の作業で次の手順に進む、作業を取り消すといった操作に使用するコマンド ボタン。「コマンド領域」も参照してください。

## コミット ページ (commit page)

ユーザーが作業の実行をコミットするウィザード ページの種類。作業をコミットすると、[戻る] ボタンや [キャンセル] ボタンをクリックして作業を取り消すことはできません。

## 完了ページ (completion page)

ウィザードの最後を示すためのウィザード ページ。"設定完了" ページの代わりに使用することもあります。「["設定完了" ページ](#)」も参照してください。

## "設定完了" ページ (congratulations page)

ウィザードの最後を示すためのウィザード ページ。このページは現在は推奨されていません。ウィザードは、コミット ページや、必要に応じて、フォローアップ ページや完了ページを使用すると、より効果的に完了します。「[コミット ページ](#)」、「[完了ページ](#)」、「[フォローアップ ページ](#)」も参照してください。

## 承認 UI (Consent UI)

保護された管理者が各自の特権を一時的に昇格できるユーザー アカウント制御 (UAC) で使用されるダイアログ ボックス。

## 制約 (constraint)

テキスト ボックスなどのユーザー入力にかかるコントロールでは、入力の制約はエラーを回避するための有効な方法です。たとえば、特定のコントロールで唯一有効な入力が数値である場合、この要件を強制するために適切な値の制約をコントロールに指定できます。

## コンテンツ領域 (content area)

ダイアログ ボックス、コントロール パネル アイテム、ウィザードなど、オプションの表示、情報の提供、およびコントロールの説明専用の UI 画面領域。コマンド領域、作業 ウィンドウ、ナビゲーション 領域とは区別されます。

## コンテキスト タブ (contextual tab)

特定のオブジェクトの種類を選択している場合にのみ関連するコマンド群を表示するタブ。「[リボン](#)」も参照してください。

## コントロール パネル (Control Panel)

ユーザーのために、ハードウェアおよびソフトウェアのセットアップや構成などのコンピューターのシステムレベルの機能を収集して表示する Windows プログラム。ユーザーは、コントロール パネルで個々のアイテムをクリックして、システムレベルの機能を構成したり、関連作業を実行することができます。「[コントロール パネル アイテム](#)」も参照してください。

## コントロール パネル アイテム (control panel item)

コントロール パネルにある個々の機能。たとえば、[プログラム] と [コンピューターの簡単操作] は 2 つのコントロール パネル アイテムです。

## 資格情報 UI (Credential UI)

標準ユーザーが各自の特権の一時昇格を要求できるユーザー アカウント制御 (UAC) で使用されるダイアログ ボックス。

## 重大 (critical)

最も高い重要度。たとえば、エラー メッセージや警告 メッセージでは、データやプライバシー、システムの整合性などの損失は重大な状況になる可能性があります。

## カスタム アイコン (custom icon)

プログラムに特有の画像 (Windows システム アイコンとは逆)。

## カスタムの外観 (custom visuals)

特にプログラム用に開発されたグラフィックス、アニメーション、アイコンなどの視覚的要素。

## 既定のコマンド ボタンまたはリンク (default command button or link)

ユーザーが Enter キーを押すと呼び出されるコマンド ボタンまたはリンク。既定のコマンド ボタンやリンクは開発者によって割り当てられますが、すべてのコマンド ボタンやリンクは、ユーザーが Tab キーで指定すると既定になります。

## 既定のモニター (default monitor)

[スタート] メニュー、タスク バー、および通知領域が表示されたモニター。

## 遅延型のコミット モデル (delayed commit model)

ユーザーがコミット ボタンをクリックして明示的にコミットするまで変更が行われない、コントロール パネル アイテムのスポーク ページで使用されるコミット モデル。ユーザーは、[戻る] ボタン、[閉じる] ボタン、またはアドレス バーを使って移動することによって、作業を破棄できます。「[即時型のコミット モデル](#)」も参照してください。

## デスクトップ (desktop)

実際の机の作業面に似た、Windows の画面上の作業領域。「[作業領域](#)」も参照してください。

## リスクのあるコマンド (destructive command)

大きな影響が生じ、簡単に元に戻すことができないか、すぐに気付くことができない操作。

## 詳細ウィンドウ (details pane)

選択された項目の詳細 (ある場合) を表示する Windows エクスプローラーのウィンドウの下部にある ウィンドウ。選択された項目の詳細がない場合は、フォルダーの詳細を表示します。たとえば、Windows フォト ギャラリーでは、画像名、ファイルの種類、撮影日、タグ、評価、寸法、ファイル サイズが表示されます。「[プレビュー ウィンドウ](#)」も参照してください。

## ダイアログ ボックス (dialog box)

ユーザーがコマンドを実行したり、ユーザーに質問したり、情報や進行状況フィードバックを提供するためのサブ ウィンドウ。

## ダイアログ ボックス起動ツール (dialog box launcher)

リボンの中で、一部のグループの下部に表示され、グループに関連する機能を設定するダイアログ ボックスを開くボタン。「[リボン](#)」も参照してください。

## ダイアログ ユニット (dialog unit)

DLU ともいい、現在のシステム フォントに基づくレイアウトに使用するデバイスに依存しない測定基準。

## 直接操作 (direct manipulation)

UI (アイコン、コントロール、ナビゲーション要素など) でのユーザーとオブジェクトとの直接の操作。マウスとタッチは、直接操作の一般的な方法です。

## ドッキング ウィンドウ (docked window)

オーナー ウィンドウの端に固定されて表示されたウィンドウ。「[浮動ウィンドウ](#)」も参照してください。

## ドロップダウン矢印 (drop-down arrow)

矢印をクリックすると関連する一覧を表示できることを示す、ドロップダウン リスト、コンボ ボックス、分割ボタン、およびメニュー ボタンに関連付けられた矢印。

## ドロップダウン リスト (drop-down list)

互いに排他的な値の一覧から選択できる一般的な Windows コントロール。リスト ボックスとは異なり、この選択肢の一覧は、通常、非表示になっています。

## E

## 有効解像度 (effective resolution)

現在の解像度 (dpi) 設定によって正規化されたモニターの物理的な解像度。96 dpi では、有効解像度は物理的な解像度と同じですが、他の dpi では、比率を保って拡大縮小が行われます。通常、有効解像度は次の式で計算できます。

$$\text{有効解像度} = \text{物理解像度} \times (96 / \text{現在の dpi 設定})$$

「[相対ピクセル](#)」、「[物理的な解像度](#)」も参照してください。

#### システム特権を持つ管理者 (elevated administrator)

ユーザー アカウント制御では、システム特権を持つ管理者は管理者特権を持っています。管理者は、昇格を行わず、最小限の特権の状態で実行します。必要な場合のみ、承認 UI ダイアログを使用して、管理者をシステム特権を持つ状態に昇格させます。「[保護された管理者](#)」、「[標準ユーザー](#)」も参照してください。

#### 拡張ツールヒント (enhanced tooltip)

ポイントされているコマンドを簡潔に説明するポップアップ ウィンドウ。通常のツールヒントと同様、拡張ツールヒントにはコマンドのショートカット キーが表示されている場合があります。ただし、通常のツールヒントと異なり、補足情報やグラフィックス、ヘルプがあることを示すインジケーターが表示されている場合もあります。また、拡張ツールヒントでは、リッチ テキストや区切り記号を使用することもあります。「[ツールヒント](#)」も参照してください。

#### エラー (error)

問題が発生している状態。「[警告](#)」も参照してください。

#### 展開可能な見出し (expandable headings)

アイテム グループを表示または非表示にするために見出しを展開または折りたたみできる、段階的表示のシェブロンのパターン。「[段階的表示](#)」も参照してください。

#### 拡張選択 (extended selection)

リスト ビューおよびリスト ボックスで、隣接する値または隣接しない値のグループを個々に選択することによって、1つのアイテムの選択を拡張できる選択モード。拡張選択を行うには、ドラッグするか、Shift キーを押しながらクリックまたは Ctrl キーを押しながらクリックします。「[複数選択](#)」も参照してください。

#### F

#### フリック (flick)

画面上で指やペンをすばやく一直線に動かすこと。フリックはジェスチャーとして認識され、ナビゲーションや編集コマンドとして解釈されます。

#### 浮動ウィンドウ (floating window)

画面上のユーザーが望むあらゆる場所に表示できるウィンドウ。「[ドッキング ウィンドウ](#)」も参照してください。

#### ポップアップ (flyout)

一時的に詳細情報を表示するポップアップ ウィンドウ。Windows デスクトップでは、ポップアップは、ガジェットをクリックすると表示され、ポップアップの外側の任意の場所をクリックすると消えます。ポップアップは、ドッキング状態と浮動状態の両方で使用できます。

#### フォローアップ ページ (follow-up page)

ユーザーがフォローアップとして実行する可能性のある関連タスクを表示するためのウィザード ページ。「設定完了」ページの代わりに使用することもあります。

#### フォント (font)

文字の属性セット。

#### 全画面表示 (full screen)

枠のない最大化されたウィンドウ。

## G

### ガジェット (gadget)

ユーザーのデスクトップ上でホストされる簡単なミニアプリケーション。「[サイドバー](#)」も参照してください。

### ギャラリー (gallery)

グラフィカルに表示されたコマンドまたはオプションの一覧。結果ベースのギャラリーでは、コマンドそのものではなく、コマンドまたはオプションの効果が示されます。ギャラリーはラベルを付けたり、グループ化したりできます。たとえば、書式設定オプションをサムネイルギャラリーに表示できます。

### ジェスチャー (gesture)

画面上での指やペンのすばやい動き。コンピューターでは、マウスの移動や入力または描画の操作ではなく、コマンドとして解釈されます。

### "作業の開始" ページ (getting started page)

ウィザードを正常に実行するための前提条件やウィザードの目的について説明するオプションのウィザードページ。

### グラス (glass)

ユーザーが外側のインターフェイスではなくコンテンツや機能に集中できるようにするための、半透明を特徴とするウィンドウ枠オプション。

### グリフ (glyph)

図表または記号画像を言い表すための総称。矢印、シェvron、行頭文字は、Windows で一般的に使用されるグリフです。

### グループ ボックス (group box)

関連するコントロールセットの関係を示す一般的な Windows コントロール。

## H

### 手書き認識 (handwriting recognition)

インクをテキストに変換するソフトウェア。

### ヘルプ (Help)

プライマリ UI よりも詳細な情報を提供するユーザー アシスタンス。通常、メニュー や [ヘルプ] リンクまたはアイコンをクリックしてアクセスし、コンテンツには、操作手順、概念の説明、より視覚に基づいた誘導型のチュートリアルなどのさまざまな形式を使用できます。

### ハイコントラスト モード (high-contrast mode)

前景と後景の視覚要素を極端に対比させる特定の表示設定(白地に黒または黒字に白)。特にアクセシビリティ面で役立ちます。

### ハブ ページ (hub page)

コントロール パネル アイテムのハブ ページは、使用頻度の最も高いタスク(タスクベースのハブ ページを使用)または利用可能なオブジェクト(オブジェクトベースのハブ ページを使用)などの高度なレベルの選択肢を表示する。スポーク ページに移動すると、特定のタスクを実行できます。「[スポーク ページ](#)」も参照してください。

### ハイブリッド ハブ ページ (hybrid hub page)

コントロール パネル アイテムのハイブリッド ハブ ページは、一部のプロパティやコマンドもページ上に直接表示されているハブ ページ。ユーザーがコントロール パネル アイテムを使用してプロパティまたはコマンドにアクセスする可能性が高い場合は、ハイブリッド ハブ ページを使用することを強くお勧めします。

## I

### 即時型のコミット モデル (immediate commit model)

ユーザーが変更を行ってすぐに変更が有効になるハイブリッドハブページで使用されるコミット モデル。このモデルでは、コミット ボタンは使用しません。「[遅延型のコミット モデル](#)」も参照してください。

### インプレース メッセージ (in-place message)

別のウィンドウではなく、現在の UI 画面のコンテキスト内に表示されるメッセージ。別のウィンドウとは異なり、インプレース メッセージでは、利用可能な画面領域か動的なレイアウトが必要になります。

### 間接ダイアログ ボックス (indirect dialog box)

タスクの間接的な結果、またはシステムやバックグラウンド プロセスで問題が発生した結果として、コンテキストとは直接関係なく表示されるダイアログ ボックス。

### 誘導的なユーザー インターフェイス (inductive user interface)

複雑なタスクを、簡単で、容易に説明でき、明らかな目的を持って明確に説明された手順に分解する UI。

### 情報ヒント (infotip)

ポイントされたオブジェクトの簡潔な説明を表示する小さなポップアップ ウィンドウ。説明が表示されるオブジェクトには、ツールバーのコントロール、アイコン、グラフィック、リンク、Windows エクスプローラーのオブジェクト、スタートメニューのアイテム、タスク バーのボタンなどがあります。情報ヒントは段階的表示の形式で表示されるため、情報ヒントを使用すると、画面上に常に説明テキストを表示しておく必要がなくなります。

### インク (ink)

ペンの出力の原料。このデジタルインクは、そのまま手書きとして保持するか、手書き認識ソフトウェアによってテキストに変換できます。

### INLINE

リンクまたはメッセージを、関連する UI のコンテキストに直接配置すること。たとえば、INLINE リンクは、切り離されているのではなく、別のテキスト内に配置されます。

### INPUT FOCUS

ユーザーが現在入力を行うよう指定している場所。UI 内の場所が強調表示されているという理由だけでは、必ずしもこの場所に入力フォーカスがあるということにはなりません。

### INSTANCE

プログラムのセッション。たとえば、Windows Internet Explorer では、一度に複数の独立したセッションを実行できるため、プログラムの複数のインスタンスを実行できることになります。設定はプログラムのセッション間で保存できます。「[値の保持](#)」も参照してください。

J

K

### KEY TIP (keyTip)

リボンにおいて、アクセス キーを表示するためのしくみ。アクセス キーは、アクセス キーの表示に通常使用される下線付きの文字とは異なり、各コマンドやグループをポイントして表示される小さなヒントの形式で表示されます。「[アクセス キー](#)」も参照してください。

L

### LANDSCAPE MODE (landscape mode)

オブジェクトを縦長ではなく横長になるように配置する表示オプション。「[縦長](#)」も参照してください。

## **最小限の特権のユーザー アカウント (least-privilege user account)**

通常は最小限の特権を使って実行するユーザー アカウント。「[ユーザー アカウント制御](#)」も参照してください。

### **リスト ボックス (list box)**

一覧に表示された値セットから選択でき、ドロップダウンリストとは異なり、常に一覧が表示されている一般的な Windows コントロール。単一選択または複数選択がサポートされています。

### **リスト ビュー (list view)**

単一選択または複数選択によるデータ オブジェクトのコレクションの表示と操作が可能な一般的な Windows コントロール。

### **リアルタイムのプレビュー (live preview)**

ユーザーが操作をコミットすることなく、選択時やポイント時にすぐにコマンドの結果を表示するプレビュー技術。たとえば、テーマ、フォント、色などの書式設定オプションは、リアルタイムのプレビューによって、ユーザーに結果を表示する手間を最小限に抑えることができます。

### **ローカライズ (localization)**

さまざまな国、言語、文化、または市場向けにソフトウェアを適応させるプロセス。

### **ログ ファイル (log file)**

コンピューター システム上の活動に関するさまざまな情報のファイルベースのリポジトリ。多くの場合、ログ ファイルの調査は管理者が行い、一般的なユーザーは、通常行いません。

## **M**

### **メイン指示テキスト (main instruction)**

ウィンドウまたはページでの操作を簡潔に説明する、目立つように表示されたテキスト。指示テキストは、具体的な文、必須の指示、または質問である必要があります。優れたメイン指示テキストは、単に UI 操作に注目させるだけではなく、ユーザーの目的を明確にするものになっています。

### **管理環境 (managed environment)**

個人ユーザーではなく、IT 部門やサード パーティのプロバイダーによって管理されるネットワーク コンピューター環境。管理者は、その他のタスクのうち、パフォーマンスの最適化と、オペレーティング システムやアプリケーションの更新プログラムの適用を実行できます。

### **操作 (manipulation)**

入力操作と、タッチされているオブジェクトの実際の操作に対する自然な反応が直接対応している、タッチ操作の種類。

### **最大化 (maximize)**

ウィンドウを最大サイズで表示すること。「[最小化](#)」、「[元に戻されたウィンドウ](#)」も参照してください。

### **メニュー (menu)**

現在のコンテキストでユーザーが使用できるコマンドまたはオプションの一覧。

### **メッセージ ボックス (message box)**

ユーザーに特定の状況に関する情報を伝えるために表示されるサブ ウィンドウ。

### **ミニ ツール バー (mini-toolbar)**

ポイント時にコンテキストに応じて表示されるツール バー。

### **最小化 (minimize)**

ウィンドウを非表示にすること。「[最大化](#)」、「[元に戻されたウィンドウ](#)」も参照してください。

### **混在状態 (mixed state)**

アイテムのグループにチェック ボックスが適用されている場合、混在状態とは、一部のアイテムがオンになっており、その他のアイテムはオフになっていることを示します。

## モーダル (modal)

あるモードでの操作に起因して制限された対話操作。モーダルは、多くの場合、オーナー ウィンドウの対話操作を制限するサブ ウィンドウを指します。「[モードレス](#)」も参照してください。

## モードレス (modeless)

制限されない対話操作。モードレスは、多くの場合、オーナー ウィンドウの対話操作を制限しないサブ ウィンドウを指します。「[モーダル](#)」も参照してください。

## 複数選択 (multiple selection)

リストまたはツリーで複数のオブジェクトを選択できること。

## N

### 重大ではないシステム イベント (non-critical system event)

直ちに対処する必要のない種類のシステム イベント。多くの場合、システムの状態に関連するものです。「[重大](#)」も参照してください。

## 通知 (notification)

ユーザーに簡単に表示される重大ではない情報。通知は、タスク バーの通知領域にあるアイコンからバルーンの形式で表示されます。

## O

### オプトイン (opt in)

ユーザーが明示的にオプションの機能を選択できる機能。ユーザーの希望を推測しないため、特にプライバシー関連やマーケティング関連の機能では、オプトアウトほど面倒ではありません。「[オプトアウト](#)」、「[オプション](#)」も参照してください。

### オプトアウト (opt out)

ユーザーが選択を解除することによって不要な機能を削除できる機能。ユーザーの希望を推測するため、特にプライバシー関連やマーケティング関連の機能では、オプトインよりも面倒です。「[オプトイン](#)」、「[オプション](#)」も参照してください。

### オプション (options)

プログラムのカスタマイズに利用できる選択肢。たとえば、[オプション] ダイアログ ボックスでは、プログラムのオプションの表示や変更を実行できます。「[プロパティ](#)」も参照してください。

## コンテキスト外の UI (out-of-context UI)

ユーザーの現在の操作に直接関連しないポップアップ ウィンドウに表示された UI。たとえば、通知やユーザー アクセス制御の承認 UI はコンテキスト外の UI です。

## 所有されるウィンドウ (owned window)

補助的なタスクの実行に使用されるサブ ウィンドウ。最上位のウィンドウではなく(したがって、タスク バーには表示されません)、オーナー ウィンドウに "所有" されています。たとえば、ほとんどのダイアログ ボックスは所有されるウィンドウになります。「[子ウィンドウ](#)」、「[オーナー ウィンドウ](#)」も参照してください。

## オーナー コントロール (owner control)

ヒント、バルーン、またはポップアップの所有元。たとえば、入力の制約のあるテキスト ボックスでは、バルーンを表示して、ユーザーにこれらの制限について知らせます。この場合、テキスト ボックスはオーナー コントロールと見なされます。

## オーナー ウィンドウ (owner window)

所有されるウィンドウの所有元のウィンドウ。z オーダーは、所有されるウィンドウの下に表示されます。「[所有されるウィンドウ](#)」、「[親ウィンドウ](#)」、「[z オーダー](#)」も参照してください。

## ページ (page)

ウィザード、プロパティ シート、コントロールパネルアイテム、Web サイトなどのタスクベースの UI のナビゲーションの基本単位。ユーザーは、1つのホスト ウィンドウ内のページ間をナビゲーションしてタスクを実行します。「[ページフロー](#)」、「[ウィンドウ](#)」も参照してください。

## ページ フロー (page flow)

ユーザーがタスクを実行するページの集まり。「[ページ](#)」、「[タスク](#)」、「[ウィザード](#)」、「[コントロール パネル](#)」も参照してください。

## ページ領域コントロール (page space control)

階層別に整理されたオブジェクトの集まりを表示および操作できる。ページ領域コントロールはツリー コントロールに似ていますが、外観は若干異なります。主に Windows エクスプローラーで使用されます。

## パレット ウィンドウ (palette window)

ツールバーまたは色、パターン、フォント、フォント属性などのその他の選択肢を表示するモードレスのサブ ウィンドウ。

## パン (pan)

地図や写真などの場面を2次元で直接ドラッグして動かすこと。これは、2つの点でスクロールとは異なります。まず、スクロールされるコンテンツは、通常、1つの主な次元を持ち、多くの場合、その次元に沿ってのみスクロールされます。また、スクロール中のコンテンツは、通常、ユーザーがドラッグするスクロールバーとは逆方向に動いているように見えます。

## ペイン (pane)

ユーザーが移動やサイズ変更を行ったり、非表示にしたり、閉じたりできるウィンドウ内の四角い領域。通常、ペインは親 ウィンドウの側面にドッキングされています。他のペインに隣接していることもありますが、重なっていません。ペインをドッキング解除すると、子 ウィンドウに変換されます。

「[ウィンドウ](#)」も参照してください。

## 親 ウィンドウ (parent window)

子 ウィンドウのコンテナー(コントロールやペインなど)。「[オーナー ウィンドウ](#)」も参照してください。

## ペン (pen)

ポインティング、ジェスチャー、簡単なテキスト入力、自由形式の手書きに使用するスタイラス。ペンには、正確なポインティング、入力、インクでの描画をサポートする細くて滑らかな先端があります。また、オプションのペン ボタン(右クリックの実行に使用)や消しゴム(インクの消去に使用)もあります。

## 値の保持 (persistence)

オブジェクトの状態やプロパティが自動的に保持されるという原則。

## 個人設定 (personalization)

プログラムでのユーザーの個人識別に不可欠なコア エクスペリエンスをカスタマイズすること。これに対し、通常のオプションやプロパティは、プログラムでのユーザーの個人識別に不可欠ではありません。

## ペルソナ (personas)

想像上の人物の詳細な説明。ペルソナは、実際の人物に関して熟知された非常に詳細なデータで構成されています。

## 物理的な解像度 (physical resolution)

コンピューターのモニターのハードウェアで表示できる縦横のピクセル数。

## ポップアップ グループ ボタン (pop-up group button)

リボン内にある、グループ内のすべてのコマンドとオプションをまとめたメニュー ボタン。狭い領域でリボンを表示するために使用します。

## 縦長 (portrait mode)

オブジェクトを横長ではなく縦長になるように配置する表示オプション。「[横長](#)」も参照してください。

## 基本設定 (preferences)

使用しません。代わりに、[オプション](#)または[プロパティ](#)を使用してください。

## プレビュー (preview)

オプションを選択したときに表示されるもの。プレビューは、オプションの一部として、または [プレビュー] または [適用] ボタンで要求したときに静的に表示できます。

## プレビュー ウィンドウ (preview pane)

選択したオブジェクトに関するプレビューやその他のデータを表示するためのウィンドウ ペイン。

## 主コマンド (primary command)

ウィンドウの主な目的を果たす中心をなすアクション。たとえば、[印刷] ダイアログ ボックスでは、"印刷" が主コマンドになります。「[副コマンド](#)」も参照してください。

## メイン ツール バー (primary toolbar)

メニュー バーを使用する必要がなくなるくらい十分にコマンドが網羅されるようにデザインされたコマンドの集まり。「[補助ツール バー](#)」も参照してください。

## メイン ウィンドウ (primary window)

メイン ウィンドウはオーナー ウィンドウを持たず、タスク バーに表示されます。主なプログラム ウィンドウは常にメイン ウィンドウになります。「[サブ ウィンドウ](#)」も参照してください。

## プログラム (program)

コンピューターで実行できる一連の指示。一般的な種類のプログラムには、生産性アプリケーション、消費者向けアプリケーション、ゲーム、キオスク、ユーティリティなどがあります。

## 進行状況バー (progress bar)

特定の操作の進捗をグラフィカルなバーで表示する一般的な Windows コントロール。

## 段階的表示 (progressive disclosure)

使用頻度の少ない情報(通常、データ、オプション、またはコマンド)を必要に応じて表示できる技術。たとえば、他の情報が必要になることがある場合、シェブロン ボタンをクリックしてコンテキスト内に表示できます。

## 段階的なエスカレーション (progressive escalation)

ユーザーに情報を伝えるための UI が、イベントがより重要になるにつれて徐々に目立ってくる過程。たとえば、ユーザーは最初は安全に無視できるイベントに通知を使用できますが、状況が深刻になるにつれて、モーダル ダイアログなどのより目立った UI を使用する必要があります。

## プロンプト (prompt)

テキスト ボックスまたは編集可能なドロップダウン リスト内に既定の値として配置されたラベルまたは短い指示。静的テキストとは違い、ユーザーがコントロールに入力を開始するかテキスト ボックスに入力フォーカスが移動すると、プロンプトは消去されます。

## プロパティ (properties)

ファイル名や読み取り専用の状態などのユーザーが変更できるオブジェクトの設定と、ファイルのサイズや作成日などのユーザーが直接変更できないオブジェクトの属性。通常、プロパティでは、オブジェクトの状態、値、または外観が定義されます。

## 保護された管理者 (protected administrator)

ユーザー アカウント制御で、最小限の特権の状態で実行している管理者。「[システム特権を持つ管理者](#)」、「[標準ユーザー](#)」も参照してください。

## Q

### クイック アクセス ツール バー (Quick Access Toolbar)

使用頻度の高いコマンドを表示する小さなカスタマイズ可能なツール バー。

### クイック起動バー (Quick Launch bar)

[スタート] ボタンの横にあり、ユーザーが選択したプログラムのアイコンが挿入された、Windows デスクトップ上の直接アクセス ポイント。Windows 7 では削除されました。

## R

### ラジオ ボタン (radio button)

互いに排他的な、関連する選択肢のセットから選択できる一般的な Windows コントロール。

### 相対ピクセル (relative pixels)

96 dpi (ドット/インチ) では物理ピクセルでの表示と同じであるが、他の dpi では比率を保って拡大縮小が行われる、デバイスに依存しないメトリック。「[有効解像度](#)」も参照してください。

### 元に戻されたウィンドウ (restored window)

最大化も最小化もされていない、画面の一部に表示されたウィンドウ。「[最大化](#)」、「[最小化](#)」も参照してください。

### リボン (ribbon)

ウィンドウまたは作業領域の上部にあり、場所と高さが固定されている、コマンドとオプションのタブ付きコンテナー。リボンには、通常、アプリケーションメニューとクイック アクセス ツール バーがあります。「[メニュー](#)」、「[ツール バー](#)」も参照してください。

### リスクのある操作 (risky action)

悪い結果になる可能性があり、容易に元に戻せないユーザー操作。リスクのある操作には、コンピューターのセキュリティに障害をもたらしたり、コンピューターへのアクセスに影響を与えたり、誤ってデータを損失する可能性のある操作などがあります。

## S

### 視線の通り道 (scan path)

ユーザーがウィンドウ内で対象を読み取って指定するときにたどる可能性のある経路。特に、ユーザーが深くテキストを読み込んでいない場合に重要です。

### スクリーン リーダー (screen reader)

視覚的要素を音声に変換することによって、視覚障碍のあるユーザーが、ユーザー インターフェイスを解釈およびナビゲートできる支援テクノロジー。これにより、スクリーン リーダーのコンピューター化された音声によって、テキスト、コントロール、メニュー、ツールバー、グラフィックス、およびその他の画面要素が読み上げられます。

### スクロール バー (scroll bar)

ウィンドウのコンテンツを縦または横にスクロールできるコントロール。

### 副コマンド (secondary command)

便利ではあるが、そのウィンドウの目的の本質ではない周辺的な操作。たとえば、[印刷] ダイアログ ボックスでは、"プリンターの検索"、"プリンターのインストール" が副コマンドです。「[主コマンド](#)」も参照してください。

### サブ ウィンドウ (secondary window)

オーナー ウィンドウを持ち、そのためタスク バーに表示されないウィンドウ。「[メイン ウィンドウ](#)」も参照してください。

### セキュリティで保護されたデスクトップ (**secure desktop**)

システム上で実行中のプログラムから分離され、ログオン、パスワードの変更、UAC の昇格 UI などの安全性の高いタスクのセキュリティを向上させるために使用される、セキュリティで保護された環境。「[ユーザー アカウント制御](#)」も参照してください。

### セキュリティ シールド (**security shield**)

#### 選択済み (**selected**)

操作を実行するためにユーザーが選択していること。「強調表示された」ともいいます。

### センテンス スタイルの大文字化 (**sentence-style capitalization**)

センテンス スタイルの大文字化では、以下のこと留意します。

- 新しいセンテンスの最初の語は、常に大文字にする。
- コロンの後の語は、単語が固有名詞ではない場合や、コロンの後のテキストが完全なセンテンスになっていない場合、大文字にしない。
- 全角ダッシュの後の語は、単語が固有名詞ではない場合、ダッシュの後のテキストが完全なセンテンスになっていても、大文字にしない。
- 句点の後の新しいセンテンスの最初の語は、常に大文字にする。大文字と小文字を区別する、小文字の語で始まるセンテンスを書き直します。

### 設定 (**settings**)

プログラムまたはオブジェクトを構成するために(ユーザーによって、または既定で)選択された特定の値。

### ショートカット キー (**shortcut key**)

頻繁に実行する操作にすばやくアクセスするために押すことのできるキーまたはキーの組み合わせ。通常、ショートカット キーの最適な候補は、**Ctrl** キーと文字の組み合わせおよびファンクションキー(**F1** ~ **F12**)です。その名が示すとおり、ショートカット キーは、インターフェイスの他の場所で実行される機能をキーボードで実行するためのものです。したがって、ショートカット キーは、特定の操作にアクセスする唯一の方法として使用しないでください。

アクセシビリティの向上を目的としたアクセス キーとは対照的に、ショートカット キーは主に詳しい知識のあるユーザー向けにデザインされています。ショートカット キーは、UI 自身に直接含まれていないため(メニュー やツールバーのツールヒントに含まれていることはあります)、記憶するためのものであり、したがってアプリケーション内やさまざまなアプリケーション間で一貫して割り当てられる必要があります。

### サイドバー (**Sidebar**)

Windows Vista でガジェットを表示するためのユーザーのデスクトップの側面の領域。「[ガジェット](#)」も参照してください。

### 単一点エラー (**single-point error**)

単一コントロールに関連するユーザー入力エラー。たとえば、間違ったクレジットカード番号の入力は単一点エラーとなり、間違ったログオンは二重点エラーになります。これは、ユーザー名かパスワードのいずれかが問題になっている可能性があるためです。

### スライダー (**slider**)

明るさや音量などの連続した値の候補の範囲から 1 つの値を表示および設定する一般的な Windows コントロール。

### スマート タグ (**smart tag**)

必要に応じて、コンテキスト内に一時的に表示されるボタン(通常、分割ボタン)。たとえば、Microsoft Word では、ユーザーが貼り付けを行った後、すぐに貼り付けオプションを含むスマート タグが表示さ

れます。

### スピン ボックス (spin box)

テキスト ボックスと関連するスピン コントロールの組み合わせ。数値を増減させるには、スピン ボックスの上矢印または下矢印をクリックします。相対量に使用するスライダー コントロールとは異なり、スピン ボックスは正確な既知の数値にのみ使用します。

### スピン コントロール (spin control)

値を変更するためにクリックするコントロール。スピン コントロールでは、上矢印と下矢印で値を増減させます。

### スプラッシュ スクリーン (splash screen)

プログラムの起動中に表示される切り替え効果のスクリーン イメージ。

### 分割ボタン (split button)

メイン ボタンの右端にある下向きの三角形が表示された小さなボタンを含む、2つに分割されたコマンド ボタン。三角形をクリックすると、ドロップダウン メニューにさまざまなコマンドが表示されます。「[コマンド ボタン](#)」も参照してください。

### スpoke ページ (spoke page)

コントロール パネル アイテムで、ユーザーがタスクを実行する場所。スpoke ページには、タスク ページとフォーム ページの2種類があります。タスク ページは、タスクまたは手順と、特定のタスク ベースのメイン指示テキストを表示し、フォーム ページは、全般的なメイン指示テキストを基に、関連するプロパティやタスクのコレクションを表示します。「[ハブ ページ](#)」も参照してください。

### 標準ユーザー (standard user)

ユーザー アカウント制御では、標準ユーザーはコンピューター上で最小限の特権を持ち、管理タスクを実行するために、コンピューター上での管理者のアクセス許可を要求する必要があります。保護された管理者とは逆に、標準ユーザーは昇格することはできません。「[システム特權を持つ管理者](#)」、「[保護された管理者](#)」も参照してください。

### 静的テキスト (static text)

対話型コントロールに含まれていないユーザー インターフェイス テキスト。ラベル、メイン指示テキスト、補足指示テキスト、補足説明などがあります。

### 補足指示テキスト (supplemental instructions)

メイン指示テキストに情報、詳細、またはコンテキストを追加するユーザー インターフェイスのオプション フォーム。「[メイン指示テキスト](#)」も参照してください。

### 補助ツール バー (supplemental toolbar)

メニュー バーと共に動作するようにデザインされたコマンドの集まり。「[メイン ツール バー](#)」も参照してください。

### システム カラー (system color)

GetSysColor アプリケーション プログラミング インターフェイス (API) を使用して、特定の目的のために Windows で定義された色。たとえば、COLOR\_WINDOW ではウィンドウの背景色を定義し、COLOR\_WINDOWTEXT ではウィンドウのテキストの色を定義します。システム カラーは、テーマ カラーほど豊富ではありません。「[テーマ カラー](#)」も参照してください。

### システム メニュー (system menu)

移動、サイズ、最大化、最小化、閉じるなどの基本的なウィンドウ コマンドの集まり。タイトル バーのプログラム アイコンで表示するか、タスク バー ボタンを右クリックして表示します。

T

### タブ付きダイアログ (tabbed dialog)

関連する情報をラベル付きの個別ページ(タブ)に表示するダイアログ ボックス。同様にタブ付きが多

いプロパティ シートとは異なり、タブ付きダイアログ ボックスは、オブジェクトのプロパティの表示には使用しません。「[プロパティ](#)」も参照してください。

## タスク (task)

多くの場合、単一の UI 画面(ダイアログ ボックスなど)や一連のページ(ウィザードなど)で示されるユーザー操作の単位。

## タスク ダイアログ (task dialog)

タスク ダイアログ API を使用して実装されたダイアログ ボックス。Windows Vista® 以降が必要です。

## タスク フロー (task flow)

ウィザード、エクスプローラー、ブラウザーのいずれかでタスクを実行できる一連のページ。

## タスク リンク (task link)

他のページやウィンドウに移動したり、オプションを選択したり、ヘルプを表示するリンクではなく、タスクを開始するためのリンク。

## 作業 ウィンドウ (task pane)

ダイアログ ボックスに似ているが、別ウィンドウではなく、ウィンドウ内に表示される UI の種類。そのため、ダイアログ ボックスよりも直接的で、コンテキストに即している印象を与えます。作業 ウィンドウには、選択されたオブジェクトやプログラム モードに関連する少数のコマンドを表示するメニューが含まれています。

## タスク バー (taskbar)

デスクトップ プレゼンス機能を持つプログラムを実行するためのアクセス ポイント。タスク バー ボタンと呼ばれるコントロールを操作して、プログラム ウィンドウの表示、非表示、および最小化を行います。

## テキスト ボックス (text box)

特にテキストを入力するためにデザインされたコントロール。テキストや数値を表示、入力、または編集できます。

## テーマ カラー (theme color)

GetThemeColor API と構成要素、状態、および色を使用して、特定の目的のために Windows で定義された色。たとえば、ウィンドウの構成要素で FillColor や TextColor を定義します。テーマ カラーはシステム カラーよりも豊富ですが、テーマ サービスが実行されている必要があります。「[システム カラー](#)」も参照してください。

## タイトル スタイルの大文字化 (title-style capitalization)

タイトル スタイルの大文字化では、以下のことに留意します。

- すべての名詞、動詞("is" や "to be" の他の形式など)、副詞("than" や "when" など)、形容詞("this" や "that" など)、および代名詞("its" など)は大文字にする。
- 品詞にかかわらず、最初と最後の単語は大文字にする("The Text to Look For" など)。
- 動詞句("Backing Up Your Disk" など)に含まれる前置詞は大文字にする。
- タイトルの最初の単語でない場合、冠詞("a"、"an"、"the")は、大文字にしない。
- タイトルの最初の単語でない場合、等位接続詞("and"、"but"、"for"、"nor"、"or")は、大文字にしない。
- タイトルの最初の単語でない場合、4 文字以下の前置詞は大文字にしない。
- タイトルの最初の単語でない場合、不定詞句("How to Format Your Disk" など)に含まれる "to" は大文字にしない。
- 複合語の 2 番目の語が、名詞または "e-word" のような固有名形容詞であるか、それぞれの語の重要度が同じである場合("E-Commerce"、"Cross-Reference"、"Pre-Microsoft Software"、"Read/Write Access"、"Run-Time" など)、大文字にする。2 番目の語が、前置詞や他の付属語など別の品詞である場合("Add-in"、"How-to"、"Take-off" など)、大文字にしない。
- 通常は大文字にしないユーザー インターフェイスやアプリケーション プログラミング インターフェイスの

用語は、大文字と小文字を区別する場合 ("The fdisk Command" など) を除いて、大文字にする。プログラミング言語のキーワードやその他の専門用語 ("The printf Function"、 "Using the EVEN and ALIGN Directives" など) は、従来の大文字化に従ってください。

- 各列の見出しの最初の語のみ大文字にする。

## ツールバー (toolbar)

効率的にアクセスするために最適化されたコマンドのグラフィカルな表示。

## ツールヒント (tooltip)

ラベルのないツールバーのコントロールやコマンド ボタンなど、ラベルのないコントロールをポイントしたときに表示される小さなポップアップ ウィンドウ。

## タッチ (touch)

指を使ってコンピューターの表示を直接操作すること。

U

## ユーザビリティ調査 (usability study)

UI デザインをテストし、実際の対象ユーザーからフィードバックを収集することで、ユーザー エクスペリエンスを向上できる調査方法。ユーザビリティ調査には、ユーザビリティ ラボなどで行う正式な方法もあれば、ユーザー自身のオフィスなどで行う非公式な方法もあります。ただし、この調査で一貫しているのは、参加者から情報を収集し、意味のある傾向やパターンに関する情報を評価し、最後に、調査で確認された問題に対処する論理的な変更を実装することです。

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

ユーザー アカウント制御 (UAC、以前の名称は "最小限の特権のユーザー アカウント (LUA)" ) が有効になっている場合、対話型の管理者は、通常は最小限のユーザー特権で実行しますが、自己昇格を行って、承認 UI によって明示的に承認を行うことで管理タスクを実行できます。このような管理タスクには、ソフトウェアやドライバーのインストール、システム全体の設定の変更、他のユーザー アカウントの表示/変更、および管理ツールの実行があります。

## ユーザー アカウント制御シールド (User Account Control shield)

### ユーザー入力の問題 (user input problem)

ユーザー入力が原因のエラー。ユーザー入力の問題は、操作を続行する前に修正する必要があるため、通常、重大ではありません。

### ユーザー シナリオ (user scenario)

特定の環境におけるユーザーの目的、問題、またはタスクの説明。

V

W

## 警告 (warning)

今後問題を引き起こす可能性のある状況について説明するメッセージ。警告はエラーや質問ではありません。Windows Vista 以降では、警告メッセージは、通常、タスク ダイアログで表示され、わかりやすく簡潔なメイン指示テキストを使用し、テキストの表示を強化するために標準的な警告アイコンを表示します。

## ウェルカム ページ (welcome page)

ウィザードの目的を説明するための最初のページ。ウェルカム ページは現在では推奨されていません。このページがなくても、ユーザー エクスペリエンスの効率を高めることができます。

## ウィンドウ (window)

プログラムやコンテンツが表示されるコンピューターの画面上の四角い領域。ウィンドウは移動、サイズ変更、最小化、または閉じることができます。他のウィンドウに重ねることもできます。子ウィンドウをドッキングすると、ペインに変換されます。「ペイン」も参照してください。

## **Windows ロゴ キー (Windows logo key)**

Windows ロゴが表示された修飾キー。このキーは、多くの Windows ショートカットに使用され、Windows の使用のために予約されています。たとえば、Windows ロゴ キーを押すと、Windows のスタートメニューの表示または非表示を行います。

## **ワイヤーフレーム (wireframes)**

ウィンドウの機能やレイアウトを表示し、完成した外観は表示しない UI モックアップ。ワイヤーフレームでは、色、複雑なグラフィックス、またはテーマを使用せず、線分、コントロール、およびテキストのみを使用します。

## **ウィザード (wizard)**

ユーザーが複数の手順から成る、あまり実行しないタスクを実行できる一連のページ。効果的なウィザードでは、別の UI に比べて、タスクを実行するために必要な情報が少なくて済みます。

## **作業領域 (work area)**

作業を実行するほか、プログラム、ドキュメント、およびショートカットを保存できる画面上の領域。[「デスクトップ」](#)も参照してください。

X

Y

Z

## **Z オーダー (Z order)**

表示上のウィンドウの階層関係。