Windows® Internet Explorer® 8

Microsoft

ブラウザー パフォーマンスの測定

ベンチマークとパフォーマンス分析の問題を理解する

2009 年 3 月 公開

最新の情報は http:/www.microsoft.com/ie8 をご覧ください

Version 1.0

概要

この資料では、ブラウザーのさまざまなコンポーネントがベンチマーク時のパフォーマンスにどのように影響するか解説しています。さまざまなベンチマーク ツールの利点と問題点を比較し、それらの影響を受けないテストの設計についても説明します。またこの文書で解説しているテスト プロセスを実行するための、ベンチマーク環境を構築する方法についての概略的な手引も含んでいます。

このドキュメントに記載されている情報は、対象事項に関する発行時における米国 Microsoft Corporation の考えを表しています。マイクロソフトは市場の変化に対応する必要があるため、このドキュメントの内容に関する責任をマイクロソフトは問われないものとします。また、発行日以降に発表される情報の正確性を保証できません。

このホワイト ペーパーは情報提供のみを目的としています。**マイクロソフトは、この文書およびその記載事項について、明示と暗示とを問わずなんら保証を行いません。**

この文書およびソフトウェアを使用する場合は、適用されるすべての著作権関連の法律に従っていただくものとします。このドキュメントのいかなる部分も、米国 Microsoft Corporation の書面による許諾を受けることなく、その目的を問わず、どのような形態であっても、複製または譲渡することは禁じられています。ここでいう形態とは、複写や記録など、電子的な、または物理的なすべての手段を含みます。ただしこれは、著作権法上のお客様の権利を制限するものではありません。

マイクロソフトは、このドキュメントに記載されている事項に関して、特許、申請中特許、商標、著作権、および他の知的財産権を所有する場合があります。マイクロソフトからの書面のライセンス同意書の記載事項に従う場合を除き、このドキュメントを所持することは、その所有者にこれらの特許、商標、著作権、および他の知的財産権を付与するものではありません。

注記がない限り、例として示された会社、組織、製品、ドメイン名、電子メール アドレス、ロゴ、人名、場所、イベントは架空のものであり、実際の会社、組織、製品、ドメイン名、電子メール アドレス、ロゴ、人名、場所、イベントとは無関係です。

Microsoft、Windows、SmartScreen、InPrivate と Internet Explorer は、米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。このドキュメントで使用されている実在の会社名および製品名は、該当各社の商標です。

目次

ブラウザー パフォーマンスの測定 4
現実環境でのテスト 4
テスト ツール 5
キャッシング動作 5
測定する事によるオーバーヘッドとリソース競合 6
ネットワーク接続とネットワーク デバイスの遅延 7
インターネット ベースのデータを取り扱う 7
ブラウザーに特化したコンテンツ 7
コンピュータの構成 8
拡張機能の問題 9
定義の不一致 10
自分で測定を行う 10
手順 1. テストのターゲットを選択する 11
手順 2. システムをセットアップする 11
テストを順番に行う 11
手順 4. ワーム スタートでテストする 11
手順 5. ベンチマークを実行する 12
手順 6. 繰り返す 13
結果を取りまとめる 13

付録 A: ブラウザー パフォーマンス テストの結果 14

ブラウザー パフォーマンスの測定

ユーザーがブラウザーを利用している時間が、コンピュータ上にインストールされている他のどのアプリケーションより最も多いのはおそらく間違いない事でしょう。ブラウザーのベンダーは、製品が安定していてトラブルが無く、かつ素早く動作する事に注力すべきです。ブラウザーのスピードのベンチマークは、口で言うほど簡単ではありません。この資料ではブラウザーのさまざまなコンポーネントがベンチマーク時のパフォーマンスにどのように影響するか解説しています。ベンチマーク ツールの比較も行い、問題の起きないテスト方法を示します。このホワイトペーパーの最後には、ここで解説しているテスト プロセスを実行できるベンチマーク環境を構築する方法についての、概略的な手引も含んでいます。ベンチマーク テストを実行するステップ バイ ステップの解説をすぐに読みたい場合は、“自分で測定を行う” のセクションに直接進んでください。

製品に退行がないか検出するため、私たちは Internet Explorer テスト ラボの管理されたテスト環境で段階検査モデルを使い、毎月毎日ベンチマーク手順を実行しています。このプロセスで、数万の Web サイトを計測しテストする能力だけでなく、パフォーマンスに対する規則的で一貫した視点も得る事ができました。日ごとのテストは付録 A のリストに示したようにおよそ 25 の Web サイトを対象にしていますが、具体的なサイトのリストは毎日変わります。日々のテストに利用する Web サイトはさまざまな理由で選択されますが、選択の基準で重要なのは、現在のインターネット ユーザーの代理となるよう、リストが国際的な Web サイトを含んだ幅広い多様性のある Web サイトを包含する事です。私たちが公開した Internet Explorer 8 のパフォーマンスについてのビデオでは、ユーザーが Web 上で最もよく訪問するサイトを反映させるため、世界的なインターネット情報調査会社である comScore 社の測定した最上位 25 の Web サイトを選択しました。

現実環境でのテスト

実際に “活動中” の Web サイトを使ってブラウザーのパフォーマンスを測定する事は困難です。ネットワークの遅延やネットワークの混雑、テストの対象となる “活動中”の Web サイトのトラフィックの水準など、とても多くの変化する要素が測定に影響を与えます。これに対して人工的な検証環境でのテスト (ベンチマーク) は、管理された設定の下でブラウザー パフォーマンスの特定の側面を測定するように作られています。これらのベンチマークは必然的に、とても不自然な方法によってブラウザーの特定の少数の機能だけで特徴づけられています。エンド ユーザーは管理された環境下で操作していませんし、ブラウズ作業の特定の部分だけを行う事もしません。エンド ユーザーがブラウザーを使う時は、ニュースを読む、経路検索する、オンライン ショッピングするといった全体的な作業を行うのです。

ブラウザーはいくつかのサブシステムから構成されており、それらは全体のパフォーマンスに関してそれぞれの役割を果たしています。エンド ユーザーは JavaScript を実行するだけ、テスト ページを読み込むだけといった使い方はしないので、“マイクロ ベンチマーク” でのテストをエンド ユーザーのエクスペリエンスに直接置き換える事はできないだけでなく、全体的なパフォーマンスを誤って伝える結果になる事もあります。エンド ユーザーにとっては、そうした単

独の操作ではなく、作業の全体を示す総合的な “ページ表示時間” を測定するようなテストの方が、より実際に即しているでしょう。

例えば食料品店で買い物をするといった現実の作業について考えてみましょう。その作業は、それぞれが作業全体を完了させるのに重要な、多くの細かな役割によって占められています。しかし個々の手順の速度は作業全体の速さを正確に示すものではありません。たとえば同じ買い物リストを持った隣同士に住む二人がいるとします。一人はステーション ワゴンで店に向けて出発し、同時にもう一人はスポーツ カーで出発した場合でも、スピードの出る車に乗った人が買い物を早く済ませるわけではありません。店に着いた後に店の通路で迷ってしまい、買い物リストの品物を見つけるのに手間取るかもしれません。会計の時も現金で支払うのに対して小切手で支払うと余計な時間がかかるでしょう。この例が示すように、作業の一部分しか測定しない場合、全体のパフォーマンスについて誤った結論を導き出す事になりかねません。

現実のユーザーにとってのパフォーマンスの基準は、ページがどれだけ素早く表示されるかであり、個々のサブシステムのパフォーマンスではありません。したがってこの資料は Web ページ全体の表示時間の測定方法に焦点を定めて、それに関連する複雑な事項を取り上げます。取り上げる内容はテスト ツールとその制限事項、キャッシングの影響、測定する事によるオーバーヘッドとリソース競合の影響、ネットワーク接続やその他のリソースの制限などです。この資料では前述した項目ついて解説するとともに、それらがページ表示時間の計測するパフォーマンス テストにどのような影響を与えるのかについても述べています。

テスト ツール

多くのベンチマーク テストが SunSpider や Celtic Kane、V8、iBench などのツールの結果を引いています。こうしたツールはどれも、少なくとも何らかの方法でエンド ユーザーでのシナリオの一部についてのみテストするよう設計されています。SunSpider は JavaScript のパフォーマンスと、その他の最近の Web サイトでよく使われる機能を主にテストします。他の定評のあるベンチマークである Celtic Kane と V8 も JavaScript エンジンのテストにのみ向いています。それらとは異なり iBench スイートはページの表示を完了したというブラウザーからの報告に依存した動作をします。ただしページ表示の完了の定義はブラウザーによって異なるため、このテストも信頼性の置けない結果をもたらします。

キャッシング動作

キャッシングは、ユーザーのオンライン エクスペリエンスを最適化するためにブラウザーが利用するよく知られた機能ですが、この機能はベンチマーク テストにも影響を与えます。ブラウザーは同じコンテンツを繰り返しリクエストする事が無いよう、コンテンツをローカルにキャッシュ (保存) するよう設計されています。多くの企業ネットワークやインターネット サービス プロバイダー (IPSs) も同様に、ネットワーク伝送ラインを通じて繰り返しリクエストされるコンテンツの総量を大きく減らすため、キャッシングを利用しています。そのためユーザーは、企業ロゴやその他の一般的なグラフィックスのように複数のページに同じコンテンツが含まれているサイトをブラウズする時、待たされる事がなくなります。

しかしながらこの時間を節約するための設計は、ベンチマーク結果に適切でない影響を与える要因となります。たとえば一つ目のブラウザーの 10 個のタブで 10 の異なる Web サイトを開いた後、二つ目のブラウザーでも同じ 10 個のタブを開くテストをすると、二番目のブラウザーが早いという誤った結果が得られます。実際にはこの違いは、主に最初のブラウザーがページをリクエストした際にコンテンツが近くのサーバーに保存された事により生じています。

> **Internet Explorer テスト ラボの場合**: テストを開始する前に利用するすべてのサイトを訪問しています。テストに先立ってキャッシュを “プリロード” する事でシステムを既知の状態にできます。
>
> **自分でテストする場合:** ネットワーク レベルのキャッシング装置の影響を避ける事はほぼ不可能であるため、実際のテストを開始する前にテストに使用するすべてのブラウザーを使い、テストに使用するすべての Web サイトを訪問する事がひとつの解決策として考えられます。この “プレ キャッシング” 手順により、それぞれのブラウザーをより等しい条件でテストできます。

測定する事によるオーバーヘッドとリソース競合

ベンチマークの別の問題は、測定のためのプロセスがリソースを消費し、測定しようとするアプリケーションに影響を与える事です。こうした動作は “観察者効果” として知られています。たとえばアプリケーションにデバッグ コードを追加すると、プログラムは通常より遅く動作するでしょう。“ハイゼンバグ” として知られている、関連する状況もあります。これは測定のためにコードを変更する行為 (またはバグのテストのため条件を単純化する行為) により、最終的には変数やその他のシステム状態が与える影響を変化させる結果になる事です。

測定のプロセスによって生じるオーバーヘッドを完全に取り除く方法は存在しないため、目標はそれを最小化する事になります。よく使われるアプローチは特定のシナリオに集中したテストを利用する事です。これは一定のパフォーマンスへの影響があるとしても、テストを最小化すれば余分な動作もより少なくなるという理屈です。

ベンチマークでのより大きな問題は、リソースの競合かもしれません。実際のどのシステムでもリソースには限りがあり、すべてのアプリケーションに対して同時に全部のリソースを利用可能にはできません。プロセスはリソースが利用可能になるまで開始できませんが、リソースは常に変化しているため、ベンチマークの結果を一貫性のない物にしてしまう可能性があります。他のアプリケーションの動作もリソースの競合を生むので、これを軽減するか分離する努力が必要です。

> **Internet Explorer テスト ラボの場合:** 不要なアプリケーションとサービスを無効にしています。
>
> **自分でテストする場合:** 効果的な構成となるよう、テスト中は他のアプリケーションを無効にすべきです。

ネットワーク接続とネットワーク デバイスの遅延

ネットワーク接続は絶えず増大しているのにネットワーク リソースは無限ではないという別の問題もあります。ローカル エリアでの輻輳、基幹線の輻輳、ピアリングの輻輳などの現象は、ネットワーク リソースの可用性に影響を与えます。ネットワーク層で動作しているアプリケーション (たとえばプロキシ サーバーやロード バランサー、ファイアウォールなど) にベンチマークのデータに影響を与えるリソース競合の問題が存在している事もあります。

> **Internet Explorer テスト ラボの場合:** テスト用ネットワークでの多くの問題を軽減するため、テストでは DTAP を使用しています。
>
> **自分でテストする場合:** こうした問題を完全に取り除く事はできないため、最良の方法は毎回同じような負荷の状態でテストして平均的な読み込み時間を確認する事です。

インターネット ベースのデータを取り扱う

インターネット ベースの (“活動中の”) Web サーバーを使ったテストには、パフォーマンス データの観測に影響する要素があります。単純に言えば、サーバーがどのように動作し、アプリケーションがどのように設計されているのかによってベンチマークのプロセスに影響がでます。幸いな事に、アプリケーションの問題の多くは分離するか完全に取り除く事ができます。しかしながら、サーバーの問題は同じ方法で取り除く事ができません。

> **Internet Explorer テスト ラボの場合:** 時間をおいて何回かのテストを行い、統計的な平均を得るため異常値を除外しています。
>
> **自分でテストする場合: こうした問題に対抗する最良の解決策は、一定期間何度もテストを実行し、テスト期間中ネットワーク層のリソースの状態を一定に保つようにする事です。**

ブラウザーに特化したコンテンツ

多くの Web サイトはユーザーが利用しているブラウザーに合わせて可能な限り最良のエクスペリエンスを提供するため、ブラウザーの検出を行っています。これはあるブラウザーと別のブラウザーで Web サイトが異なる動作をするという事ですが、クライアントのブラウザーに応じて異なるコンテンツを提供しているという事でもあります。こうした変化する動作はベンチマークにエラーを発生させる事があり、またあるブラウザーでのコンテンツの表示結果を別のブラウザーでの結果と直接比較できないという変則的な状態を発生させます。CSS やレイアウトの少しの違いは避けがたい事が多々ありますが、おおむねは許容範囲です。しかし大きな相違は避けなければなりません。ベンチマーク テストに先立って、Web サイトが異なるブラウザーで見たとき大きく異なっていないか確認するため、それぞれの Web サイトを目視で調査する事が重要です。

> **Internet Explorer テスト ラボの場合:** テストを開始する前に、テストに使用するそれぞれのブラウザーを使ってリストにある Web サイトを調査しています。より一貫して同じ

条件で比較できるよう、大きく異なって見える Web サイトはテスト用リストから除外されます。

自分でテストする場合: ベンチマーク テストに利用する Web サイトのリストを作成します。テストするブラウザーごとにそれぞれの Web サイトを訪問し、外見やレイアウトに大きな違いが出ていない事を確認するためよく見て調べます。

コンピュータの構成

人間と同じように、2 台のコンピュータが厳密に同じになる事はありません。この点についての実例ですが、Internet Explorer のパフォーマンス検証環境には日々刻々パフォーマンス テストを実行している大量のコンピュータがあります。検証環境を最大限に柔軟にすべく、私たちは交換しても首尾一貫したパフォーマンス データのセットを作り出せるような、“まったく同一”のコンピュータをひと揃い作ろうと企てました。これらのコンピュータは連番のシリアル ナンバーが付けられており、同じ組み立てラインで生産され、それぞれのコンポーネントの部品は“まったく同一” でした。

この点について考える上で重要なのは、ハードウェア ベンダーはコストを管理しつつ十分な供給を確保するため、幅広い供給元からパーツを調達している事です。たとえばイーサネット ポートのような一般的なコンポーネントの場合、多ければ 3 つの供給元から調達しているでしょう。それぞれの製品が適切な技術仕様に適合している限り、それらのパーツは交換可能なものとして採用されます。カテゴリー 5 ケーブルが他のどんなカテゴリー 5 ケーブルとも置き換えできるのと同様、イーサネット ポートはイーサネット ポートです。双方が規定された技術標準に適合しているなら、両方とも同じように動作するはずです。問題は、デバイス ドライバーのような若干の相違がそれ自身のパフォーマンス上の問題を引き起こす事です。すべてのシステム コンポーネントを確実に完全に同じにできる簡単でスケーラブルな方法はないので、最良の方法は異なったコンピュータから得られた結果をできるだけ比較しない事です。

こうした理由により、システムを複製しようとした努力にもかかわらず Internet Explorer のパフォーマンス検証環境で収集したデータは極めてまちまちで、2 台の異なったコンピュータから得られたパフォーマンスの結果は比較しない方が良いとわかりました。

Internet Explorer テスト ラボの場合: 一連のベンチマーク テストを開始する前に、コンピュータができる限り初期状態に戻るようイメージから復元しています。

自分でテストする場合: ベンチマーク テストのためのコンピュータをセットアップする場合、パフォーマンスに影響を与えないよう他のアプリケーションをインストールせず、既定の設定を保ちます。一貫性を確保するため、すべてのテストで同じコンピュータを使用します。

拡張機能の問題

拡張機能、あるいはアドオンの問題も、ベンチマークを混乱させる可能性があるもう一つの要素です。アドオンはユーザーにブラウジング エクスペリエンスを向上させる強力な手段を提供しますが、パフォーマンス上の問題を引き起こす可能性もあり、ブラウザー “本来の” 問題とサードパーティーのコードによって生じる問題の区別を困難にしています。Mozilla のチーフ エバンジェリストである Mike Shaver はあるインタビューでこの問題について、“Firefox を利用している人の中には大量の拡張機能を組み込んで「以前よりパフォーマンスが悪くなった」とか「クラッシュするようになった」と言う人がいるんだ” と明らかにしています。[1]

この問題を考慮するには、どのようなアドオンもない “クリーン” なインストール状態と、一般的によく使われているアドオン (Adobe Flash や Microsoft Silverlight®、 Windows Media® Player など) を有効にした状態の両方でテストする事が重要です。この二つのデータを比較する事で、サードパーティーの要素の影響が見えてくるでしょう。

Internet Explorer 8 でアドオンを無効にするには、**ツール**メニューから**アドオンの管理**を選択し、**アドオンの管理**ダイアログ ボックスを表示します。**すべてのアドオン**が表示されている事を確認し、アドオンをすべて無効にします (コマンド プロンプトで `iexplore.exe -extoff` を実行する方法もあります)。またはすべてのアドオンを無効にする “セーフ モード “ でブラウザーを起動する事もできます。このモードでの起動メニューはスタート メニューの**アクセサリ > システム ツール**にあります。他のブラウザーでもセーフ モードと同様の操作ができるものがあり、異なるブラウザーを通じてアドオンなしでのベンチマークが容易に行えます。

ブラウザー	セーフ モードを有効にする
Internet Explorer	**スタート > すべてのプログラム > アクセサリ > システム ツール > Internet Explorer (アドオンなし)**
Firefox	**スタート > Mozilla Firefox > Mozilla Firefox (セーフ モード)**

Internet Explorer テスト ラボの場合: 実際のユーザー エクスペリエンスを確認するため、一般的な Web サイトで広く使われている機能に必要な、よく使われているアドオン (Flash、Silverlight と Windows Media Player) をインストールしてテストしています。それと並行してデータを計測するため、どのアドオンもインストールしないテストも行っています。

自分でテストする場合: ブラウザーにインストールされているアドオンの一覧を作り、どのブラウザーでも同じか確認します。テストしようとするブラウザーのいずれかで利

[1] Scott Swigart and Sean Campbell. How Software Is Built. 2008 年 3 月 20 日 http://howsoftwareisbuilt.com/2008/03/20/interview-with-mike-shaver-chief-evangelist-mozilla/

用できない (または同等でない) アドオンは削除するか無効にします。さらに、ベンチマーク データの比較のためテストはアドオンなしでも実行すべきです。

定義の不一致

何をもってタスクの完了とするのか一貫した基準を定める、ということがベンチマークに影響を与える別の要因として存在します。馬や自動車の競争ではそのゴールは明白です。そこには決勝線があり、すべての競争の参加者はレースを終えるために決勝線を越える必要があります。ブラウザーのベンチマークでも、Web ページの読み込み完了を定義する同様の方法が必要です。問題は、ブラウザーはいつページの読み込みの “完了” を報告できるのか、または報告すべきなのかについて、標準が存在しない事です。どのブラウザーもそれぞれ異なった方法でこれを扱っているため、この指標に従うと結果は非常に変動的になってしまいます。

最良のアプローチは、インジケーターが実際に Web ページが読み込まれ利用可能になった時にどのようになるか確認した上で、ブラウザーの進行状況のインジケーターを利用する事です。たとえば特定の Web ページがどれだけ早く読み込めるかテストする場合、ページが最初に読み込まれる際に、特定のオブジェクトを観察するかページに対する操作を試してみます。進行状況のインジケーターが完了を示す前に Web ページか読み込まれたように見え操作可能になっているのなら、測定ではインジケーターは無視してページの外見を利用するのが合理的です。ページが利用可能で無ければ、進行状況のインジケーターはさまざまなブラウザー間でページのダウンロードの速さを初期的に評価するために十分役立ちます。

> **Internet Explorer テスト ラボの場合:** “完了” のステータス通知を利用した場合の矛盾を回避するため、ページが完了した事を判定するのに視覚的な状態での完了を指標としています。特徴的な視覚的指標が十分で無いサイトでは、ページが “完了” しているかどうか判断するため視覚的な指標とページ上のコンテンツが操作可能かどうかの組み合わせを利用しています。AJAX やそれに類する技術を利用して設計されているサイトでは “完了” の判定はより複雑になるため、視覚的な状態での完了を指標とするのが確実な方法です。
>
> **自分でテストする場合:** ブラウザー自身のステータス情報は参照せず、自分自身のエクスペリエンスと応答を信頼します。ページが操作可能で利用可能に見えるのであれば、その状態になるタイミングのデータを利用可能と判断してよいでしょう。ページが非常に単純化されていて (たとえば数個のオブジェクトだけがあるようなページ) ページが利用可能になった事を確認するのが困難な場合は、ページ上の視覚的な要素を見つけてそれが読み込まれた時にページが完了したとみなす方法が簡単でしょう。

自分で測定を行う

ベンチマークの複雑な問題とテストでの落とし穴を避ける方法が理解できたら、ベンチマーク用のテスト ベッドを構築しましょう。

Internet Explorer テスト ラボの場合: すべてのテストを監視するためにビデオ録画システムを利用しています。ビデオをフレーム単位にスロー再生できるので、測定の指標を読み取って非常に精密に時間的パフォーマンスを捉えられます。

自分でテストする場合: Internet Explorer ラボとそっくり同じテストをおこなうのは現実的ではないので、安定して評価を完了するため以下の手順で基本的な環境を構築します。

手順 1. テストのターゲットを選択する

テスト対象の選択を多様にします。 人気のある Web サイトのリストを用意するのは重要です。しかしそれらのサイトがコンテンツの内容でも機能でも似すぎていると、ブラウザーのサブシステムを完全にテストする事はできないでしょう。広範でいろいろな要素が混じり合ったテスト対象を用意する事で、SunSpider などの多くのマイクロ ベンチマークのように一つのサブシステムだけではなく、ブラウザーの異なるすべてのサブシステムについて確実なテストが行えます。私たちがテストに使用したサイトのリストは付録 A に含まれています。

テストを進める前に、テスト対象ごとにユーザーの操作手順を明確にし、十分に理解しておきます。実際のテスト プロセス中にサイトが利用可能かどうか十分に判断できるよう、テスト対象のサイトに通じている必要があります。つまりユーザーがサイトへナビゲートでき、操作できる必要があります。テストが特定の視覚的な指標に基づいて進められるのであれば、指標となるアイテムが認識できるよう十分に確認しておきます。

手順 2. システムをセットアップする

キャッシュをクリアします。 テストを開始する前に、どのブラウザーもクリーンな状態にする事が極めて重要です。ブラウザーごとにキャッシュをクリアする方法は少し異なっています。Internet Explorer の場合は、**セーフティ**メニューから**閲覧の履歴の削除**を選択します。この機能では一つのメニューで複数のアイテムが削除できますが、**インターネット一時ファイル**だけ選択するようにしてください。これでどのブラウザーも同じ状態になります。

他のアプリケーションを停止します。 開いている他のアプリケーションを閉じ、システム トレイに格納されているアプリケーションも確認します。可能なものについてはシステム トレイにあるすべてを閉じます。(CTRL+SHIFT+ESC を押して) **Windows タスク マネージャ**を起動し、アプリケーション タブのタスクを (テスト済みのブラウザーを含めて) 選択して**タスクの終了**をクリックします。

テストを順番に行う

ブラウザーを一つ選択します。 並行してテストを進めない事が重要です。二つのブラウザーを同時に動作させてテスト対象の Web サイトにアクセスしてはいけません。これをしてしまうといずれかまたは両方のブラウザーに影響を与え、結果を歪めてしまいます。

手順 4. ワーム スタートでテストする

ブラウザーを読み込みます。 ブラウザーを起動し、キャッシュが用意され、さまざまなブラウザーのコンポーネントが読み込まれるよう、テストのために選択した対象の Web サイトを訪問

します。これにより部品の読み込みによるパフォーマンスへの影響や余分な遅延を抑える事ができます。先に行っているブラウザーのテスト サイクルが完了するまで、他のブラウザーのウォーム スタートは行わずに待ちます。

手順 5. ベンチマークを実行する

ビデオ記録装置をセット アップします。正確なタイミング情報を捉える事は詳細なブラウザー パフォーマンスの比較において非常に重要です。コンピュータのビデオ出力のキャプチャーや単独のカムコーダーなどの外部的なビデオ録画装置を使用すると、ベンチマーク結果に作用するシステムのオーバーヘッドや悪影響を回避でき、後から結果を比較するために記録をスロー再生できます。Internet Explorer チームはベンチマークのタイミングを記録するためコンピュータの出力から直接ビデオ キャプチャーしています。スクリーンが明瞭に見える位置にデバイスをセット アップし、画面全体が取り込めるようフレームを決めます。ビデオ イメージを可能な限り大きく鮮明にすると、画面に表示される小さなオブジェクトでも識別できるようになります。ビデオ キャプチャーが完了したら、よりきめ細かなタイミング情報を確認するためメディア プレーヤー ソフトウェアを使って再生します。Internet Explorer チームはフレーム単位[2]で再生できるメディア プレーヤー ソフトウェアを利用して、おおよそ 30 分の 1 秒単位の精度でタイミング データを得ています。Windows Media Player でフレーム単位の再生をするには、**表示**メニュー、**拡張設定**、**再生速度の設定**を順に選択します。これで 1 フレームごとの再生が可能なコントローラーが表示されます。他のメディア プレーヤー ソフトウェア アプリケーションでもフレーム単位の再生をサポートしているので、Windows Media Player の代わりに好みのソフトウェアを使う事ができます。

> **タイミングを計る別の選択肢: ストップウォッチを使います。**ビデオ記録装置を用意できない場合は、タイミング データを捉えるためストップウォッチを使う事も可能です。しかしながらストップウォッチを使う方法は潜在的にタイミングの遅延と不正確さをもたらします。つまりその結果の信頼性は高くなく、可能な限りこの方法は使うべきではありません。ストップウォッチを使う方法の場合、“個人誤差” (PE) として知られる少しの遅延が発生します。これは目が事象を捉えてから手がストップウォッチを押すまでの反応時間です。個人誤差は多くの人で約 0.4 秒なので、それぞれの結果からこれを除外する必要があります。

テストを開始します。Web サイトのリストを使い、ブラウザーのアドレスバーに URL を入力します。ストップウォッチを準備して、ENTER を押したら計時を開始します。Web ページの読み込み完了を報告する方法についてのブラウザーごとの不一致を避けるため、視覚的な標識が表示されたか、またはサイトがユーザー操作可能になった瞬間のどちらかを “完了” 時刻として記録します。たとえばサイトにテキスト入力ボックスがある場合、“完了” とはユーザーがテキストを入力可能になった時を意味します。

複数回測定します。内部的および外部的要素が測定に影響を与える可能性があるため、リスト上の Web サイトのそれぞれについて 10 回ずつテストする事が重要です。テストが完了したら、上位 2 つと下位 2 つの結果を除外し、残りの平均を求めます。

[2] NTSC ビデオは 1 秒あたり 30 フレーム (30 fps).記録します。

手順 6. 繰り返す

他のブラウザーをテストします。テストするそれぞれのブラウザーごとに手順 4. と手順 5. を繰り返します。

後でもう一度テストします。一日の中の時間帯の違いによって生じるネットワーク トラフィックや利用度の相違の影響による問題を避けるため、すべてのテストは別の時間帯で再度実行するべきです。理想的には一回目のテストは通常の業務時間帯内に実行し、二回目はそのおよそ 12 時間後に実行します。通常の業務時間帯は所定の Web サイトがターゲットとしているユーザーのそれに合わせるべきです。

結果を取りまとめる

この分析で説明したように、ベンチマークには多くの要素が複雑に入り組んでおり、有効な結果に到達するには複雑な手順が必要です。この資料ではベンチマーク テストを実施し、ブラウザーのパフォーマンスの正しい分析のために外部的な要素を分離するのに必要な情報を提供しました。広く知られているベンチマークツールには限界があります。それを理解する事は、ブラウザー サブシステムのすべての要素を評価するような総合的なレベルのパフォーマンスや、より全体的なページのロード時間に着目することの重要性を理解する手助けとなるでしょう。

付録 A: ブラウザー パフォーマンス テストの結果

私たちが公開した Internet Explorer 8 のパフォーマンス ビデオでは、ユーザーが最も良く訪れるサイトを反映するように comScore の測定による世界で最上位 25 の Web サイトを選びました。下に掲げる表は Google Chrome と Mozilla Firefox、Internet Explorer 8 で行った 25 の Web サイトでの一連のベンチマーク テストの結果を示しています。

先に記したように、マイクロソフトは毎月数万の候補から日々のテストごとにおよそ 25 の Web サイトを選んでいます。ベンチマークに利用する Web サイトのリストには、ユーザーのインターネットでのエクスペリエンスであるパフォーマンスの全体像が描けるよう、国際的な Web サイトを含む多様な Web サイトが含まれるべきです。

ブラウザーの読み込み時間の概要（単位：秒）				
順位	Web サイト	Chrome 1.0	Firefox 3.05	Internet Explorer 8
1	google.com	0.28	0.22	0.20
2	yahoo.com	2.15	1.30	1.32
3	live.com	3.48	3.42	3.40
4	msn.com	0.55	0.97	0.83
5	youtube.com	3.07	2.80	2.55
6	microsoft.com	3.83	3.47	3.75
7	wikipedia.org	1.22	1.35	1.88
8	blogger.com	2.07	2.88	1.52
9	facebook.com	2.12	2.08	1.98
10	qq.com	6.82	7.88	7.33
11	baidu.com	1.40	1.47	1.50
12	myspace.com	2.13	5.53	3.68
13	wordpress.com	0.95	1.58	1.45
14	ebay.com	4.06	4.43	3.42
15	sina.com.cn	5.48	6.37	8.03
16	mozilla.com	1.28	1.53	1.27
17	adobe.com	9.50	9.37	8.85
18	aol.com	3.02	2.57	2.32
19	amazon.com	2.12	3.35	2.82
20	apple.com	2.52	3.07	1.63
21	soso.com	3.25	2.64	2.07
22	xunlei.com	8.60	9.97	8.70
23	163.com	14.87	14.75	15.02
24	google.cn	1.38	0.85	1.05
25	ask.com	2.17	1.77	1.73

この資料で示したテスト ガイドラインに沿って、ページの読み込みが完了したタイミングの指標としてブラウザーの “完了” インジケーターを使用しました。“完了” インジケーターが表示された後も読み込みと変化が続くページについては、タイミングとして一般的な視覚的標識を使用しました。時間の計測は**移動**ボタンが押された時に開始しています。上記の結果は 2009 年 1 月に計測しました。インターネット上のコンテンツは常に変化しているため、同じテストを行っても結果が異なる場合があります。