キャッシュと同期の推奨設定

イントロダクション

このドキュメントでは、Curl RTE のパフォーマンスを最大限に活用できるように Curl RTE の設定と Web ブラウザのキャッシュの特徴について説明します。
この設定を採用することで、正確なプログラムの動作を維持しながら、Curl アプリケーションにおいて最も効果的なキャッシュと同期を行うことができます。

推奨設定の概要

このセクションでは、推奨する設定についてそれぞれ簡単に説明します。詳細はその後のセクションで述べられています。

サーバーの設定

サーバーと開発用マシンの時刻設定の同期をとってください。

Curl のコンテンツの開発や配信に関わるすべてのマシンに時刻同期ソフトウェアを実行し、すべてのマシンが正確な時刻を維持できるようにしてください。また、クライアントマシンも同様に同期させておくとよいでしょう。

HTTP サーバーでは、アプレットの起動ファイルの有効期限を短く設定し、その他のファイルにはこの設定を行わないでください。

HTTP サーバーでは、必要に応じてアプレットの起動ファイルにおけるキャッシュの有効期限(以下、有効期限)短く設定してください。ただし、Curl のアプレットで参照するその他のファイル（例えば pcurl ファイル、画像ファイルなど）はキャッシュの有効期限を設定しないでください。

ファイル変更時刻の管理

Curl のファイルをディプロイする時に、ファイルの更新日時は未来の日時にしないでください。

Curl のコンテンツファイルを HTTP サーバーにディプロイした時に、サーバーの時刻に対して未来のファイル更新日時にしないでください。前述の設定（マシンの時刻設定の同期）がなされていれば、この様な問題は起こらないでしょう。

ミラーサーバーにある同じファイルの更新日時も必ず一致させてください。

一つの HTTP アドレスからアクセス可能な複数の HTTP サーバーに Curl のコンテンツが配布される場合は、全てのサーバーにあるファイルが同一のファイル更新日時を保持するようにしてください。例えば、ロードバランサーなどの負荷分散機を利用して、複数台のサーバから同じ Curl アプレットのコンテンツを提供する場合がこのケースにあたります。

変更されていないファイルの更新日時は、サーバに配布する際に変更されないようにしてください。

ファイル コンテンツが変更されない場合は、サーバー上にある、そのファイルの更新日時の更新を避けてください。

ファイルが更新された場合、そのファイルの更新日時を古くすることを避けてください。

HTTP サーバー上のファイルを、古い更新日時に変更されたものと入れ替えないようにしてください。更新日時を”更新”する(新しくする)ことをお勧めします。

アプレット

applet 宣言の中で resync-as-of を利用してください。

HTTP サーバー上にディプロイした最新版コンテンツの更新日時以降となる日時を指定して、resync-as-of 宣言を applet に追加してください。

resync-as-of を設定する場合は、クライアントの時刻設定との差異を考慮してください。

resync-as-of の設定はクライアントマシンの設定時刻に基づいて解釈されるので、今後クライアントとサーバー間で起こりえる時間差を考慮して resync-as-of に指定する日時

を調整してください。たとえば、クライアントの時刻がサーバーの時刻より２時間遅れることが考えられる場合、サーバーにディプロイした更新日時に２時間を追加して resync-as-of に指定してください。

アプレットの起動ファイルのサイズを最小化してください。

再ロード時のダウンロード負荷を最小限にするために、最初に読み込まれるファイルのサイズを可能な限り小さくしてください。代わりに、コンテンツはパッケージや、そのファイルがインクルードするファイルに含めてください。

クライアントの設定

クライアント側のコントロールパネルにある強制再同期オプションに依存しないでください。

クライアントのコントロールパネル設定を制御する方法がなく、またそれらの設定はいつでもエンドユーザーによって変更できます。そのため、Curl RTE のコントロール パネルにある「全てのアプレットの再同期を強制する。」のオプション設定を使用しても、効果的なキャッシュと同期を実現することはできません。その代わりにアプレットに適切な resync-as-of 宣言を使用してください。

推奨の詳細

サーバーの設定

以下のことを推奨します。

・ サーバーと開発用マシンの時刻設定を同期させてください。

・ HTTP サーバーでは、アプレットの起動ファイルのキャッシュ有効期限を短く設定し、その他のファイルにはこの設定を行わないでください。

Curl のコンテンツの開発、ディプロイ、配信に関わるすべてのサーバーと開発用マシンの時刻設定を常に同期させた状態にしてください。Curl RTE の同期メカニズムは Web ブラウザのメカニズムと同様に、静的なコンテンツのファイル変更時刻に依存しています。

開発およびディプロイメントのためのマシンの日時を常に同期させることが重要です。それらのマシンのファイル更新日時が、HTTP サーバーにディプロイされるファイルの更新日時に影響を与えるからです。また、サーバーマシン同士を常に同期させることも重要です。そのことが低レベルの HTTP のキャッシュ動作に影響を与えるからです。サーバーと開発マシンを同期させておくことで、以下に説明するような未来のファイル更新日時によって間違ったキャッシュの結果を引き起こす可能性を軽減することができます。

マシンを同期させる適切な方法はプラットフォームによって異なりますが、その方法は難しくありません。Windows では、[日付と時刻] のコントロール パネルで時刻の同期を設定できます。サーバーマシンをタイム サーバーとして設定しますが、ファイアウォールが外部のタイム サーバーに NTP の要求を許可しておく必要があるかもしれません。Linux 上では、NTP サービスのセットアップが必要となります。

HTTP サーバーで、アプレットの起動ファイルの有効期限を短く設定することを推奨します。ただし、アプレットの起動ファイルから直接または間接的に呼び出される更新の必要がないその他のファイルに対しては、この設定はしないでください。
アプレットの resync-as-of 宣言によって、Curl RTE がロードするファイルに関しては同期されているか確認できますが、アプレットの起動ファイル自体の同期は Web ブラウザによって行われているため、Curl RTE は制御できません。エンドユーザーは [Ctrl] キー（IE の場合）または [Shift] キー（Netscape/Mozilla の場合）を押しながら、再ロード ボタン（通常は F5 キーに割り当てられています）を押すことにより、ブラウザで強制同期ができますが、これにはエンドユーザーがいつ強制同期をすべきか知っている必要があります。有効期限が切れる時期はアプリケーションの性質によって異なりますが、アプレットが非常に大きい場合や、ユーザーが変更にすぐに気づく必要がない場合を除き、有効期限がすぐに切れるよう設定することが理にかなっています。アプレットが使用する他のファイルに有効期限を設定すると不必要な HTTP の同期が発生するので、推奨できません。

ファイル変更時刻の管理

・未来のファイル更新日時で Curl のファイルをディプロイしないでください。

・ミラーサーバー間の同一ファイルは、ファイル更新日時も必ず同期させてください。

・ディプロイメントの際に、変更されていないファイルの更新日時が変更されないようにしてください。

・ファイル更新日時を過去に戻さないようにしてください。

最終更新日時が重要なファイル属性となります。これが Curl RTE と、基礎となる HTTP レイヤの中で静的に生成されたコンテンツのキャッシュと同期を制御します。サーバー上のコンテンツを配信する際に、Web ブラウザによってファイルの最終更新日時が HTTP ヘッダーの Last-Modified に設定されます。この値はファイルのコンテンツをキャッシュするときは HTTP キャッシュによって保存され、また、Curl のパッケージをキャッシュするときは Curl RTE によって保存されます。この振る舞いはファイルによって異なります。ファイル更新日時が変わると、コンテンツが変更され再ロードが必要であると解釈されます。

HTTP の標準では、サーバーが未来日時の Last-Modified ヘッダーを返さないようにし、未来日時は現在の日時に変換するように規定されています。これにより、サーバーに未来日時のファイルを置いた場合、サーバーの設定日時が現在の日時となるまで、サーバーからファイルがロードされるたびに異なる Last-Modified となってしまいます。Curl のパッケージやマニフェスト ファイルでこれが起こると、Curl RTE がコンテンツに実際にはなかった変更があったとみなし、不必要な再コンパイルが実行されます。これは、大きなアプリケーションではかなり負担となります。したがって、このような状況は絶対に避けるべきです。前述の推奨にあるように Curl のコンテンツの開発、ディプロイ、及び配信に関わるマシンがお互いに同期されていれば、このようなことが起こる可能性は低いでしょう。

同様の理由で、すべてのミラーサーバー上の同じファイルが同じ更新日時であることが非常に重要となります。そうでない場合、クライアントがひとつのサーバーからファイルを読み込み、その後別のサーバーから同じファイルを読み込むと、 Last-Modified が異なるためにクライアントではファイルが期限切れだと判断して、ファイルの再ロードや、それに依存するすべてのコンポーネントの再構築を要求します。これでは、負荷分散のために複数設置されている HTTP サーバーからロードする利点がなくなります。

Curl パッケージの不必要な再ロードや再コンパイルを最小限に抑えるために、サーバーに配置されたときから変更のかかっていないファイルに対しては、異なる更新日時にしないようにしてください。

また、サーバー上のファイルを、現在の更新日時よりも古い更新日時のファイルと置き換えないようにしてください。クライアントがそれらのファイルを適切に更新しない可能性があります。HTTP プロトコルの設計は、 Last-Modified の日時が修正される直前のファイルの更新日時よりも先に進むことを前提としています。したがって、ほとんどの HTTP クライアントとサーバーは、Last-Modified の日時が以前の日時に変更されても、コンテンツを更新しないように実装されています。このような場合、Web ブラウザと Curl RTE は、クライアント上のブラウザ HTTP キャッシュがクリアされるまでサーバー上の変更に気づかない可能性があります。

アプレット

・ アプレットで resync-as-of 宣言を使用してください。

・ resync-as-of を設定する場合は、クライアントとサーバー間の時刻設定の差異を考慮してください。

・ 最初に読み込まれるファイル のサイズを最小限にしてください。

メインのアプレットの起動ファイルがブラウザによってロードされ、そのコンテンツが Curl RTE に渡されると、アプレットで使用される残りのファイルの同期は、以下に示すようなアプレットのヘッダーの resync-as-of 宣言を使用して制御することができます。

```
{curl 5.0 applet}
{applet
    manifest = "manifest.mcurl",
    resync-as-of = {utc-date-time "2007-01-16 12:34 +0900"}
}
```

resync-as-of 宣言は、クライアントマシン上で設定されている日時に基づいて、アプレットで使用されるすべてのファイルとキャッシュされたコンテンツ（たとえば Curl パッケージとマニフェスト）を同期するよう、Curl RTE に指示します。すべてのファイルと他のキャッシュされたコンテンツが resync-as-of で指定された日時より後に同期をとりクライアントにキャッシュされた場合、Curl RTE はアプリケーション自体に要求されない限り、サーバーとそれ以上の通信を必要としません。resync-as-of 宣言に指定する日時が、変更されたコンテンツがはじめて Web サーバー上に配置される日時より後であるこ

とが重要です。また、サーバーの時刻設定とクライアントの時刻設定の差を考慮することも重要です。たとえば、クライアントの時刻設定が４時間サーバーからずれている場合、resync-as-of 宣言の日時に少なくとも４時間を加えてください。(この場合のクライアントとサーバーの時刻設定の違いとは、タイムゾーンでの違いではなく、マシンの協定世界時、又はグリニッジ標準時の定義について言及しています。)

アプレットでの resync-as-of 宣言を省略し、代わりにアプレットのマニフェスト ファイルで宣言することも可能ですが、より多くの設定が必要で、効率が悪いので推奨しません。

前述のように、アプレットの起動ファイルの有効期限を短く設定することが求められるので、ダウンロードの負荷を最小限に抑えるためには、ファイルのサイズを小さくしておく必要があります。そして、多くの機能はパッケージの中に含めてください。パッケージはバイナリ形式でキャッシュされるので、2 回目以降のアプレットのロード時間が短縮されます。アプレットの宣言以下に記述されているコンテンツを、他のファイルに分散し、そのアプレットでインクルードすることも可能です。それにより、アプレットが再ロードされても再同期の必要がない場合は、メインのアプレットの起動ファイルだけは Web ブラウザによってサーバーから取得される必要がありますが、その他のファイルはクライアントの HTTP キャッシュから取得できます。

クライアントの設定

・ クライアント側のコントロール パネルの強制再同期設定オプションに依存しないでください。

一般的に、クライアントのコントロールパネルの設定は既定のままにしておくことをお勧めします。Curl RTE は、コントロールパネルのオプションに強制同期を提供しています。しかしながら、ディプロイされた Curl のアプリケーションがこの設定に依存することを推奨しません。アプリケーションを提供する側が、すべてのクライアントでこの同期設定がなされているかを確かめる方法がなく、エンドユーザーは容易に設定を変えることができてしまいます。クライアントの設定を変更することなく、アプリケーションを提供する側が必要に応じて resync-as-of 宣言を変更できるので、アプレットに resync-as-of 宣言を記述することを推奨します。アプリケーションを提供する側が、アプレットを修正する際に、実際のディプロイメント日時に近い値を resync-as-of の日時に設定したくない場合、代わりに将来の日時を設定することができます。これにより、再ロードのたびに強制的に同期が行われます。

また、Curl RTE のバージョン 4.0 と 5.0 には既知のバグがあります。コントロールパネルで「全てのアプレットの再同期を強制する。」を指定すると、バックグラウンドでパッケージのキャッシュを行う際に余分な同期が実行されてしまい、パフォーマンスの低下を招きます。この問題は、4.0.4 パッチと 5.0.2 パッチのリリースにより解決されました。

デバッグ

ツール

HTTP 監視

・Fiddler HTTP監視ツール

Fiddler は Windows のための HTTP 監視ツール です。Fiddler は IE で使えるよう設定されていますが、設定を変えることで Firefox でも使用可能です。

・LiveHTTPHeaders FireFox拡張

Fiddler ほど有用ではありませんが、FierFox に統合されています。Windows 上では、Curl は HTTP 通信レイヤを使用しており、このツールで Curl からの HTTP 要求をキャプチャーすることができます。

リファレンス

HTTP プロトコル

・RFC 2616: HTTP/1.1 (pdf version)

・W3C HTTP Protocol Homepage

HTTP サーバー

Apache

・Apache HTTP Server Project

・Apache 2.2 HTTP Documentation

・Apache 2.2 HTTP Caching Guide

・Apache Bug Database