### ソフトウェアテストシンポジウム2007東京

Apache JMeterで 負荷試験をしよう

水野浩典 コアテクノロジー部 コンサルティング・インテグレーショ 日本ヒューレットパッカード株式会



### 2007年1月30日

© 2006 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice





### • 負荷試験の重要性

- Apache JMeter
  - 概要、入門、実践編
- 負荷試験方法
  - プロジェクトにおける負荷試験の実例
  - 負荷試験方法のセオリー





# 負荷試験の重要性

- ・必要な理由
  - 負荷試験は、単体テストや結合テストでは発見することが出来なかった問題を見つけるために絶対に必要
- ・必要な理由(その2)
  - 想定外の負荷が掛かることも考慮し、想定外の負荷試験を実施す ることも重要
- ・本当の理由
  - システムのカットオーバー後の夜中や休日に、緊急に呼び出されない ようにするため!





# 現実に発生している問題

- ・負荷試験が、そもそも計画に入っていない。
- 納期に間に合わないので、機能試験はしたが負荷試験まで出来なかった。
- 負荷試験を実施するには、商用製品が必要なようだが、予算的に買えない。
- 試験ツールの使い方が良く分からない。
- とりあえず、負荷試験をしてみたが、試験 結果の評価がよく分からない?
- 以前に負荷試験をしてみたが、大変だったので出来ればやりたくない。

経験によるところも多いのですが、 負荷試験の基本的な方法につい て御説明致します。



下さい。

負荷試験の重要性をプロジェクト

内で認識し、実施するようにして

Freeで使用できるApache JMeterの使用を検討して下さい。

今から、御説明致します。

# Apache JMeter概要





# JMeterとは

### JMeterはJakartaプロジェクトが開発しているパフォーマンス計 測/負荷テストツール

### ● オープンソース、 無償

- 100% pure JavaなのでWindows、Linuxなど、実行環境に依存しない
- GUIを用いて操作できる
- 大量のクライアントからのリクエスの生成/実行が可能
- HTTP(HTTPS)、データベース、FTP、WEBサービスなどの様々なプロ トコルでの負荷試験やパフォーマンス検証が可能
- エラー発生の有無、期待したデータがレスポンスに含まれているかの 確認なども簡単に行うことが可能
- Webブラウザ上で行った操作をテストのシナリオとして記録し、それを テスト・スクリプトとして繰り返し利用、簡単に編集可能





# JMeterの動作概要

- 複数クライアントの実行
  - Javaのスレッドを用いて、複数クライアントの同時実行を実現
- •実行
  - JMeterにテストプランを設定して、これを実行する
- ・テストプラン
  - •手動で作成するか、プロキシ機能を用いて自動で作成
  - ・生成するスレッド数(クライアント数)と実行回数を指定



# Apache JMeter入門





# JMeterのインストール

### ファイルをダウンロードし、適当なディレクトリに展開 するだけです。

1. JMeterのダウンロード Jakartaサイトの以下ページより最新のJMeterバイナリをダウンロード <u>http://jakarta.apache.org/site/binindex.cgi#jmeter\_binaries</u> 最新版は JMeter2.2です。 ここではjakarta-jmeter-2.2.zipをダウンロード

JMeterのインストール
 ダウンロードしたzip / tarファイルをJMeterをインストールしたいディレクトリに展開





### JMeterの起動

JMeterを展開してできたディレクトリにあるbinディレクトリのJMeter起動ファイル jmeter.bat をダブルクリックして下さい。

jmeter.bat

正しく起動されていれば、コマンドプロンプトと以下の通りの画面が表示されます。

🔁 Apache JMeter	
ファイル 編集 実行 オブション ヘルブ	<b>—</b>
テスト計画	
名前 ワークベンチ	

<u>テスト計画</u>

実際のテストシナリオを作成する場所

<u>ワークベンチ</u>

テストに使用しない要素を一時的保管 HTTPプロキシを動作させる場所





1.スレッドグループの作成

### JMeterのテスト実行計画を作成する場合、まずユーザ(スレッドグループ)を作成します。

テスト計画を選択して、右クリックします。 追加 スレッドグループを選択します。

🔁 Apache JMeter	
ファイル 編集 実行 オプション ヘルブ	わかりやすい名前を付けます。
<ul> <li>         ・ マスト計画         ・ Webテスト         ・ Webテスト         ・ サンプラーエラー後のアクション         ・ 後行 ○ スレッド停止 ○ テスト停止         ・ ズレッドプロパティ         スレッド数 1         Ramp-Up 期間(秒): 1         ループ回数 □ 無限ループ 1         こ スケジューラ         ・     </li> </ul>	◆ 実施すべきテストプランに沿った スレッド数(ユーザ数)等の設定をします。



# 2. デフォルトHTTPリクエストの作成

スレッド<u>共通のHTTPリクエストのデフォルト値</u>を設定エレメントの作成より設定します。 その他 設定すべき設定エレメントがあれば設定します。

作成したスレッドグループを選択して、右クリックします。 追加 設定エレメント HTTPリクエスト初期値設定を選択します。

😼 Apache JMeter		
ファイル 編集 実行 オプション	ヘルプ	
<ul> <li>マークベンチ</li> </ul>	HTTP リクエスト初期値設定         名前:       Webサーバ設定         プロトコル:       http         サーバ名またはIP:       127.0.0.1         パス:	━━━ テスト対象のサーバ名/IPを定義します
12 半版19年2月14日	<u>名前:</u> ( ▲ ▲	



isvent

# 3.サンプラーの作成

#### 実際にHTTPリクエスト処理を行うものとして、サンプラーを作成します。

作成したスレッドグループを選択して、右クリックします。 追加 サンプラー HTTPリクエストを選択します。

😼 Apache JMeter		
ファイル 編集 実行 オプション イ	ヽルブ	わかりやすい名前を付けます。
<ul> <li>マネト計画</li> <li>マレンジン</li> <li>マレンジン</li> <li>ワークベンチ</li> </ul>	HTTP リクエスト 名前: トッブページ Web サーバ サーバ名または IP: ポート番号: HTTP リクエスト	- アクセス対象は、設定エレメントで設定
	プロトコル メソッド: ● GET ○ POST パス: /index.jsp □ 自動リダイレクト ビ リダイレクトに対応 ☑ Keeph	してます。 ここでは、実際のHTTPリクエストの パス等のみ設定します。



# 4.リスナーの作成

#### <u>テスト結果の表示と保存</u>のためにリスナーを追加します。 リスナーは、各々のテストプランにおいて必要なものを設定します。

#### 作成したスレッドグループを選択して、右クリックします。 追加 リスナー 統計レポートを選択します。





# 5. テストプランの保存

### テストプランを実行する前に、 テストプランの保存をします。

メニューのファイル テスト計画を保存 または、ファイル テスト計画に名前をつけて保存を選択します。





# 6. テストプランの実行

### テストプラン実行開始の準備が完了後、テストプランの実行を開始します。

メニューの実行 開始を選択、または、「Ctrl<sub>1+</sub>'R<sub>1</sub>を押します。







# 7.実行結果の確認

### テストが終了したらリスナーにおいて、テスト実行結果を確認します。

### 追加 リスナー 統計レポートを選択します。

🍒 mytest-simple.jmx (C:¥home)	¥hmizuno¥java¥jmeter¥exec¥jakarta-jmeter-2.1.1¥bin¥mytest-simple.jmx) – Apach 🔳 🗖 🔀
ファイル 編集 実行 オプション	ヘルプ
♀- 益 テスト計画	統計レポート
♀ ■ スレットクループ	
→ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■	
	全てのデータをファイルに出力
→ Wiyベージ 一 💽 統計 レポート	ファイル名 参照 □ ログエラーのみ Configure
└─ ∭ ワークベンチ	URL # Sampl Average Median 90% Li Min Max Error % Throughput KB/sec
	トップページ 3 203 200 211 200 211 0.00% 3.0/sec 29.10
	Myページ 3 200 200 201 200 201 0.00% 3.0/sec 60.55
	合計 6 202 200 211 200 211 0.00% 5.0/sec 74.30
	↓ 天1」 ↓ 平均応答時間 ↓ 最小・最大応答 ↓ スループット
	(msec) 」 「 時間(msec) 」





# Apache JMeter実践編







# JMeterのチューニング タイマ テストプランの自動生成



# JMeterのチューニング



JMeterで数多くのスレッド(クライアント)を実行すると、容量不足になり以下のような エラーが発生することがあります。

ERROR - jmeter.threads.JMeterThread: Test failed! java.lang.OutOfMemoryError: Java heap space

### JMeterのチューニングが必要です!!

- ・起動スクリプト(\$JMETER\_HOME/bin/jmeter.bat)を編集します。
- ・Javaヒープの大きさを大きくして下さい。

テストとマシンのスペックに応じて チューニングします。

set HEAP=-Xms256m -Xmx256m







●JMeterはリクエストを送信するとき、各リクエスを連続して送信します。 ●タイマを利用することで、各リクエストの送信の間に間隔を入れることが出 来ます。

左フレームの『スレッドグループ』を選択した後、右クリックまたは『編集』メニューか 追加 タイマを選択して、「定数タイマ」や「定数スループットタイマ」等を追加します。

「定数タイマ」は、待ち時間をミリ秒で指定します。

「定数スループットタイマ」は、1分間に送出するリクエスト数で指定します。 例)60を設定すると1秒間に1つのリクエストが送出されます。

🍃 Apache JMeter	
ファイル 編集 実行 オプション	・ ヘルプ
<ul> <li>マ シスト計画</li> <li>マ シスト計画</li> <li>マ シスレッドグループ</li> <li>定数タイマ</li> <li>HTTP リクエスト</li> <li>第 ワークベンチ</li> </ul>	定数タイマ 名前: <sup>定数タイマ</sup> スレッド遅延時間(ミリ秒): 3000



# タイマを使用したときの実行イメージ







リクエスト初期値設定

統計レポート



スレッド





24







# 負荷試験









負荷試験を実施する時のフローの概要







# プロジェクトにおける負荷試験の実例

・Aさんは、下記のようなシステム用件で負荷試験をすることになりました。

- ・応答時間は2秒以内
- ・最大同時ユーザー数は100人
- ・試験シナリオ

・商品購入シナリオ(ログイン、商品検索、カートに商品を入れる、購入、ログアウト) ・試験環境を構築し、JMeterのシナリオを準備しました。

### 試験を実施するにあたりAさんは以下のように考えました

・最大同時ユーザが100人なので、JMeterは100スレッドで試験しよう。
・あまり多く実行すると怖いから、とりあえず1スレッドにつき1シナリオ実行しよう!
・そして、全てのリクエストの応答時間が2秒以内であればシステム要件を満たすことになる。
・これで僕の仕事も終了だ!







結果

応答時間がとんでもない値になってしまいました。 キャッシュが原因かなぁ!? もう一回実施しよう! ぁれ? 変わらないなぁ? ぁと、もう一回実施する。うんん......変わらないなぁ! やっぱ、フリーのJMeterはだめだなぁ!

待ってください! JMeterは 悪くありません。

なぜ、このようなことになって しまったのでしょうか?





# Aさんが実施した試験

- Aさんはタイマーを使用せずに、Rump-up時間の設定
   も行いませんでした。
- どのような試験を実施したことになるのでしょうか?
   100人が同時に、商品購入のシナリオを1回実施
  - その上100人が、各操作を必死で休み無く実行
- これってどんな負荷?
  - 人気商品を特定の時刻に売り出した時のような負荷





# Aさんが実施した試験の概要







# Aさんの試験の改善方法







# 負荷試験方法のセオリー

ところで...

実際の試験では、何スレッドで試験をすれば良いのでしょうか?

シナリオは何回実施すれば良いのでしょうか?

よほど自信がある場合以外は、以下のように小さな負荷から試 験を実施して下さい。

### 1.想定負荷の検討

2.基礎データの取得

3.基礎試験(低負荷試験)

4. 想定負荷の負荷試験

5. 高負荷試験





# 1. 想定負荷の検討

実際のプロジェクトなどでは、性能の目標値が無かったり、不明確であることが多々あります。

-実例

・最大ユーザー数は100人、応答時間は、2秒以内 同時ユーザ数は?
 ・ユーザー数は100人、秒間50個の処理 50個の単位は?

### ・最低、確認すべき項目

-ユーザー数はシステムに登録される人数?それとも同時にシステムにアク セスする人数?

•JMeterで試験をする場合のスレッド数とは、同時アクセス数に対応します。

-スループットの単位はHTTPリクエスト? それともシナリオ単位?

•JMeterでの試験結果の集計は、HTTPリクエスト単位で行われます。







# 2. 基礎データの取得

- ・1ユーザがシナリオを1回だけ実行した時のリクエストの応答時 間を測定します。
  - 1.1つのブラウザからシナリオを実行した時の応答時間の測定
  - 2. JMeterから1スレッド、1回のシナリオ、リクエストの間隔1秒で実施し、各 リクエストの応答時間を計測
- ・何が分かるの?
  - システムにおける最高の応答時間が分かります。
  - 1の試験の結果が、応答時間の要件を満たしていない場合

•これ以上試験をする必要はありません。早速、遅い原因を調査して下さい。

- 2の試験の結果が、1の試験結果と大きく異なっている場合は、作成した シナリオの内容やJMeterの設定などを確認してください。





# 3. 基礎試験(低負荷試験)

 いよいよ負荷をかけていきます。あせる気持ちを抑えて以下の 順番で試験を実施しましょう。

### ・試験A:1多重複数回(1スレッド、100シナリオ)

- 応答時間が、だんだん悪くなるようなことはありませんか?
- スループットを記録して下さい。
- ・試験B:2多重複数回(2スレッド、100シナリオ)
  - 平均応答時間は、どの程度悪くなりましたか?
  - スループットは、試験Aの2倍ぐらいの値になっていますか?
- ・試験C:4多重複数回(4スレッド、100シナリオ)
  - 平均応答時間は、どの程度悪くなりましたか?
  - スループットは、試験Aの4倍ぐらいの値になっていますか?





# 基礎試験って?

•疑問?

- 試験A、試験B、試験Cを実施しましたが、これらの試験は何のために 行ったのでしょうか?
- 次にどのような試験をすれば良いのでしょうか?
- ・基礎試験を実施した意味
  - -システムの限界性能に対して、試験Aから試験Cによる負荷がどの程度 の負荷になっているかを見るために実施しています。

### • 分析方法

-システムが限界性能に達したときに、スループットの値や応答時間の値 が一般的にどのように変化するかを知ることによって、試験Aから試験C の負荷が限界性能に対してどの程度の負荷であるかを予想することが 出来ます。





# 典型的な負荷試験の結果







## 試験結果の例







# 負荷試験のコツ

### ・シナリオの作成

- ・各リクエストの間には間隔を入れる
- ・各スレッドの開始時間をずらす
- 基礎データの取得
  - ・最初に、ブラウザから実行する
  - ・1スレッドでシナリオを1回だけ実行して、各リクエストの応答時間が要件を満たしていることを確認する
- ・基礎試験の実施
  - ・1スレッド複数回の実行
  - ・2スレッド、4スレッドと、徐々に負荷をかける
- 実施している負荷試験が、限界性能に対してどの程度の負荷を与えているかを把握しながら試験を実施して下さい。





# 負荷試験のコツ:その2

早めの負荷試験のすすめ

- ・出来れば、プロトタイプの作成時や、開発時から負荷試験を実施することをお勧めします。
- ・単体機能の負荷試験も可能です。





# まとめ

- 負荷試験の重要性
- Apache JMeter
  - 概要、入門、実践編
- 負荷試験方法
  - プロジェクトにおける負荷試験の実例
  - 負荷試験方法のセオリー













– Apache JMeter

http://jakarta.apache.org/jmeter/

– Apache Jakaruta Project : JMeter User Manual

http://jakarta.apache.org/jmeter/usermanual/index.html

- Apache Jakaruta Project : JMeter ユーザマニュアル (日 本語訳) <u>http://cgi0.biwa.ne.jp/~yabuta/study/jmeter/usermanual/</u>



- 月刊JavaWorld (ジャバ ワールド) 2005年12月号







Apache



### http://www.hp.com/jp/linux/





エレメント:スレッドグループ

・クライアントの設定

・全てのテストプランはこれから始まります。

この下にサンプラーやリクエストといっ
 たエレメントを追加していくことで様々
 な設定を行います。

🍃 Apache JMeter		
ファイル 編集 実行 オプシ	ヨン ヘルブ	
♥ よテスト計画 スレッドグループ □ スレッドグループ □ ワークペンチ	スレッドグループ 名前: スレッドグルーブ サンプラーエラー後のアクション ● 続行 ○ スレッド停止 ○ テスト停止 スレッドプロパティ スレッド数: 1 Ramp-Up 期間(初): 1 ループ回数: □ 無限ループ 1 □ スケジューラ	



#### 名前:テストケース名

サンプラーエラー後のアクション: サンプラー実行中にエラーが発生した際のアクション 続行(デフォルト)/スレッド停止/テスト停止

スレッド数:(デフォルト1) JMeterが生成するクライアント数(アクセスユーザ数)

Ramp-Up 期間(秒): (デフォルト1) 全てのスレッドが起動するまでの時間

ループ回数: (デフォルト1) 各スレッドにて実行するテストケースの回数

スケジューラ:チェックを入れると時間指定などの 詳細な設定が可能(デフォルトOFF)





# プロキシサーバの使用

- ・ファイヤーウォールやプロキシサーバ越しにテストする場合
- ファイヤーウォールやプロキシサーバの名前やポート情報をJMeterの起動コマンドに与えて起動します。

jmeter.bat –H [プロキシサーバのホスト名] –P [ポート] –u [ユーザ名] –p [パスワード]



タイマ





# タイマを『スレッドグループ』に付け加えるか、あるいは『HTTPリクエスト』等に付け加えるかによって、そのタイマの影響の範囲が異なります。



#### <u> ペープ。に 定数タイマ。をつけた例</u>

この場合は、'スレッドグループ』以下の全てのサンプラー (この例では、'リクエストA」と'リクエストB」)との間で、 タイマによって指定された分だけ待機時間が発生します。

したがって、右の例の場合は、以下のように動作します。 実行 10秒待機 リクエストA 10秒待機 リクエストB

<ul> <li> <del>フ</del>スト計画 <b>マ</b> スレッドグループ <b>ご</b>数タイマ <b>マ</b> リクエストA <b>マ</b> レポートA      </li> </ul>	定数なイフ
<ul> <li>         ・ グリクエストB         ・ レポートB         ・ ロボートB         ・ ロー 道 ワークベンチ         ・         ・         ・</li></ul>	A 30. 定数タイマ スレッド遅延時間(ミリ秒): 10000

# HTTP認証マネージャ



### HTTP認証マネージャは、<u>HTTP認証(Basic認証)</u>を利用できます。

### 追加 設定エレメント HTTP認証マネージャを選択します。

😼 Apache JMeter				×
ファイル 編集 実行 オプション	ヘルプ			
<ul> <li>♀ 盗 テスト計画</li> <li>♀ ▼ スレッドグループ</li> <li>✔ HTTP リクエスト</li> <li>〒 統計レポート</li> </ul>	HTTP 認証マネージャ 名前: HTTP 認証マネージャ 「認証マネージャに保存されて	こいる認証		
→ ₩ HTTP 認証マネージャ ● ① ワークベンチ	基底URL http://localhost/admin	ユーザー名 admin	パスワード *****	Transfer and the second s

例えば、http://localhost/admin/にHTTPのBasic認証がかけられているとします。また、その認証IDとパスワードがそれぞれ"admin"と"admin"だとします。 その場合は、上記のような設定すれば、HTTPのBasic認証をパスすることができます。

「パスワード」の部分は自動的に伏字になります







### 3.動的リクエストの利用

事前に登録しておいた<u>複数のユーザをスレッド起動毎に動的に切り替えたい場合</u>に、 ユーザ変数を利用すると大変便利です。 1人のユーザではなく、複数のリクエストパラメータとしてユーザを用いて負荷試験を する際に使用します。

パラメータ情報を記述した外部ファイルを読み込ませ、動的にユーザパラメータを切り替える方法を使います。

手順としては、 ユーザパラメータと外部ファイルの利用となります。

この手順は、JMeterのFAQで紹介されています。詳しくは、下記URL先をご参考下さい。

#### **JMeterFAQ**

How do I use external data files to in my Test scripts?

http://wiki.apache.org/jakarta-jmeter/JMeterFAQ#head-1680863678257fbcb85bd97351860eb0049f19ae









### <u>手順1.JMeterの起動を起動して、スレッドグループを作成します。</u>



### <u>手順2.スレッドグループを選択し、前処理 ユーザパラメータを追加します。</u> 『変数の追加』を選択して、変数を設定します。

### 記述形式:\${\_StringFromFile(ファイル名)}

		S Apache .INeter		
夕益		ファイル 編集 実行 オブション	n#1	
石則	<u> </u>	● 品 テスト計画 ● 第一ストッドグループ	ユーザーパラメータ	
user id	\${ StringFromFile(userid txt)}	D 2-4-45x-9	名称 ユーザーバラメータ	
		III 0+0428	そ 触り返しごとに要約 パラメータ	
password	\${ StringFromFile(password.txt)}		·····································	2-9-1 St. StringFromFile(userid b0)
1			password	Br. StringFromFile/password.Mb)
			27 800 2018	







invent

### <u>手順3.スレッドグループを選択し、サンプラー HTTP リクエストを追加します。</u> リクエストで送るパラメータの『追加』を選択して、変数を設定します。









### <u>手順4. \$JMETER\_HOME/bin以下にパラメータを記述したファイルを配置します。</u> \$JMETER\_HOMEとは、JMeterをインストールしたフォルダを示します。

\_ファイルの中身は、テキスト形式で以下のような記述となります。日本語もOKです。

user_id.txt	password.txt
taro	pass1
iiro	pass2
saburo	pass3

#### <u>手順5.テストを実行します。</u>

ファイルの内容は、スレッド起動毎に上から1行づつ読み込み、最後まで行くと最初のパラメータ を再度読み込みます。

上の例では、

- START 1回目:taro/pass1
  - 2回目: jiro/pass2
  - 3回目:saburo/pass3
  - 4回目:taro/pass1(最初のパラメータ)





Apach

### <u>手順1.JMeterの起動を起動して、スレッドグループを作成します。</u>

まず「テスト計画」の中にスレッドグループを作成します。 「テスト計画」を右クリックし、追加 スレッドグループを選択します。 「テスト計画」の下に「スレッドグループ」が追加されます。

🈼 Apache JMeter		
ファイル 編集 実行 オプシ	ヨン ヘルプ	
♀ ふ テスト計画 ○ スレッドグループ □ ワークベンチ	スレッドグループ         名前: スレッドグループ         サンプラーエラー後のアクション         ● 続行 ○ スレッド停止 ○ テスト停止         スレッドプロパティ         スレッド数 1         Ramp-Up 期間(秒): 1         ループ回数: □ 無限ループ 1         □ スケジューラ	





Apach

### 手順2. HTTPリクエスト初期値設定エレメント

共通設定がある場合は、スレッドグループにあらかじめ追加しておきます。

スレッドグループ上で右クリックした後、追加 設定エレメントより HTTPリクエスト初期値設定を追加します。

😼 Apache JMeter			
ファイル 編集 実行 オプション	ヘルプ		Űr
♀ 🦾 テスト計画 ♀ 🐨 Webテスト	HTTP リクエスト初期	植設定	
- 🎽 Webサーバ設定	名前: Webサーバ設定		
- (罰 ウークベンチ	プロトコル http		00000
	サーバ名または IP: 127.0.0.	l)	100000
	NZ:		00000
	ポート番号: 80		00000
		リクエストで送るパラ	メータ:
	名前:	値	E
			•





Apach

### <u>手順3.HTTPプロキシサーバエレメント</u>

ワークベンチ上で右クリックした後、追加 Non Testエレメントより HTTPプロキシサーバを追加します。

🍃 Apache J	Meter		
ファイル 編	集 実行 オプション ・	ルプ	
<b>२</b> 👗 न्द्राः — 💽 द्रा	†画 √ッドグループ	ワークベンチ を許 ロークベンチ	
	追加	▶ ロジックコントローラ ▶	
	<b>カット</b> Cf コピー Cf	× サンプラー ▶ -C 設定エレメント ▶	
	ペースト ci 挿入としてペースト	-∨ Non-Testエレメント → HTTP プロキシサーバ	
	開く 併合(マージ) 別名で保存 Save As Image Ci	-6	





Apach

### <u>手順3. HTTPプロキシサーバエレメント</u>

#### ポートの指定

JMeterがプロキシサーバとしてリクエストを受け付けるポート番号を指定します。(デフォルトは8080番)

### <u>テスト計画に挿入するパターンと、除外するパターンの指定</u>

テスト計画に必要なURLと、除外したいURLのパターンを指定します。 URLの指定は、Perl形式の正規表現を用います。

	HTTP プロキシサーバ 名前: HTTP ブロキシサーバ ポート: 8080
[ テスト計画に挿入するURLパターンの例 ] *index¥.jsp	Grouping: Do not group samplers ▼ 挿入するパターン <sup>*</sup> Ljsp
<b>[テスト計画から除外するURLパターンの例]</b> 画像ファイル :*¥.gif や *¥.jpg や *¥.png など CSSファイル :*¥.css Javaスクリプト :*¥.js	追加 静脉
	開始 停止 リスタート



Apach

### <u>手順4.Webブラウザのプロキシサーバの設定をします。</u>

JMeterがインストールされているマシンのブラウザのプロキシサーバの設定をします。 例えば、Internet Explorerであれば、ツール インターネットオプション 設定 LANの設定より行えます。 入力欄「ポート」にはHTTPプロキシサーバ詳細設定画面で入力したのと同じポート番号を指定してください。

#### [Internet Explorerのプロキシ設定]

<ul> <li>□ーカル エリア ネットワーク (LAN) の設定</li> <li>② ×</li> <li>自動構成         <ul> <li>自動構成にすると、手動による設定事項を上書きする場合があります。手動による設定事項を確保するには、自動構成を使用不可にしてください。</li> <li>□ 設定を自動的に検出する (A)</li> <li>□ 自動構成スクリプトを使用する (S)</li> <li>アドレス(E):</li> </ul> </li> </ul>	■LANにプロキシサーバを使用するにチェックを入れます。 ■アドレスは、自分自身のアドレス(127.0.0.1)
プロキシ サーバー 【AN にプロキシ サーバーを使用する これらの設定はダイヤルアップまたは VPN 接続には適用されません)⊗ アドレス(E): 127.0.0.1 ポート(T): 8080 詳細設定(C) □ ローカル アドレスにはプロキシ サーバーを使用しない(B)	「■ポートは、HTTPプロキシサーバで設定した値を入れます。 ここでは、8080番





Apach

### 手順5.テスト計画の記録

HTTPプロキシサーバ画面の「開始」ボタンを押すと、JMeterがプロキシサーバとして 動作し始めます。

HTTP ブロキシサーバ
名前 HTTP コロキシサーバ
ポート: 8080 Capture HTTP Headers ビ Set Keep-Alive Add Assertions
Target Controlle ワークペンチ>HTTP プロキシサーパ ▼
Grouping: Do not roup samplers 🔻
「挿入するパターン
挿入するパターン
.*t,jsp
造加
「除外するパターン
除外するパターン
.ħ.jpg
."Lgif
追加 削除





#### 手順5.テスト計画の記録

手順4にてプロキシの設定を行ったブラウザにて、 実際にテスト対象ページを閲覧、操作します。





1.URLにミスがあったり余分なページを閲覧してしまった場合は、 手動で修正や削除が可能です。

2. Webブラウザによる巡回が終わったら、JMeterの 「HTTPプロキシサーバ」で[停止]ボタンをクリックして記録を終了します。 もし、停止する前にブラウザを閉じてしまうとその動作も記録されてしまいます。





Apach



#### 手順5.テスト計画の記録

### <u>『プロキシサーバ』を停止させ、テスト計画が記録されていることを確認します。</u>

『プロキシサーバ』の起動後から「停止」ボタンを押し『プロキシサーバ』を停止します。 テスト計画の『スレッドグループ』以下に結果が組み込まれているのが確認できます。

# − ト: 8080 I Capture HTTP Headers I Set Keep-Alive Adv Assertions
Target Controller: ワークベンチ > HTTP ブロキシサーパ ▼
Grouping: Do not group samplers 👻
挿入するパターン
挿入するバターン
追加 所除
除外するパターン
"Njpg The off
- syn
<b>追加</b>
開始 停止 リスタート
60 平成19年2月14日

編集 実行 プション	~#/	1970	
9210	HTTP U/JIAF		
/ yegi-bin/test-egi	表版 //cp-bin/tett-cpi		
/ icg-bin/printers /manual/index.imi)	Web #// #//6_#.5.14.19(197.168.1.10		
Actual -	#->====================================		
HTTF #D459-15	HTTP UDIAL		
	TOF DAL MODE STAR & CET C POST		
	vt3t //gi-bn/test-cg		
	□ 自動リディレクト □ リディレクトに対応 W KeepAline を有効にする リクエストで送るパラメータ:		
	All II Encode! I	同会む	
	<b>4</b> 9 (1)		
	リクエストと一種に通信されなファイルに		
	291/86	Plle	
	173-98		
	MIME 2171		
	オブションタスク □ 金でのイメージとアプレットを接り着しグランロードする00004、ファイルのみり □ モニタク	LT (B)	



Apach

#### 手順6. テスト計画の編集

自動的に追加された「スレッドグループ」下の「HTTPリクエスト」に余分なものがある 場合は、そのリクエストを右クリックして[削除]を選択します。 リクエストをクリックすれば、URLの修正などが行えます。CGIなどの動的コンテンツ で、サーバに渡すパラメータがある場合はここで指定を行います。

・キスト11回 pro (thomeshini)a イル 編集 実行 オブション	arta preter 2 3rc1/bin/# A > 118, ync1 - Apache yMeter	-0
A 72108	HTTP 102221	
· / /rgi-bin/hein-egi	表記 /rg-bin/test-cgi	-
<ul> <li>/ (cp-on/proteins</li> <li>/manual/index.cml)s</li> </ul>	Web 7-1/ 7-1/6,85,11.00[192.168.1.10	
<ul> <li>/mdex.smlja</li> </ul>	#->======	-
HTTP #0459-1	HTTP UDIAL	
	TOF DAL MO STAR STORE OF CONST	
	vt3t /kg-binnest-cg	
	□ 自動リディレクト □ リディレクトに対応 记 KeepAlive を有効にする	
	リクエストで送るパラメータ」	
	6.8 8 Ercode! 99g	69
	EX THE REPORT OF	
	7.462	
	19.4.4.61	-
	MME 2471	=
	オブションタスク □金でのイメージとアプレットを抽り直しダウンロードするermaLファイルのみ0 □モニタとして	使用





Apach

### <u>手順7.リスナーの追加</u>

試験結果を見るために、必要なリスナーを追加します。ここでは、[グラフ表示]と[統計 レポート]のリスナーを追加しました。

### 手順8. テスト計画の保存

作成したテスト計画を保存します。

🏂 Apache JMeter			
ファイル 編集 実行 オ	プション ヘルプ		
-			
♥ 益 テスト計画	スレッドグルー	1	
P _ ZL → F Ø ル − 7	追加 )		
	ALD Delete	リスナー・	アサーション 結果
- m - 2 × 2 ×	11 W L 016X	サンプラー ・	結果をグラフ表示(詳細)
(in)		१ १२ १	グラフ表示
	ペースト Ctri-V	設定エレメント ♪	メーラービジュアライザ
	捕入としてペースト	前処理 ▶	Monitor Results
	M/	後処理 ▶	シンブルデータライタ
	如玄元程力		スプラインビジュアライザ
	a hh	レーブ 2	観計レポート
	复数		結果を表で表示
			ツリーで結果表示





Apach

### <u>手順9. テストの実行</u>

「スレッドグループ」の[スレッド数]や[ループ回数]などの値を調整してから実行します。 また、実行結果をクリアするには「実行」メニューの「全て消去」を選択します。

フォイル 義単 実行 オブション	^k7	フォイル 義年 実行 オブション	~N7
▼     ⇒     >>>>     >>>     >>>>     >>>>     >>>     >>>>     >>>>     >>>     >>>     >>> <td< th=""><th>グラフ表示 本称「7つ2巻や 女だのテータをファイルに出た ファイル名 ログエラーやみ Configure</th><th></th><th>読計しポート 名称 MPL ポート まだのテータをファイルに出力 ファイル名 単振_ コゴダエラーのみ Configure</th></td<>	グラフ表示 本称「7つ2巻や 女だのテータをファイルに出た ファイル名 ログエラーやみ Configure		読計しポート 名称 MPL ポート まだのテータをファイルに出力 ファイル名 単振_ コゴダエラーのみ Configure
	0 2 9/B 400 # 7 2 / 7 4 2 # 9 20 # 9 2 9/B 400 # 10 2 / 7 4 2 # 9 20 # 9	日 593¥8 日 開料に上で上 ● ○ 503¥ ● HTF7104557-11	URL         # Sangled         Average         Median         SON Line         Max         Error X         Through         #S/rec           (rop-bn         360         66         47         137         9         679.0.005         28.0/rec         12.5         13           (rop-bn         961         75         48         152         6.966.0.005         28.0/rec         12.5         13           (rop-bn         961         75         48         152         4.967.0.005         28.2/rec         0.00           (rop-bn         961         13.6         19         74         3.967.0.005         28.2/rec         0.00           (rop-bn         962         10.0         18         66         3.442.0.005         28.2/rec         0.00           (rot         3.854         52         3.1         111         3         967.0.005         131.7/rec/3.9.77







### 各試験の前には以下のような事前試験を実施して下さい。

### 試験前の事前処理

- 測定するシナリオを複数回実行する

### ·理由

-システムが保持しているキャッシュに値を入れたり、Javaのコードのコン パイル動作を事前に実施するため。

### •注意

-試験時にかける負荷に合わせた事前処理をすることをお勧めします。



