SMARTBEAR



LoadNinja - クイックスタート

LoadNinja 1.0 2018年 10月 31日



目次

LoadNinja クイックスタート	3
LoadNinja とは	3
LoadNinja によるテスト	3
LoadNinja が利用される理由	4
LoadNinja チュートリアル	7
必要要件	7
LoadNinja – クイックスタート	8
どのように動作するか見る	8
1. プロジェクトの作成	9
2. テストの記録	11
3. シナリオの作成	17
4. 負荷テストの実行と結果の表示	23
負荷テストを開始する	23
テストが開始されたら	24
テスト実行を監視する	24
次のステップ	28
拡張 Web テストの作成	28
結果の表示とデバッグ	28
さまざまな Web アプリケーション	28
お問い合わせ	29

LoadNinja クイックスタート

LoadNinja とは

LoadNinja は、Web アプリケーション用のクラウドベースの負荷テストおよびパフォーマンステストプラットフォームです。 開発者、QA チーム、およびパフォーマンス エンジニアは、Web サーバーが大量の負荷を受けているかどうか、サーバーが堅牢で拡張性があるかどうかを確認することができます。

LoadNinjaは、以下の質問に答えるのに役立ちます。

- 認知されうるスローダウンなしに、同時に何人のユーザーが Web アプリケーションを操作できますか?
- サーバーをクラッシュさせる負荷は何ですか?
- 負荷が増加すると、応答時間がどのように変化しますか?
- ハードウェアとソフトウェアの変更は、どのようにサーバーのパフォーマンスに影響しますか?
- その他。

LoadNinja によるテスト

LoadNinja を使用すると、ブラウザーで負荷テストを直接作成して実行できます。

1. Web テストを記録する

Web テストは、注文や製品カタログの検索などのアプリケーション使用のシナリオを表します。 LoadNinja に組み込まれたブラウザー レコーダーで Web テストを記録します。 URL を入力して、記録を開始するだけです。

記録中に、Web ページに期待されるコンテンツがあることを確認するため、validation(検証)を追加してください。

2. 負荷テスト シナリオを作成する

シナリオは、負荷テストの定義です。シナリオは、1つまたは複数の Web テストを組み合わせ、仮想ユーザー (VU) の数、Ramp-up (開始時) 負荷パラメーター、およびその他の詳細を指定します。

3. シナリオを実行する

LoadNinja は、クラウドで負荷テストを実行し、テスト期間中に自動的にクラウド マシンを準備します。 各仮想ユーザーは実際のブラウザーを使用してテスト アクションを再生実行するので、待ち時間やクライアント サイド スクリプトの実行など、実際のユーザーとまったく同じように Web アプリケーションと対話します。

実行中、LoadNinja は、テストの進行状況とアプリケーションの状態を監視するのに役立つリアルタイムのグラフを表示します。

4. 結果を分析する

テストの完了後、LoadNinja はテスト実行中に収集されたパフォーマンスメトリックを含むレポートを生成します。 このデータは、サーバーのパフォーマンスを見積もり、SLA を満たしているかどうかを確認するのに役立ちます。

LoadNinja が利用される理由

- 実際のユーザー アクションをシミュレートする実際の負荷テストを作成。 LoadNinja を利用すると、テストする Web サイトでユーザーアクションを 記録し、数百または数千の仮想ユーザーを使用して記録されたトランザク ションをシミュレートできます。 したがって、すべての仮想ユーザーは実 際のユーザーと非常によく似た動作をします。つまり、負荷テストは実生 活に近く、単に Web サイトの URL をリクエストするだけではありませ ん。 これは、他のオンライン テスト ソリューションが提供するものとは 異なります。
- 実際のユーザーシミュレーションに実際のブラウザーを使用。 LoadNinja は、実際のブラウザーで記録されたトラフィックを再生実行します。 各仮想ユーザーはそれぞれのブラウザーで動作し、互いに影響しません。 LoadNinja は、アクション間の一時停止を含む、ユーザーのアクションを正確にシミュレートします。 一部のテストするページでクライアント

側のスクリプトコードが実行される場合、そのコードはブラウザーでのテスト実行中にも実行されます。 つまり、数百人の数の実際のユーザーが作業するように、負荷テストが機能します。 これは、HTTP リクエストを記録して再生し、実際のユーザーアクションと実際のブラウザーの機能をシミュレートしない多くの他のオンラインおよびデスクトップ ソリューションとは異なります。

• プラットフォームに依存しない負荷テストを作成。

LoadNinja はプラットフォームに依存しません。 インターネットにアクセスできる任意のコンピューターのブラウザーでテストを作成して実行できます。

テストを素早く簡単に作成。

組み込みのブラウザーでユーザーアクションを記録してテストを作成します。 ブラウザー プラグインのコーディング、ダウンロード、インストールは不要です。

テストで明確なユーザーアクションを活用。

LoadNinja は、読みにくい要求と応答のセットではなく、理解可能なテストコマンドの形式でユーザーアクションを記録します。 セッション状態を維持するため、またはユーザー固有のデータを提供するために、記録されたWeb テストを構成するための時間と労力を費やす必要はありません。 つまり、データの相関関係は必要ありません。

• 必要に応じて負荷テストを調整。

現実のユーザー行動をシミュレートするには、リソースが必要です。デスクトップ ソリューションの場合は、負荷エージェントを実行するために追加のマシンが必要です。 LoadNinja はクラウド上で動作するため、追加の「エージェント」コンピューターを用意する必要がなく、記録された Webテストを保存したり、膨大な蓄積されたテスト結果のためのディスクスペース見つける必要はありません。 LoadNinja はこれらすべてを自動的に行います。シミュレートする仮想ユーザー数を選択するだけです。

• 一般的な画面解像度でテストを記録。

現在のWeb アプリケーションは、様々なデバイスや画面解像度によって若干異なります。LoadNinja を使用すると、ノートパソコン、iPad、携帯電話で、もっとも一般的な画面解像度で作業しているユーザーをシミュレートする負荷テストを簡単に作成できます。

仮想ユーザーの活動をリアルタイムで監視。

LoadNinja は、ユーザーアクションをシミュレートするブラウザーに接続す

るため、ブラウザー ウィンドウを簡単に表示したり、仮想ユーザーの動作 を監視したりすることができます。

LoadNinja チュートリアル

このチュートリアルでは、LoadNinja で負荷テストを作成して実行する方法について説明します。

これは、初心者のユーザーのために設計されています。

必要要件

このチュートリアルを完了するには、以下のものが必要です。

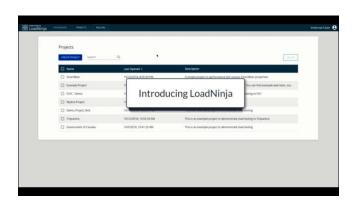
- LoadNinja アカウント(登録していない場合、<u>サインアップ</u>してください)
- 最新バージョンの次のブラウザーのいずれか:
 - Chrome
 - Firefox
 - Safari
 - Microsoft Edge

LoadNinja - クイックスタート

LoadNinja は、Web アプリケーション、サイト、およびサーバーのパフォーマンステストと負荷テストを作成および実行するためのクラウドベースのツールです。

どのように動作するか見る

次の URL をクリックして、ビデオ (2 分 25 秒) をご覧ください。 https://loadninja.com/?wvideo=6a6yaokpzi



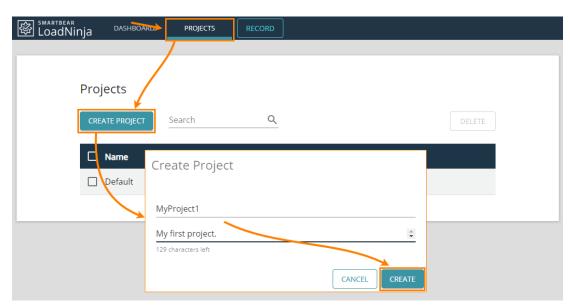
1. プロジェクトの作成

LoadNinja では、プロジェクトには負荷テストを作成して実行するためのすべての データが含まれています。

- <u>Web テスト</u> テストする Web アプリケーションでシミュレートされる一連のユーザーアクション。
- 負荷テストの定義 (シミュレートする仮想ユーザー数、実行する Web テスト、テスト期間など) を含むシナリオ。
- **プロジェクト**にはテスト結果も含まれています。

プロジェクトを作成しましょう。

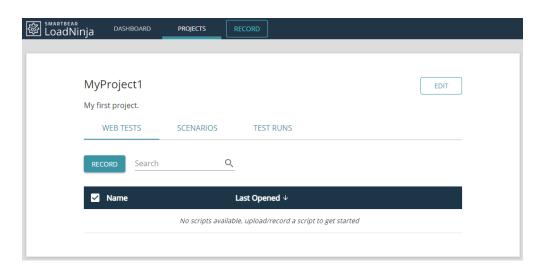
- 1. [Projects] に進み、[Create Project] をクリックします。
- 2. 表示されるダイアログでプロジェクト名を指定し、任意の説明を入力します。



LoadNinja でプロジェクトを作成

3. 準備ができたら、[Create] をクリックします。

LoadNinja がプロジェクト スペースを準備するまで、少し待ちます。



作成されたプロジェクト

これだけです。このプロジェクトを使用して、テストデータを整理します。

注意:

- プロジェクトの数は、課金プランによって制限されません。
- プロジェクト内のテストの数にも制限がありません。ただし、プランでは同時に実行する負荷テストの数が制限されています。

2. テストの記録

LoadNinja でプロジェクトを作成したら、Web テストを作成する必要があります。 Web テストは、テストする Web サイトでシミュレートする一連のアクションです。 LoadNinja では、これらを記録することで作成します。プロジェクトに複数の Web テストを記録することができます。次に、Web テストを負荷テストに組み込んで、 仮想ユーザーが Web サイト上のさまざまな訪問者のワークフローをシミュレート できるようにします。このチュートリアルでは、Web テストを1つだけ記録します。 それがどのように動作するか理解できると、さらに多くを記録することができます。

LoadNinja では、一番上のツールバーから、プロジェクト画面から、あるいはダッシュボードからの複数の場所から記録を開始することができます。 ここでは、プロジェクト画面から記録しましょう。

MyProject1
My first project.

WEB TESTS

SCENARIOS

TEST RUNS

RECORD

Search

Q

Name

Last Opened

No scripts available, upload/record a script to get started

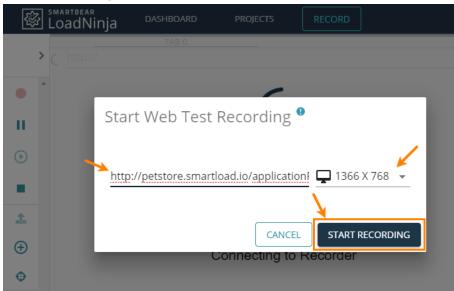
1. プロジェクト画面の [Web Tests] タブで、[Record] をクリックします。

Web テストの記録開始

2. 次のダイアログでは、テストを開始する Web ページの URL を指定し、 記録する画面の解像度を選択する必要があります (アプリケーションによっ ては、異なる画面解像度で少し見え方や動作が異なることがあります)。 この例では、サンプルのペットショップ Web アプリケーションで Web テストを記録します。

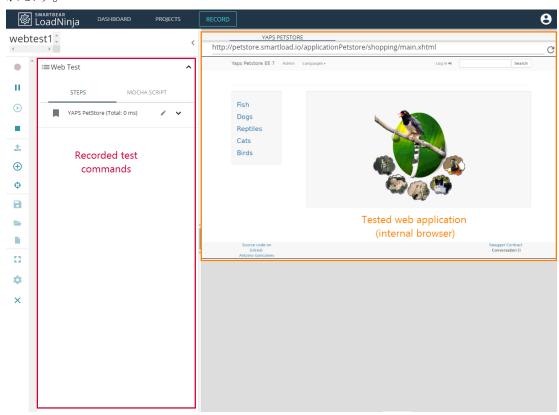
http://petstore.smartload.io/applicationPetstore/shopping/main.xhtml

URL を入力し、1366×768 解像度を選択します。準備はできましたか? [Start Recording] をクリックします。



URL を入力し、解像度を選択

3. これで、記録画面になります。テストする Web アプリケーションが右側 (内部ブラウザー領域内) に表示され、左側に記録されたイベントが表示されます。



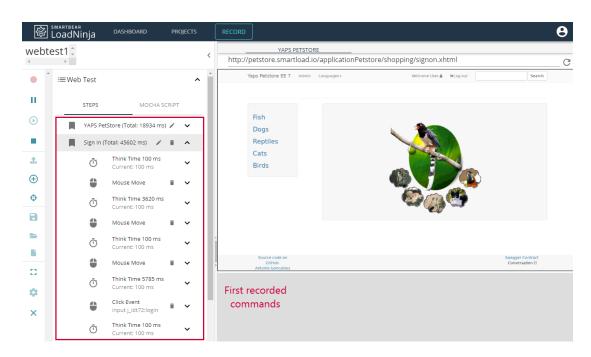
記録画面

Web テストを記録するには、内部ブラウザーでテストする Web アプリケーションを使用して作業します。リンクをクリックし、テキストを入力したり、ページをスクロールしたりします。LoadNinja は、これらのアクションを記録し、Web テストにそれらを組み込みます。

たとえば、アプリケーションのログインをテストしましょう。

- 1. **[Log in]** をクリックします。
- 2. [Returning Customer] フォームで、以下の資格情報を指定します。
 - Login: user
 - Password: user
- 3. [Sign in] をクリックします。

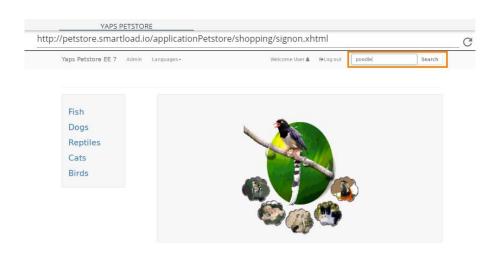
LoadNinja は、これらすべてのアクションを記録します。 これらを、左側で見ることができます。親ノードは Web ページに対応し、子ノードは ユーザーアクションおよび他のイベントに対応します。



最初に記録されたコマンド

LoadNinja では、親ノードは **Steps** と呼ばれます。 単一ページ **Web** アプリケーションでは、親ノードはアプリケーション画面に対応します。

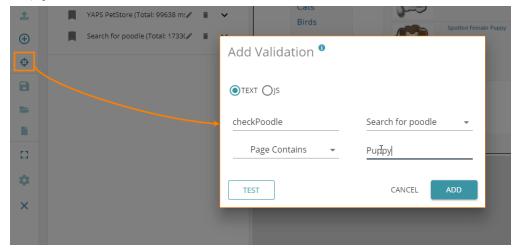
4. さらにいくつかのアクションを記録しましょう。 ペットショップでプード ルの子犬を提供しているかどうか確認してみましょう。ページ上部の検索 ボックス内をクリックし、poodle と入力して、[Search] をクリックしま す。



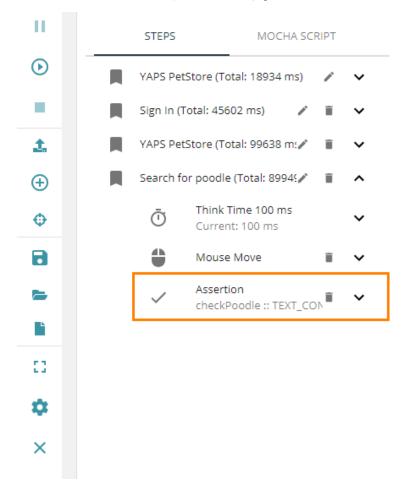
Swagger Contract Conversation []

- 5. ここで、検索結果に情報があるか確認する必要があります。これを行うには、Web テストに validation (検証) を追加します (validations は、サーバーの応答を確認し、テストする Web アプリケーションが期待どおり動作することを確認するのに役立ちます)。このチュートリアルの場合、validationは、Web ページに "Puppy" (子犬) という単語が含まれているかどうかを確認する必要があります。この validation を追加するには:

 - 2. 次の[Add Validation] ダイアログで、[Text] validation タイプを選択します。
 - 3. validation 名を入力します。
 - 4. **[Page Contains]** validation タイプを選択し、**Value** に Puppy を指定します。

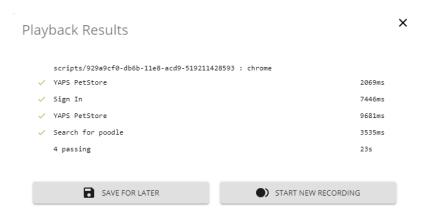


5. **[Test]** をクリックし、検証が機能することを確認します。次に、 **[Add]** をクリックします。 検証が Search for Poodle テスト ステップに追加されたことが表示されます。

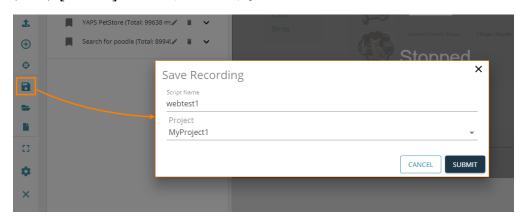


Web テストの検証コマンド

- **ヒント:** LoadNinja では、JavaScript validations を作成することもできます。それらは、ページ上で実行されているスクリプトコードが持つ Web ページ要素と同じアクセス権を持ち、それらを使用して複雑なチェックを実行することもできます。検証 JavaScript 関数は、正常終了した場合は True を返し、それ以外は false を返します。
- 6. 今すぐ記録を停止するには、[Stop] をクリックします。
- 7. **[Play]** をクリックすると、記録された Web テストが正常に動作するかどうかを確認できます。実行終了時に表示されるダイアログには、再生実行の短い抜粋が表示されます。また、Web テストを保存したり、新しいテストを記録したりするのにも使用できます。



8. [Save] をクリックして、Web テストを保存します。後続のダイアログで、Web テストの名前 (わかりやすい名前を選択)を入力し、プロジェクト ドロップダウンリストでプロジェクトが選択されていることを確認します。次に、[Submit] をクリックします。



記録保存ダイアログ

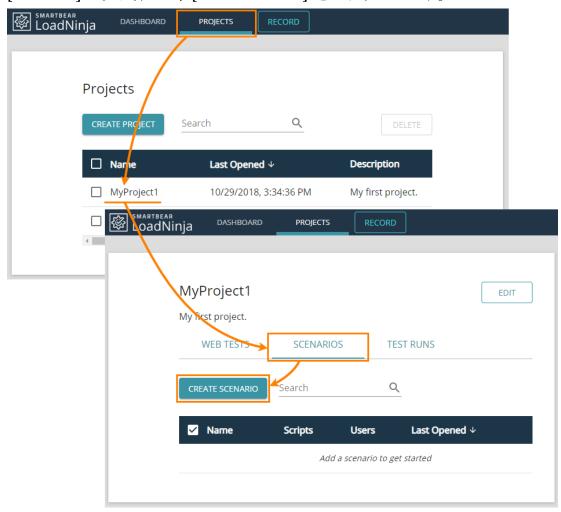
最初のWeb テストを記録して保存しました。 ここで、負荷テストのシナリオを 作成できます。

3. シナリオの作成

シナリオでは、シミュレーションする仮想ユーザーの数、これらの仮想ユーザーが実行する Web テスト、テスト期間、エラー処理手順など、負荷テストのすべての設定を定義します。

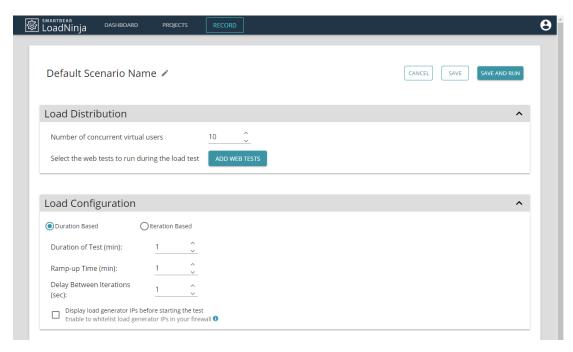
新しいシナリオを作成しましょう。 このシナリオでは、前のステップで記録した Web テストを使用します。 何らかの理由でこのステップをスキップした場合は、前のステップに戻って Web テストを記録してください。Web テストの実行準備ができたら、LoadNinja シナリオ ビルダーを使用してシナリオを作成します。

- 1. [Projects] に移動し、リスト内のプロジェクトをクリックします。
- 2. [Scenarios] に切り替えて、[Create Scenario] をクリックします。



シナリオを作成する

LoadNinja がシナリオエディターを開きます。ここでは、シナリオ設定を構成できます。



シナリオ エディター

3. エディターで、シナリオにわかりやすい名前を割り当てます。



4. [Load Distribution] セクションで、シミュレートする同時仮想ユーザー (VU) の数を指定します。このフィールドに **10** を設定しましょう。

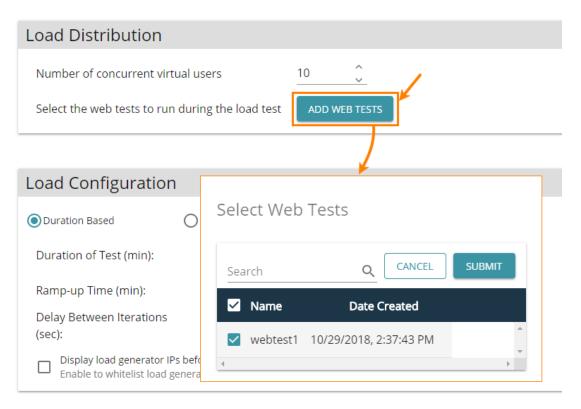
Scenario1 /



仮想ユーザー数の設定

1. この設定では、仮想ユーザーが実行する Web テストも指定する必要があります。[Add Web Tests] をクリックして、記録したい Web テストを選択します。

Scenario1

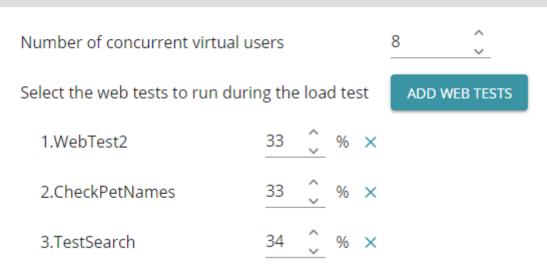


Web テストの追加

この例では、Web テストは 1 つしかなく、10 人の仮想ユーザーすべてが実行します。 プロジェクトでは、複数の Web テストを記録して使用することができま

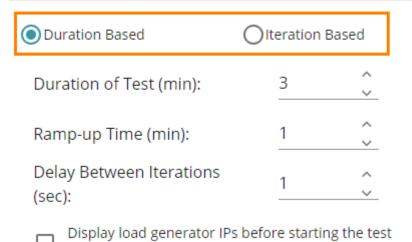
す。 [Add Web Tests] ボタンをクリックして、シナリオに追加し、各 Web テストを 実行する仮想ユーザーの割合を指定します。設定は次のようになります。

Load Distribution



2. **[Load Configuration]** セクションでは、負荷テストのタイプを duration-based または iteration-based かを選択する必要があります。

Load Configuration



Duration-based は、指定された期間、要求された数の仮想ユーザーを LoadNinja がシミュレートすることを意味します。一般的に、各仮想ユーザーは複数回実行します。仮想ユーザーの Web テストが終了すると、LoadNinja は Web テストを再度実行します (LoadNinja では、負荷テスト全体で個別の Web テストを定義するために iteration (繰り返し) を使用します)。 指定した負荷テストの期間が終了する

Enable to whitelist load generator IPs in your firewall 10

と、LoadNinja は新しい繰り返しを開始しません。同時に、仮想ユーザーの実行を 強制的に停止させることもなく、最後まで作業することができます。

Iteration-based のタイプは、LoadNinja が指定された回数だけ各仮想ユーザーを実行することを意味します。

ここでは、Duration Based タイプを選択し、[Duration of Test] フィールドで 3 分を設定します (上記の画像を参照)。

3. チュートリアルで使用する別の設定は、Ramp-up Time です。 値が 0 より大きい場合、LoadNinja はテスト開始時からすべての仮想ユーザーをシミュレートしません。 仮想ユーザーは 0 から始まり、その増分期間中に要求された VU に徐々に仮想ユーザーを追加していきます。

このフィールドに **1分**を設定しましょう。この例では、テストを開始してから 1分後に負荷が同時に 10人の仮想ユーザーに達することを意味しています。

Load Configuration

Ouration Based	Olteration Based	
Duration of Test (min):	3	
Ramp-up Time (min):	1 🗘	
Delay Between Iterations (sec):	1 🗘	

Display load generator IPs before starting the test
Enable to whitelist load generator IPs in your firewall 1

注意: Duration of Test の値には、Ramp-up Time が含まれていないことに注意してください。 つまり合計テスト時間は、Ramp-up Time と Duration of Test の合計になります。

- 4. シナリオ エディターでは、他の設定も指定できます。
 - 仮想ユーザー繰り返し間の遅延

- 仮想ユーザーをシミュレートするクラウドマシンの IP アドレスを表示する オプション (ローカル ネットワークで動作する Web アプリケーションをテストする場合に必要です)。
- *Think Time* 設定 (LoadNinja がユーザーのアクション間で一時停止をシミュレートする方法)。
- エラー処理ポリシーなど。

このチュートリアルでは、これらの設定は説明しません。デフォルトの値のままにします。

それらの詳細については、Scenario Settings を参照してください。

これで、シナリオを実行できます。これを行うには、シナリオ エディターで [Save and Run] をクリックします。 次のステップでこれをします。

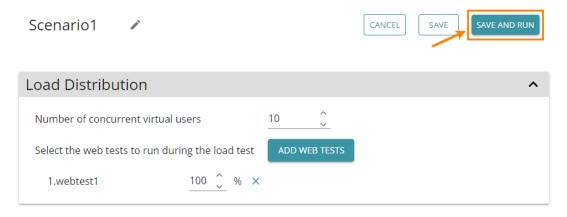
テストを実行せずに、シナリオ設定の変更を保存する場合は、シナリオ エディターで [Save] をクリックします。

4. 負荷テストの実行と結果の表示

前のステップで作成したシナリオを実行しましょう。

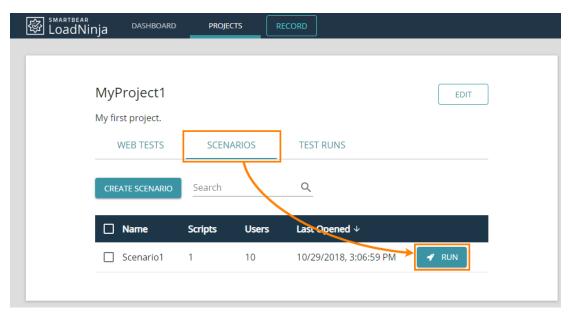
負荷テストを開始する

負荷テストを開始するには、シナリオ エディターで [Save and Run] をクリックします。



シナリオの保存と実行

別の方法は、プロジェクトのシナリオ リストからシナリオを実行することです。



シナリオ リストから負荷テストを実行する

テストが開始されたら

最初に、テストの開始時にタイマーが表示されます。コノタイマーが画面に表示されている間、LoadNinja は要求された負荷をシミュレートするために必要なクラウドマシンを割り当て、仮想ユーザーを準備します。

これらの準備ステップの後、負荷テストの進行状況をリアルタイムに監視することができます。

テスト実行を監視する

LoadNinja では、テスト結果をリアルタイムで監視し、仮想ユーザー シミュレーション中に発生する問題をデバッグすることができます。

[Charts] タブでは、グラフ形式で蓄積された結果が表示されます。

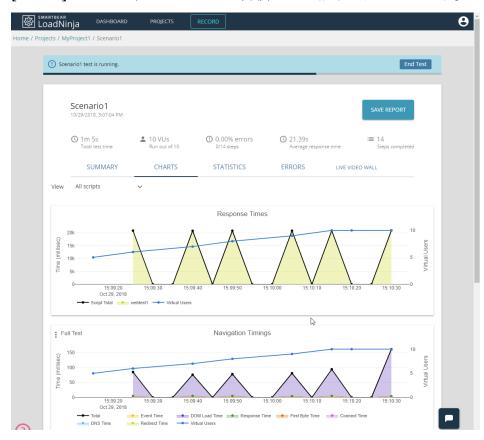
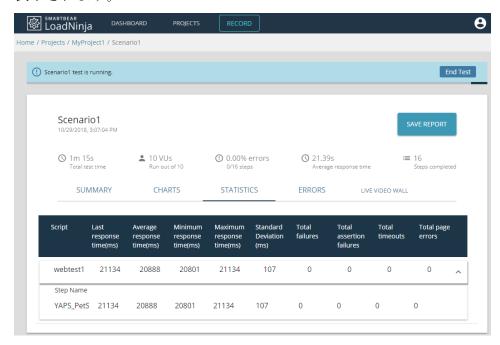


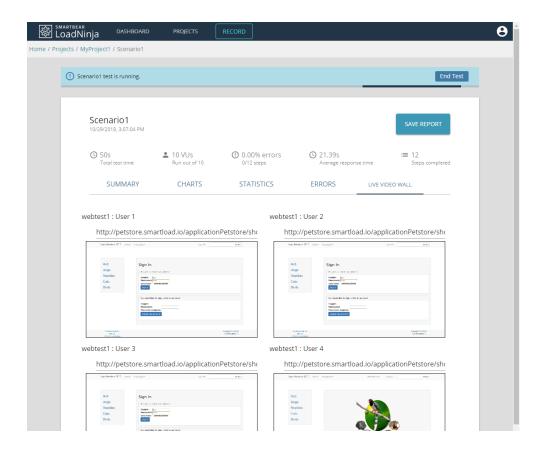
Chart タブ - テスト結果

[Statistics] タブでは、個々の Web テストの統計結果が表示されます (この例では、Web テストは 1 つだけです)。 シナリオで使用する各 Web テストの結果が表示されます。



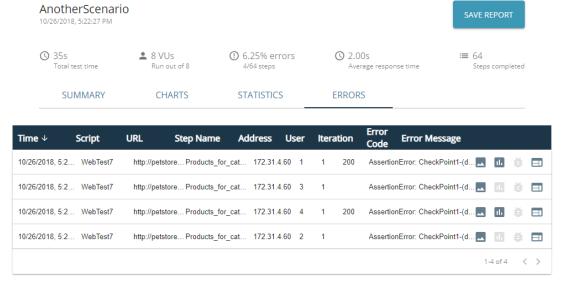
Statistics タブ - テスト結果

[Live Video Wall] タブでは、仮想ユーザーが作業しているリモート クラウド マシンのデスクトップを表示できます。各フレームには、仮想ユーザー ID、実行されている Web テストの名前、ユーザーがいる URL が表示されます。 右側では、アクティブな仮想ユーザーの割合、成功した Web テストの割合、平均スループット(Web サイトが一度に処理するリクエストの数)を確認できます。



Live Video Wall

[Errors] タブでは、実行中に発生したエラーに関する情報が表示されます。この チュートリアルのテストは、おそらくエラーなく実行されますが、テストでエ ラーが発生することもあります。 [Errors] タブの表示サンプルを次に示します。



[Errors] タブ - テスト結果の例

右側のボタンを使用して、エラーの原因を詳細に調べることができます。

コマンド	説明
**	エラーが発生したクラウドマシン上の Web ページのイメージを 表示します。
ıl.	ロードされたリソースに関する情報を取得します。
	問題のあるページの HTML コードを取得します。
ĕ	仮想ユーザーをデバッグします リモート クラウド マシンへの接続が開きます。リモート ブラウザー コンソールでコマンドを実行すると、そのページで何が起こったのかをよく理解できます。詳細については、 Debugging を参照してください。

テストが終了すると、**Video Wall** は非表示になります。他のタブでも結果を表示できます。

次のステップ

チュートリアルはこれで終了です。これが LoadNinja の習得の役に立つことを願っています。LoadNinja でプロジェクトを作成し、Web テストを記録し、シナリオを作成した後、負荷テストを実行し、その結果を分析しました。これで、ユーザーご自身で負荷テストを作成できるようになりました。以下は、テストを拡張し、高度な LoadNinja の機能を活用するのに役立ついくつかのリンクです。

拡張 Web テストの作成

- ⇒ Using Databanks in LoadNinja Tests
- **⇒** Validations

結果の表示とデバッグ

⇒ Debugging

さまざまな Web アプリケーション

- **→** Test Single Page Applications
- ⇒ Test Web Apps in Local Network

お問い合わせ

SmartBear 製品のお問い合わせは、下記のお問い合わせ窓口より承っております。

http://www.xlsoft.com/jp/services/xlsoft_form.html

エクセルソフト株式会社

〒108-0073 東京都港区三田 3-9-9 森伝ビル 6F

http://www.xlsoft.com

E-Mail: xlsoftkk@xlsoft.com