SMARTBEAR

SMARTBEAR ReadyAPI

SoapUI Pro: API 機能テスト ツール
LoadUI Pro: API 負荷テスト ツール
ServiceV Pro: サービス仮想化ツール
VirtServer: 仮想サービスサーバー

ReadyAPI スタートガイド

For ReadyAPI 3.0

XLSOFT

エクセルソフト株式会社

作成：2019. 12. 13

目次

はじめに

このガイドブックは、ReadyAPI (v3.0.0) ユーザー ガイドの中で、スタート ガイドに関して説明した部分を抜粋して翻訳したものです。 ReadyAPI はいくつかの手段を組み合わせて、Web サービスの包括的なテストの作成を支援します。

- 機能テストでは、サービスが期待どおりに機能することを確認します。
- セキュリティテストは、サービスがハッキング攻撃に対してどのように抵抗するかを検証します。
- 負荷テストは、サービスが大量のユーザーを処理できるかどうかを確認します。
- 仮想サービスは、作成される前にサービスの実際の動作をシミュレートします。

SoapUI : SoapUI　スタート ガイド

Secure: Secure スタート ガイド

LoadUI : LoadUI スタート ガイド

ServiceV: ServiceV スタート ガイド

VirtServer: VirtServer チュートリアル

インストール ファイルの入手、各機能の詳細を含んだ最新バージョンの ReadyAPI ユーザー ガイドは、SmartBear 社の下記のサイトで参照できます。

https://support.smartbear.com/readyapi/docs/get-started.html

Translated by XLsoft Corporation

第 1 章: SoapUI スタートガイド

ReadyAPI を使用すると、Web サービスの機能テスト、負荷テスト、およびセキュリティテストを簡単に実行できます。 サービスのローカル仮想コピーを作成して、実際のテストがオンラインになる前にテストを実行することもできます。

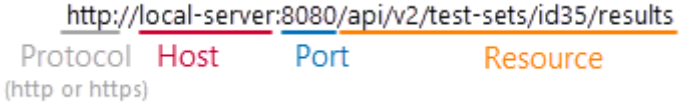
このチュートリアルでは、SoapUI で基本的な機能テストを作成する方法について説明します。 ファイルから Web サービス定義をロードし、1 つの操作のテストを作成し、このテストを実行し、アサーションを使用してテスト結果を検証します。

基本概念

ReadyAPI を使用してテストを作成および実行するには、Web サービステクノロジーとテストの原則を一般的に理解する必要があります。 これらのトピックは膨大で、ReadyAPI ドキュメントの範囲外ですが、ReadyAPI をより早く開始できるように、概要をまとめました。

用語

- Web サービスは、クライアントとサーバーが HTTP プロトコルまたは HTTP に基づいた他のプロトコルを介して Web 上でデータを交換するクライアント/サーバーアプリケーションです。 このようなアプリケーションの例には、ナビゲーションソフトウェア、オンラインバンク クライアント、気象監視システムなどが含まれます。
- クライアントがリクエストを送信する URL には、テストされるサーバー(host)、通信に使用されるポート番号、およびリクエストされたサーバー**リソース** (ページやファイルパスなど) に関する情報が含まれます。



- クライアントがサーバーに送信する**リクエスト**の構造は次のとおりです。

- HTTP メソッド (GET、POST、DELETE など)、ターゲット URL、プロトコル バージョンを指定する開始行。
- レスポンス データの予期される形式、またはリクエスト データのサイズと形式などの追加情報を渡すヘッダー。
- (オプション) リクエスト ボディ。 一部のリクエストタイプはそれを使用しません。

レスポンスには同様の構造があります。

- レスポンスコードとメッセージのある開始行。 頻繁に使用されるコードには、200 OK (成功) と 404 Not Found (失敗、リクエストされたリソースが見つかりません) があります。
- レスポンスデータ形式を説明し、Cookie、サーバー情報などの追加の値を含むヘッダー。
- レスポンスボディ。たとえば、リクエストされたデータ、画像、ファイルなどの配列。

- リクエストおよびレスポンスのボディでよく使用される形式は、JSON および XML です。
- クライアントが実行のためにサーバーに送信するコマンドは、サービスのアーキテクチャスタイル（SOAP または REST、以下を参照）に応じて、アクション、メソッド、または操作と呼ばれます。
- Web サービスの 2 つの一般的なアーキテクチャスタイルは、SOAP と REST です。
    - **SOAP サービス**は、HTTP 上に構築された SOAP プロトコルを使用します。これらのサービスは、POST タイプの HTTP リクエストを使用し、リクエストおよびレスポンスのボディで XML 形式のデータを渡します。すべてのリクエストは同じ URL に送られ、実行される操作は、リクエストボディの特別なリクエストヘッダーまたは XML 要素によって指定されます。

      SOAP サービスは、サービスでサポートされる操作、そのパラメーターの種類、およびデータ形式を厳密に記述する WSDL 定義を使用します。
    - **REST サービス**は HTTP 経由で機能します。実行される操作は、HTTP メソッドとリクエストされたリソース名の組み合わせによって設定されます。たとえば、オンライン ペットストアの RESTful サービスには、/ pets リソースを含めることができます。
      `POST http://petstore.io/pets` リクエストはペットに関する情報をデータベースに追加でき、`GET http://petstore.io/pets` リクエストは利用可能なペットに関する情報を取得できます。

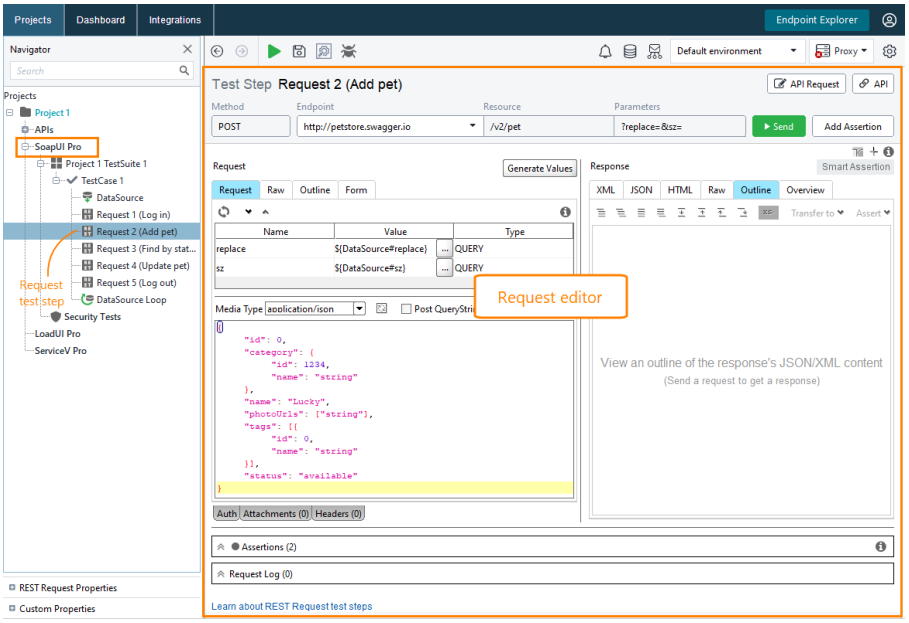
REST サービス定義には、OpenAPI (Swagger)、WADL、およびその他のいくつかの形式があります。ただし、一部の開発者は、RESTful サービスの定義をまったく提供していません。

Web サービスをテストする方法

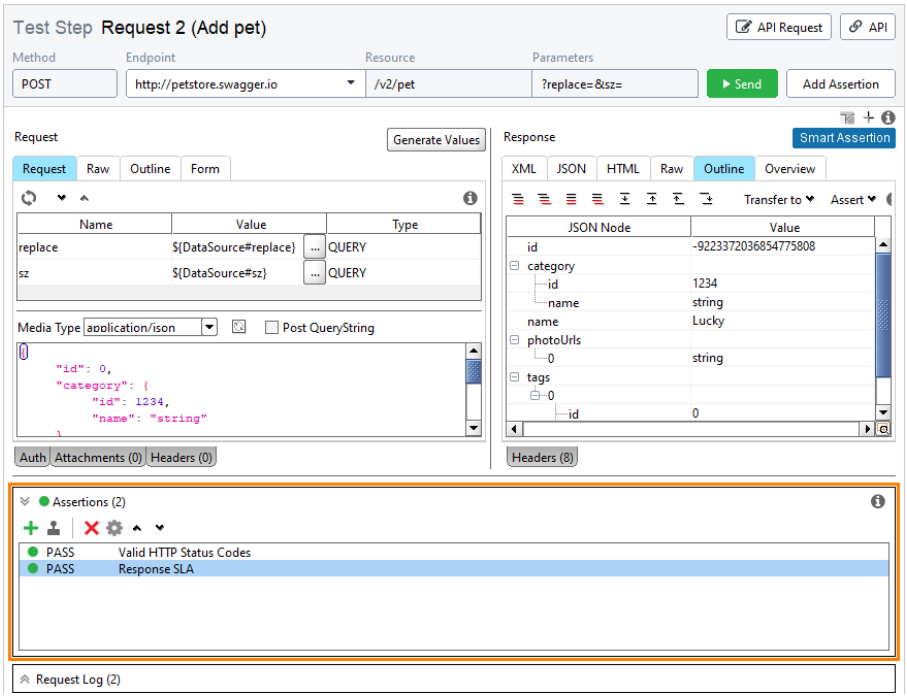
- Web サービスが正常に機能することを確認するには、**機能テスト**を作成して実行します。

  これらのテストは、サーバーに**リクエストを送信し**、そのレスポンスを**検証**します。

  ReadyAPI では、**SoapUI** で機能テストを作成します。 SoapUI という名前にもかかわらず、SOAP および REST サービスの両方のテストをそこで作成できます。特別なエディターでリクエストを簡単にシミュレートし、パラメーターをカスタマイズできます。



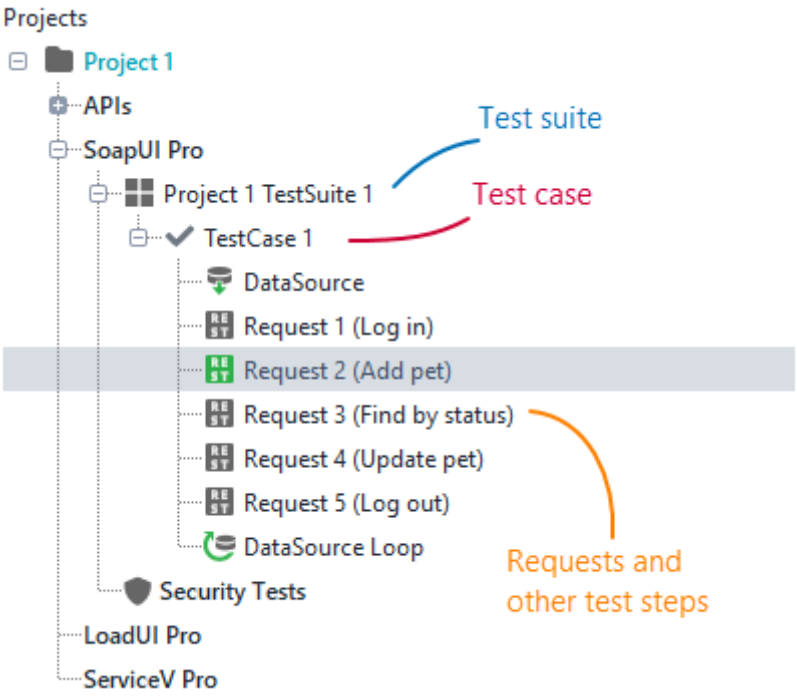
レスポンスデータとレスポンスコードを検証するには、**アサーション**をテストリクエストに追加します。



サーバーが正常に動作しているかどうかを判断する最も簡単な方法は、レスポンスコードを確認することです。 200 OK は通常、サーバーがリクエストを正常に処理したことを意味します。

- 実際には、クライアントは通常、一連のリクエストをサーバーに送信します。 たとえば、オンラインショップの場合、最初のリクエストはサインインに使用され、後続のリクエストは一部の製品の購入に使用されます。

  SoapUI では、リクエストやその他のテストステップを**テストケース**に整理することにより、この実際の動作をシミュレートします。



連携して動作する複数のテストケースは**テストスイート**にグループ化され、テストスイートはテストプロジェクトに属します（上の画像を参照）。

テストプロジェクトを作成し、自動化された機能テストを追加しましょう。

1. 機能テストの作成

必要要件

ReadyAPI で **SOAP** サービスをテストするには、このサービスの WSDL 定義が必要です。 この定義では、サービスの操作、およびリクエストとレスポンスの形式について説明します。 ReadyAPI はこの情報を使用してリクエストをシミュレートします。

REST サービスにも定義を含めることができます。 最も頻繁に使用される定義形式は、OpenAPI (以前の Swagger)、WADL、およびその他のいくつかです。 これらの定義を ReadyAPI にロードし、これらの定義の情報に基づいてテストケースを作成できます。

一般的な場合、REST サービスには定義がまったくない場合があります。 サービス URL へのリクエストを記録することにより、このようなサービスのテストを ReadyAPI で作成できます (これは API Discovery と呼ばれます)。 ReadyAPI は、追跡されたトラフィックに基づいてリクエストおよびレスポンス パラメーターに関する情報を取得します。 ただし、この「観測」データは定義からの情報ほど正確ではないため、可能な場合は定義を使用することをお勧めします。

使用する Web サービス定義

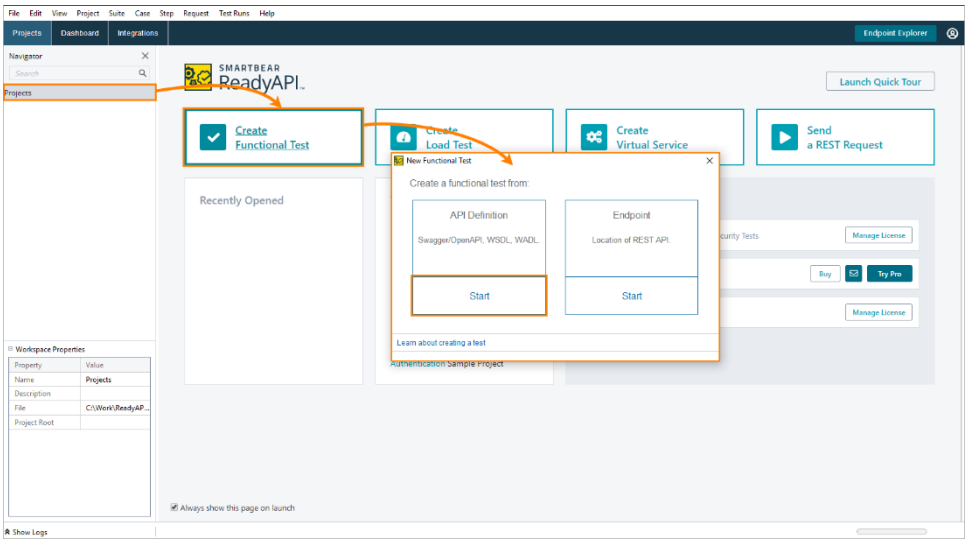
Petstore サンプル Web サービスのテストを作成します。 これは REST サービスです。 ここでその定義を見つけることができます–

```
http://petstore.swagger.io/v2/swagger.json
```

この定義には OpenAPI 2.0 (Swagger) 形式があります。 今すぐ定義をダウンロードする必要はありません。 ReadyAPI は、後で機能テストを作成するときにこれを行います。 下記参照。

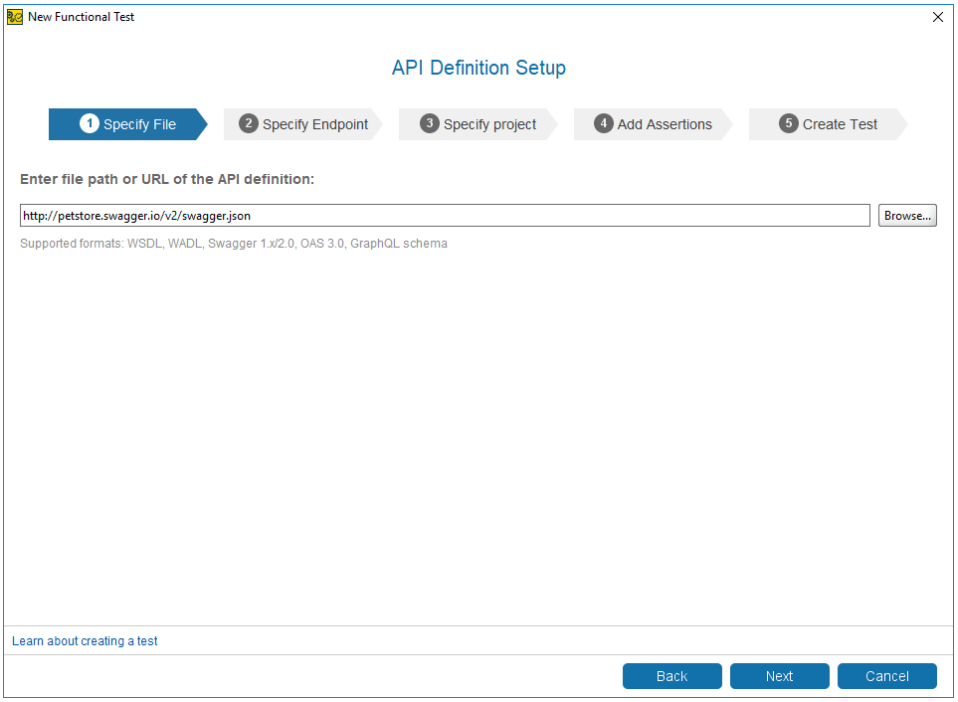
テストを作成

1. スタートページを開き、[**Create Functional Test**] をクリックし、後続のダイアログで [**API Definition** (API 定義)] を選択します。



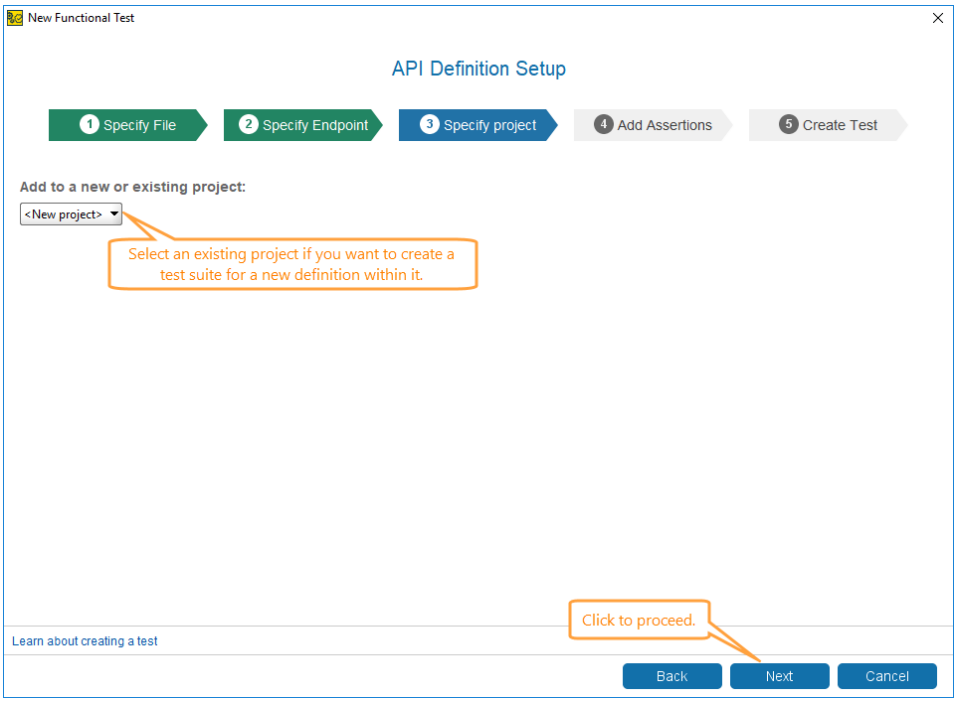
2. 後続のウィザードで、Web サービスの定義の URL を指定します。 このチュートリアルでは、次の URL を使用します。

```
http://petstore.swagger.io/v2/swagger.json
```



Next をクリックして、続行します。

3. 追加した定義の新しいプロジェクトを作成するか、既存のプロジェクトに追加するかを選択します。

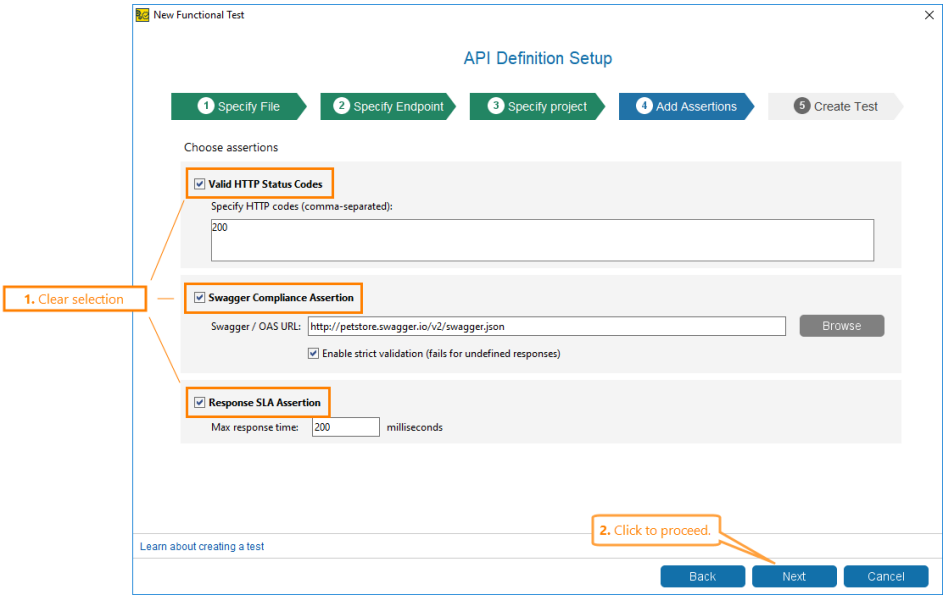


注意: ワークスペースでプロジェクトが開いていない場合、ウィザードはこの手順をスキップします。 この場合、ReadyAPI は新しいプロジェクトを作成し、新しいテストを追加します。

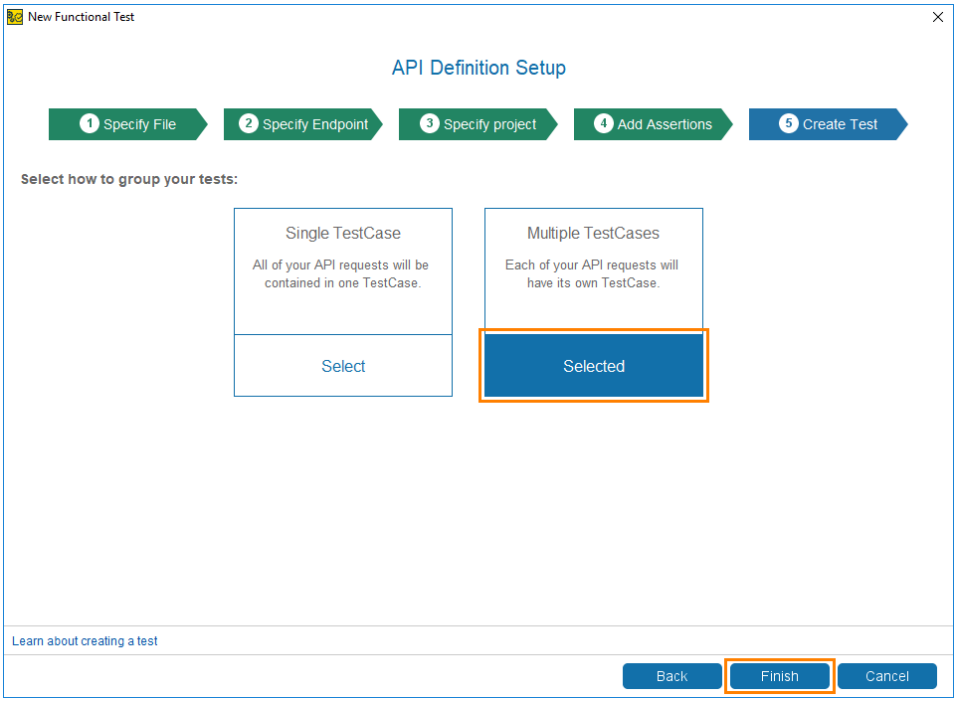
Next をクリックして、続行します。

4. ウィザードのこのページで、テストに追加するアサーションを選択できます。 ReadyAPI は、選択したアサーションを新しいテスト リクエストに追加します。

   アサーションは、API が期待どおりに機能することを確認します。 このチュートリアルの後半で、これらの詳細について説明しますが、ここで選択をクリアし、**Next** をクリックします。



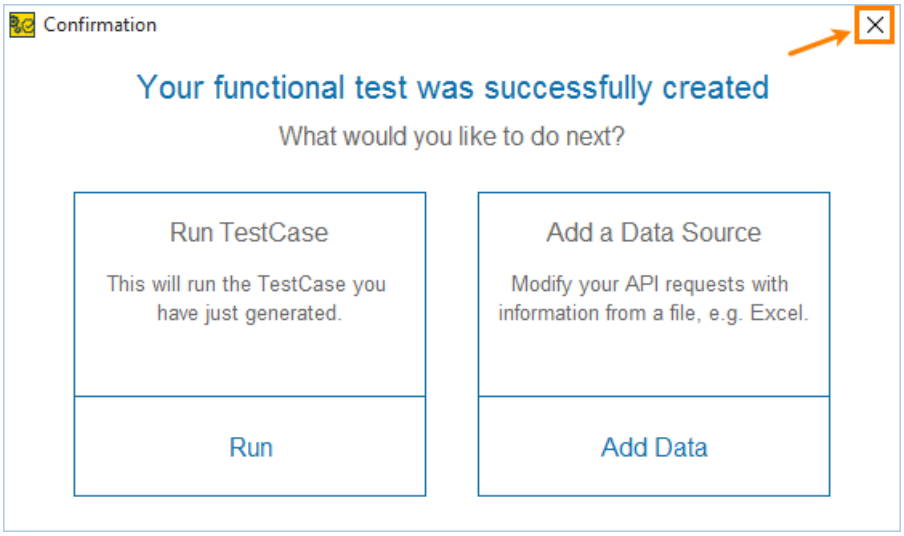
5. Web サービスに定義されているすべての操作に対して 1 つのテストケースを使用するか、複数のテストケース (各操作に 1 つ) を使用するかを選択します。 後者のオプションを使用しましょう。



Finish をクリックして、テストを作成します。

6. ReadyAPI は、テストプロジェクトを作成し、テストケースを追加します。

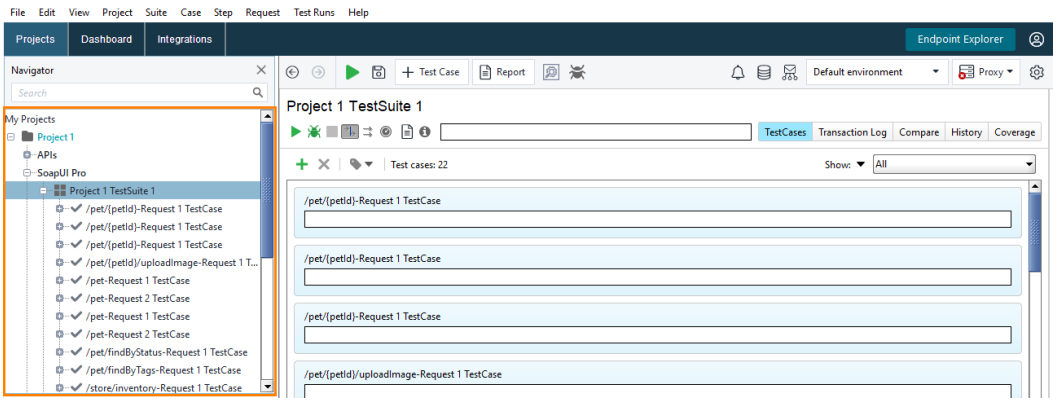
   その後、もう 1 つのダイアログボックスが表示され、作成したテストを実行したり、データソースを追加したりできます。 このチュートリアルでは、これらのオプションは使用しません。 このダイアログを閉じます。



左側のナビゲーターパネルでプロジェクトを確認できます。 このチュートリアルの次のステップでは、作成されたプロジェクト、サービス、およびその操作を調べます。

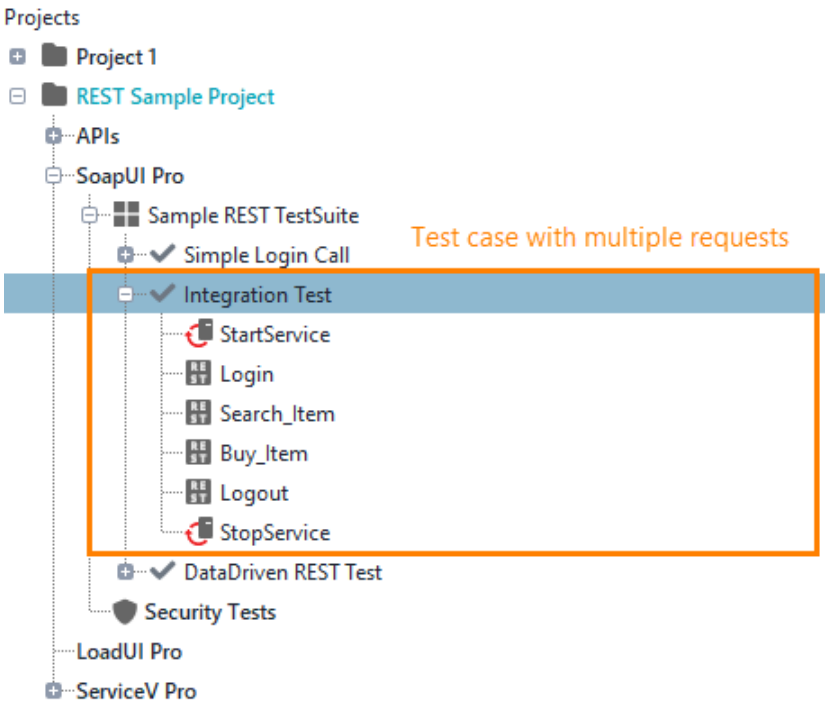
2. テストプロジェクトの探索

作成されたテストプロジェクトは、左側の[**Navigator**]パネルで確認できます。



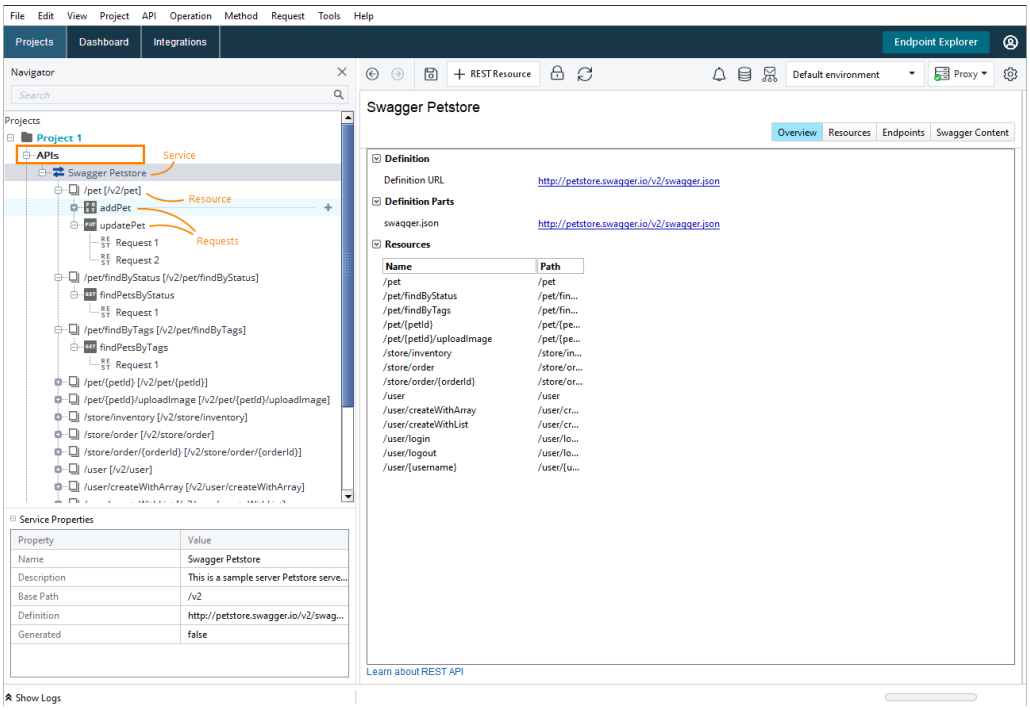
プロジェクトには複数のテストケースがあります（操作ごとに 1 つ）。 これらはすべてテストスイートにグループ化され、テストスイートはプロジェクトに属します。

この場合、各テストケースには 1 つのリクエストテストステップのみがあります。 実際には、テストケースには通常複数のステップがあります。



このチュートリアルの次のステップで、テストケースにリクエストを追加する方法を説明します。

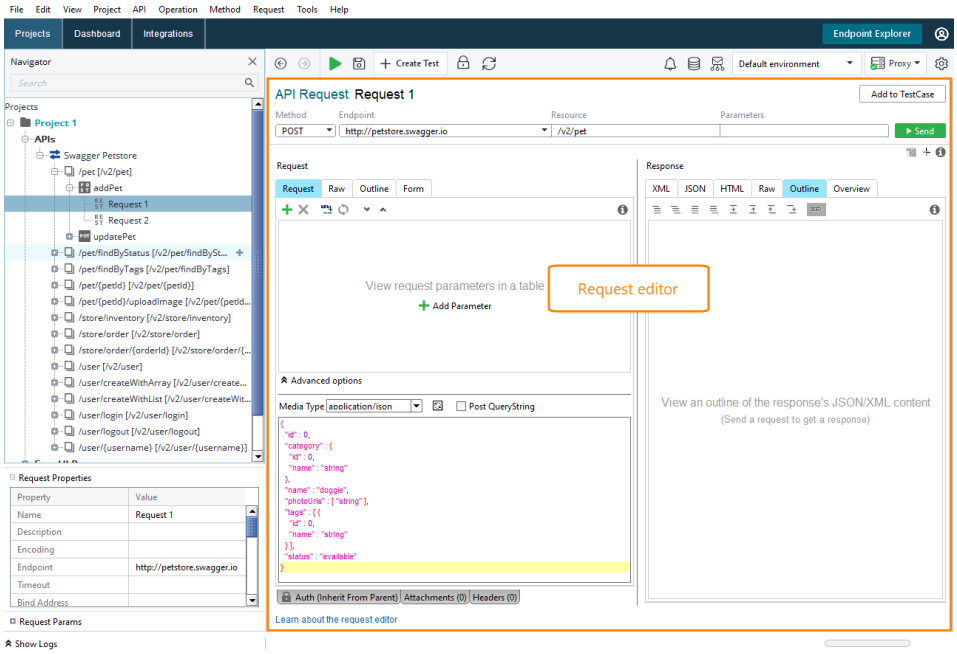
サービスを調べるには、[Navigator]パネルで **API** ノードを展開します。 サービスのリソースとリクエストのツリーのような構造が表示されます。



最上位のノードは Web サービスに対応しています。 その子ノードはリソースに対応しています。 リソースノードには、順番に、Web サービス仕様のリソースに対して定義されたリクエストに一致する子ノードがあります。

右側のエディターを使用して、選択したサービス、リソース、またはリクエストのパラメーターを表示します。

一部のリソースには、複数のリクエストが定義されています。 これらのリクエストには通常、異なる HTTP メソッドがあります。 他のリソースには、リクエストが 1 つしかありません。 **プロジェクト**のツリーに表示されるリクエストは、テンプレートリクエストとして機能します。 たとえば、ここで各リクエストに異なるパラメーターを設定し、これらのリクエストを SoapUI のリクエストテストステップのベースとして使用できます。



プロジェクトでは、リクエストに対して必要な数のテンプレートを作成できます。

リクエストエディターからリクエストを実行して、このリクエストが正しく機能するかどうかを確認することもできます。 ただし、これは実行される個々のリクエストであることに注意してください。 実際のシナリオをシミュレートするには、複数のリクエストでテストケースを実行する必要があります。

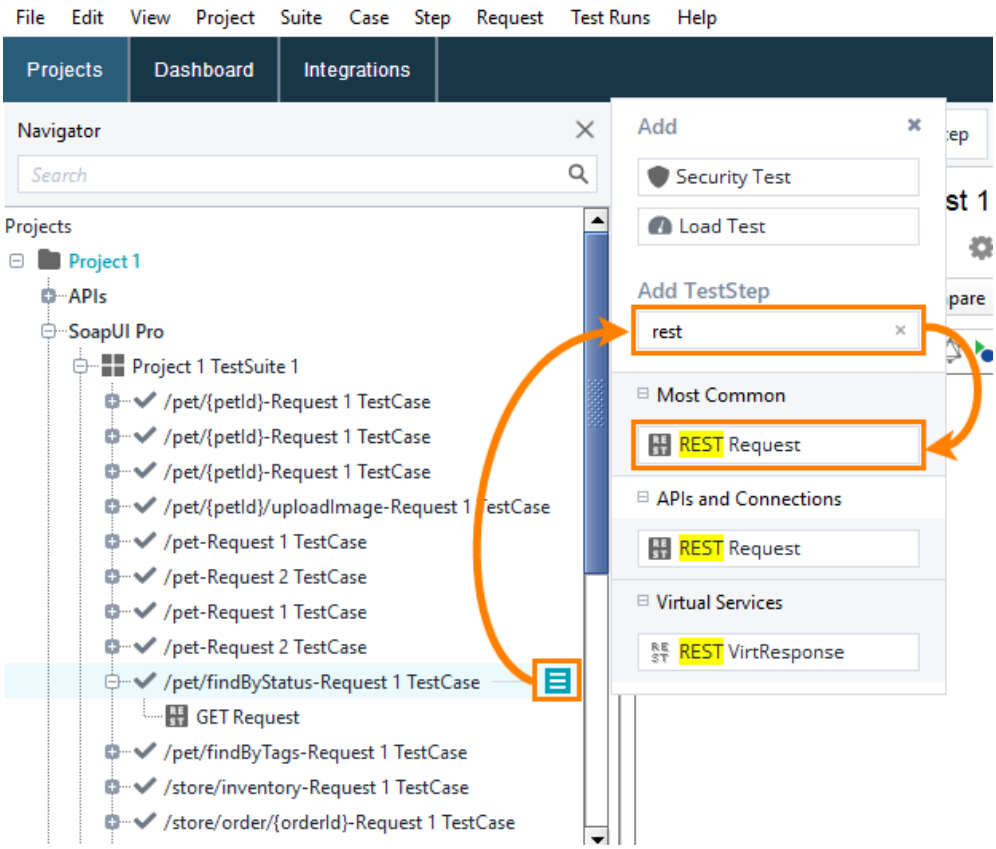
チュートリアルの次のステップでは、リクエストをテストケースに追加し、リクエストパラメータを変更します。

3. SoapUI テストの変更

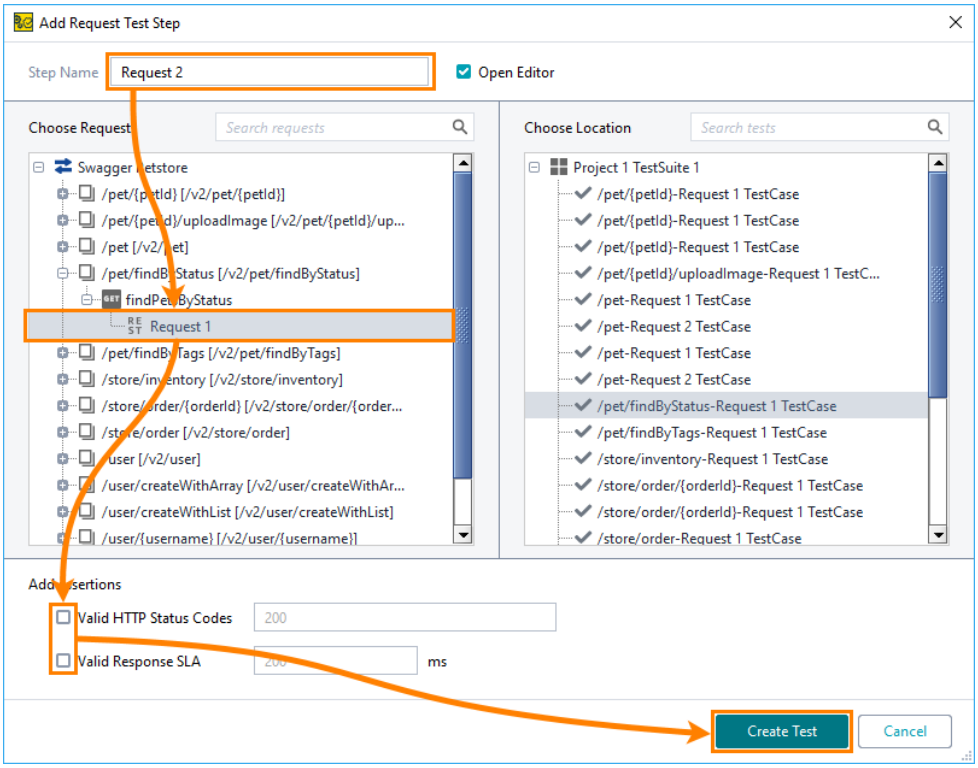
テストステップをテストケースに追加し、リクエストパラメータを変更する方法を見てみましょう。 "/pet/findByStatus-Request 1 Test Case" を変更します。 REST リクエストのテストステップを追加します。

リクエストテストステップの追加

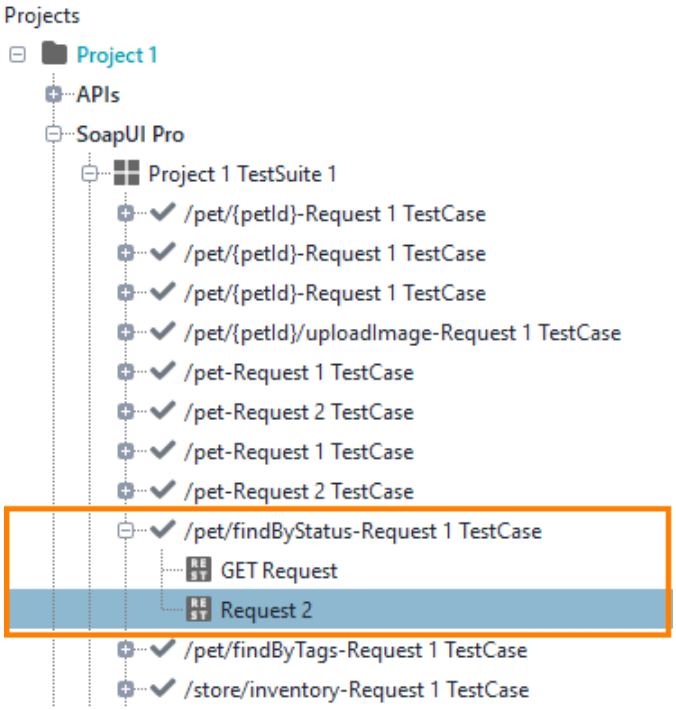
1. 左側の **Navigator** ツリーで、**/pet/findByStatus-Request 1 Test Case** ノードにカーソルを合わせ、右のボタン( ) をクリックします。
2. フライアウトメニューで、**REST リクエスト**テストステップを選択します。 すばやく見つけるには、[**検索**]フィールドに名前を入力して始めることができます。



3. 後続のダイアログで、テストステップ名を入力し、作成するテストステップのテンプレートリクエストを選択し、アサーション チェックボックスをオフにして、**Create Test** をクリックします。



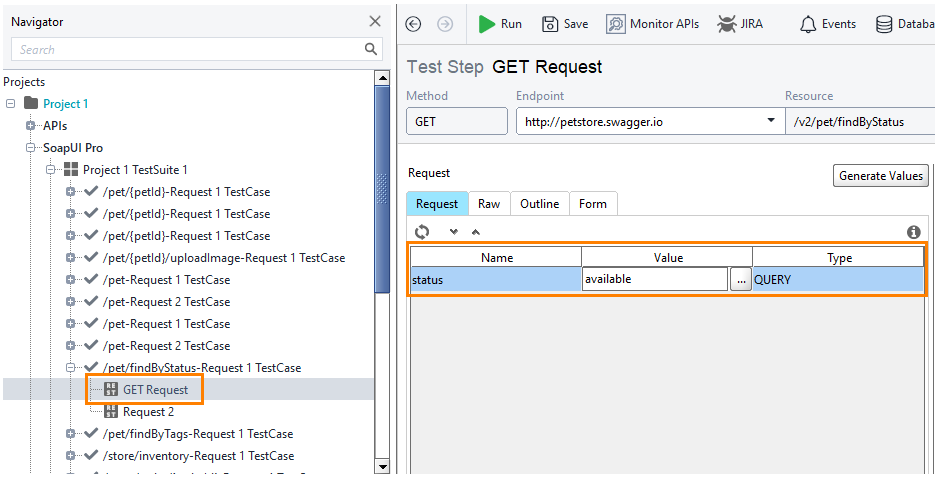
リクエストがテストケースに追加されます。



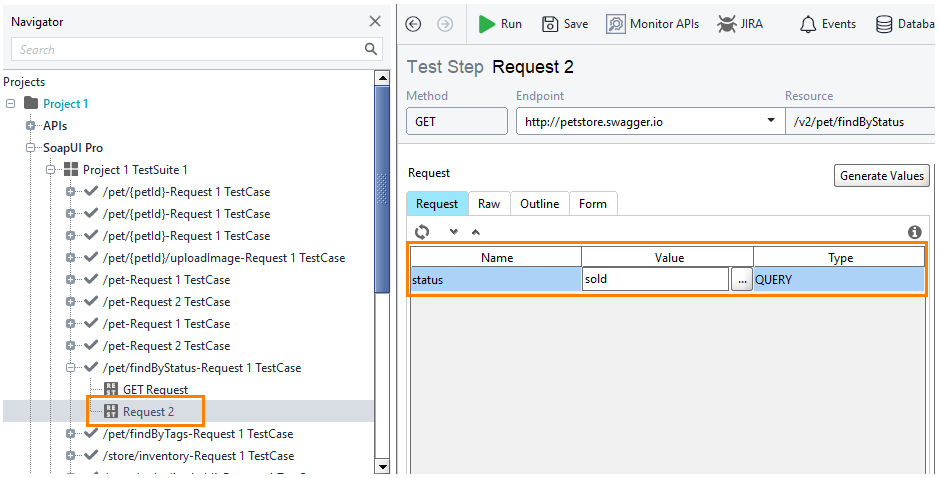
リクエスト パラメーターの変更

現在、同じ操作をシミュレートする 2 つのリクエストがあります。 リクエストが異なるデータのセットを返すようにパラメーターを変更しましょう。

1. Navigator ツリーで、**GET Request** テストステップを選択します。
2. リクエストエディターで、*status* 行の[**Value**]セル内のどこかをクリックし、**available** を入力します。 ENTER を押して変更を確認します。



3. 同様に、テストケースの 2 番目のリクエストを選択し、*status* パラメーターの [**Value**]セルに **sold** と入力します。 Enter キーを押して変更を確認します。



これで、テストケースを実行する準備が整いました。 これを次のステップで行います。

4. SoapUI テストの実行

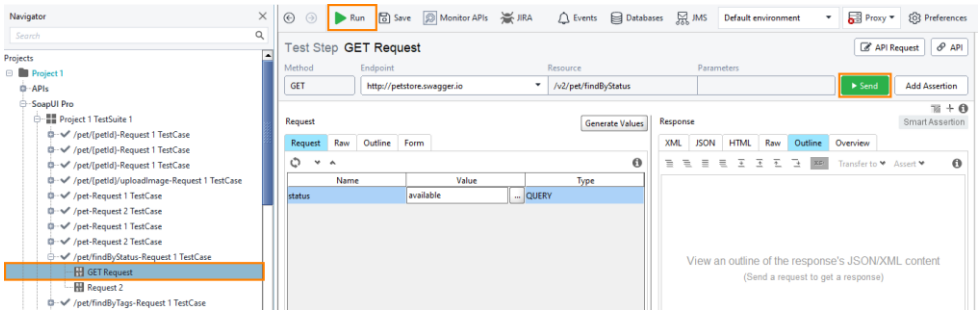
ReadyAPI では、リクエスト、テストケース、テストスイート、またはテストプロジェクト全体を実行できます。

個々のリクエストを実行することは、これらのリクエストがどのように機能するかを確認する必要がある場合に意味があります。 ユーザーシナリオをシミュレートするには、複数のリクエストをシミュレートするテストケースを実行する必要があります。 テストスイートは、複数のテストケースの実行に役立ちます。テストプロジェクトの実行は、このプロジェクトに属するすべてのテストスイートの実行を意味します。

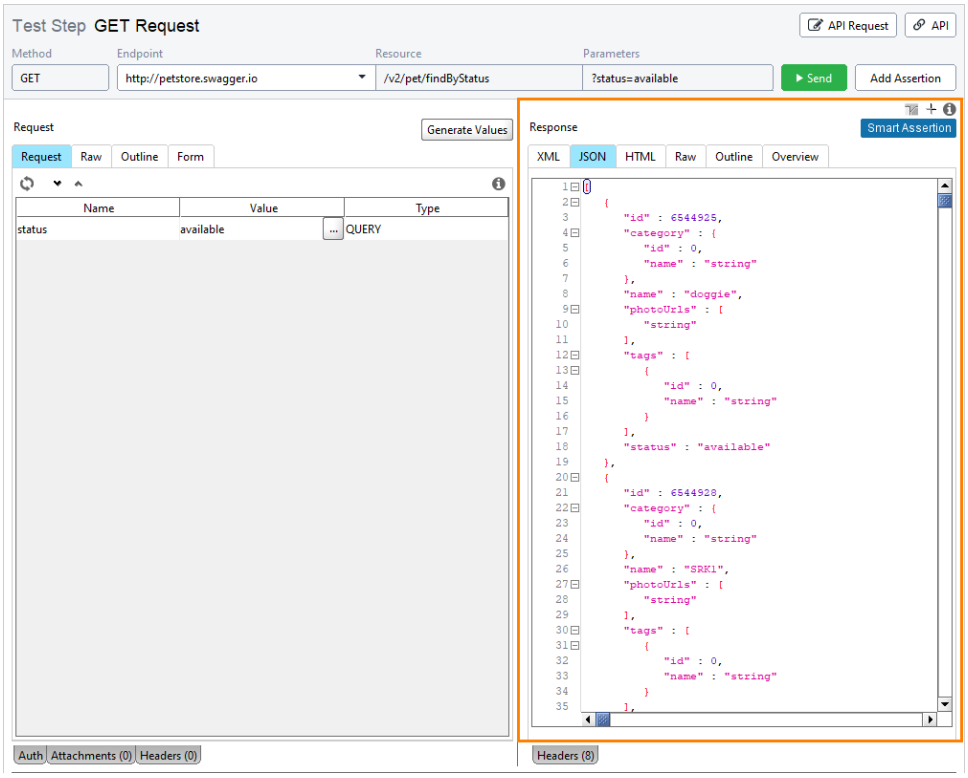
個別のリクエストを実行

通常、テストを作成するときに個々のリクエストを実行します。 これにより、レスポンスデータをすばやく表示したり、必要に応じてリクエスト パラメーターを変更したり、アサーションを追加したりできます。 複雑なテストでは、リクエストは多くの場合、前のテストステップのデータに依存します。 これらのリクエストは、個別に実行すると失敗します。

個別のリクエストを実行するには、左側の Navigator パネルで選択し、メインツールバーの ▶ をクリックするか、リクエストエディター ツールバーの ▶ Send をクリックします。



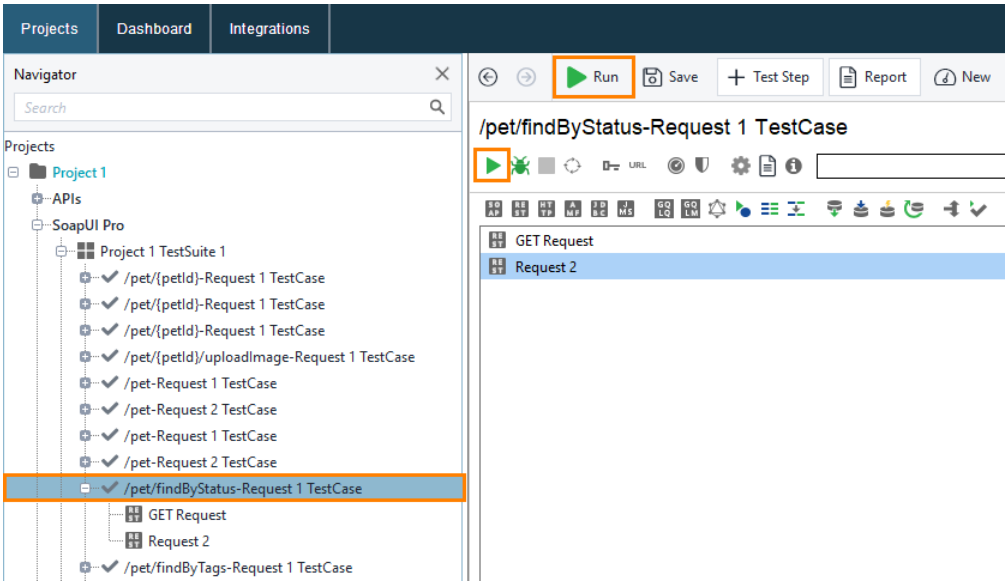
リクエストエディターの右側にレスポンスの内容が表示されます。



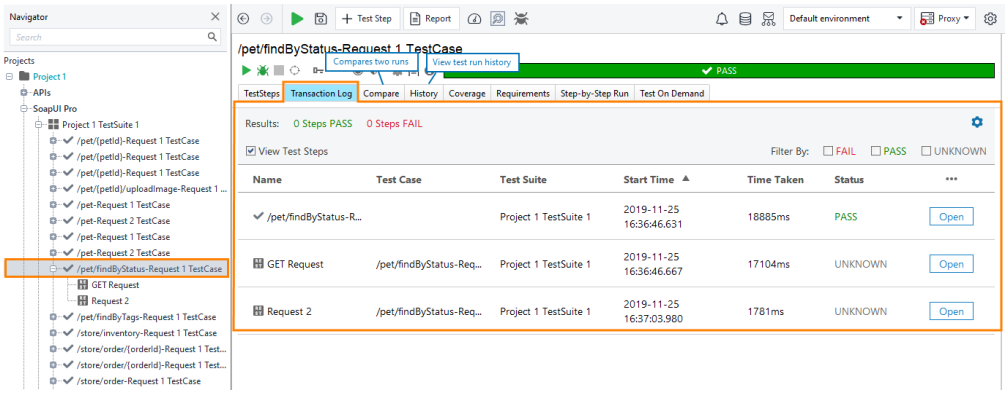
テストケースで別のリクエストを実行し、そのレスポンスを確認します。.

テストケースの実行

テストケースを実行するには、Navigator パネルでテストケースを選択し、メインまたはエディターのツールバーの ▶ をクリックします。

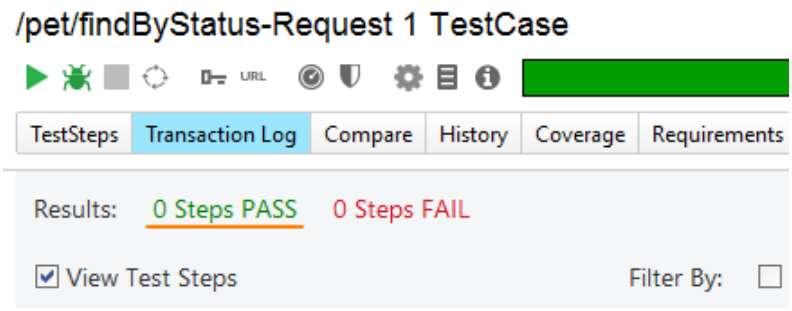


SoapUI は、テストケースのテストステップを 1 つずつ実行します。 テストケースエディターに結果が表示されます。 [**Transaction Log**] ページには、テストの実行に関する時間情報が表示されます。



ご覧のとおり、テストケースエディターには、テスト実行の傾向を表示できる **History** (履歴)や、2 つのテストログを比較できる **Compare** (比較)など、テストステップレベルでは使用できないページがあります。

リクエストが正常に実行されたことに気付くかもしれませんが、Transaction Log にはテストステップがゼロと報告されています。

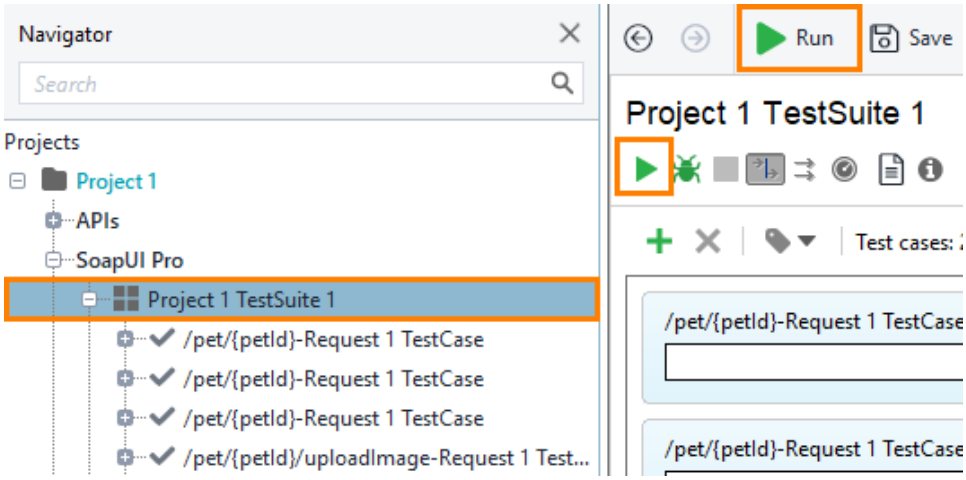


これは、テストステップに結果を検証するチェックポイント(アサーション)がないために発生します。 このチュートリアルの次のステップでアサーションを作成します。

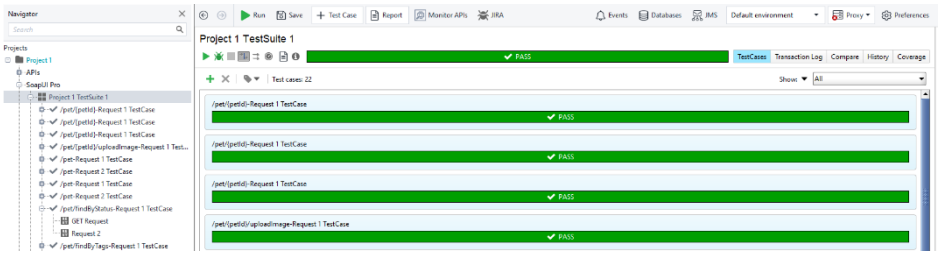
次のステップに進む前に、テストスイートとプロジェクトの実行方法について説明します。

テストスイートとプロジェクトを実行する

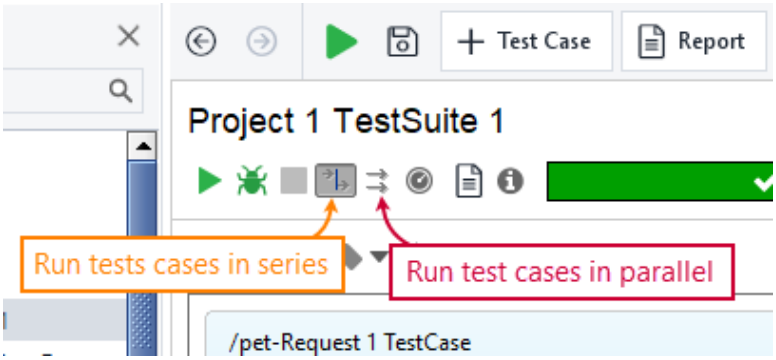
テストスイートまたはプロジェクトを実行するには、Navigator でこのスイートまたはプロジェクトを選択し、メインまたはエディターのツールバーの ▶ をクリックします。



エディターにテスト結果が表示されます。



デフォルトでは、テストスイートを実行すると、テストランナーはこのスイートのすべてのテストケースを連続して実行します。 それらを並行して実行するには、最初にツールバーの ⇉ をクリックしてから ▶ をクリックします。



プロジェクトは同様の機能を提供します。デフォルトでは、一連のテストスイートを実行します。 プロジェクトエディターのツールバーには、テストスイートを並行して実行する ⇉ コマンドがあります。

リクエスト、テストケース、またはテストスイートエディターでは、リクエストが正常に実行されたかどうかを簡単に確認できます。 レスポンスデータを検証したり、実行時間をリクエストするには、アサーションを使用します。 チュートリアルの次のステップでは、テストステップにアサーションを追加します。

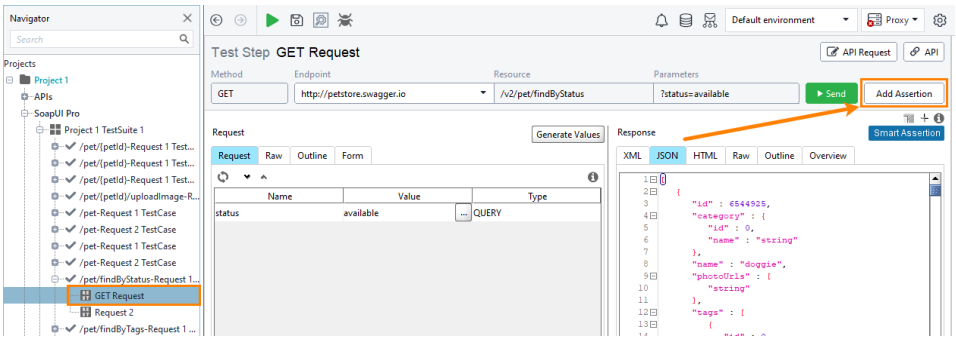
5. SoapUI テストにアサーションを追加

アサーションは、ターゲット Web サービスのパフォーマンスを確認するテスト結果に適用される検証ルールです。 SoapUI テストでは、アサーションを使用して、レスポンスコード、レスポンスヘッダー、レスポンスボディの個々の値、リクエストの実行時間を確認し、その他のチェックを行います。 テストランナーは、テストステップ(リクエスト)が終了した後にアサーションを実行します。

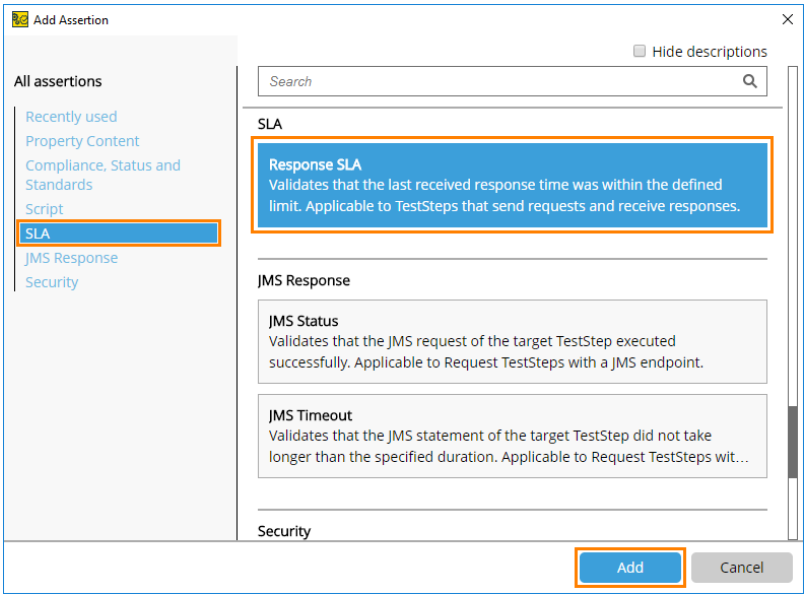
例 1 - SLA アサーション

テストされる Web サービスが事前定義された制限時間内に応答するかどうかを確認するアサーションを作成しましょう。

1. SoapUI の[Navigator]パネルで[**GET Request**]を選択し、[**Add Assertion**]をクリックします。



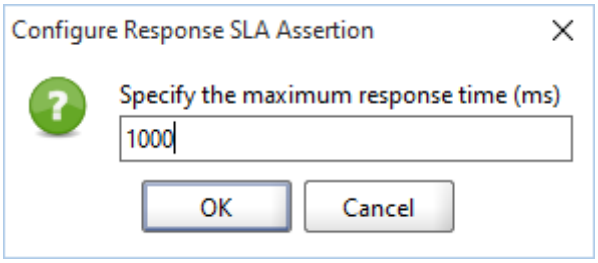
2. ダイアログで、左側の **SLA** のカテゴリを選択し、次に右側の **Response SLA** を選択し、[**Add**]をクリックします。



注意: *SLA* は、service-level agreement (サービスレベル契約)の略です。 通常、この用語は、操作の実行にかかる最大時間を意味します。 もちろん、この時間はユーザーに受け入れられるはずです。

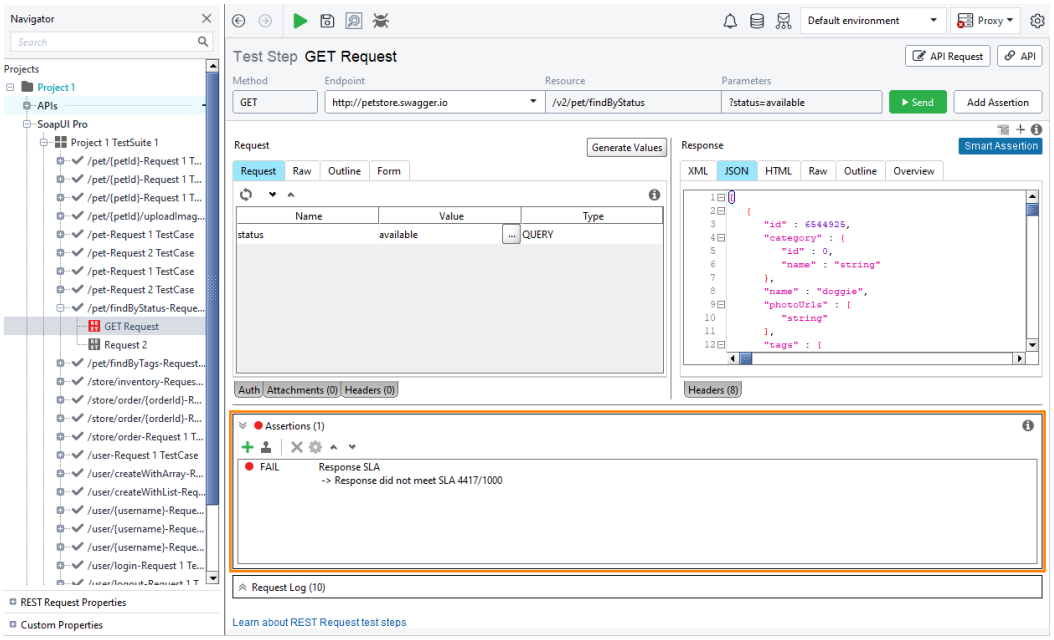
3. [Add Assertion]ダイアログで[Add]をクリックすると、アサーションパラメーターを設定できるダイアログボックスが ReadyAPI に表示されます。 このダイアログボックスは、アサーションごとに外観が異なります。 以下の画像は、Response SLA アサーションの検索方法を示しています。

   リクエストの最大許容応答時間として `1000` ミリ秒を使用しましょう。`1000` を入力して[**OK**]をクリックします。

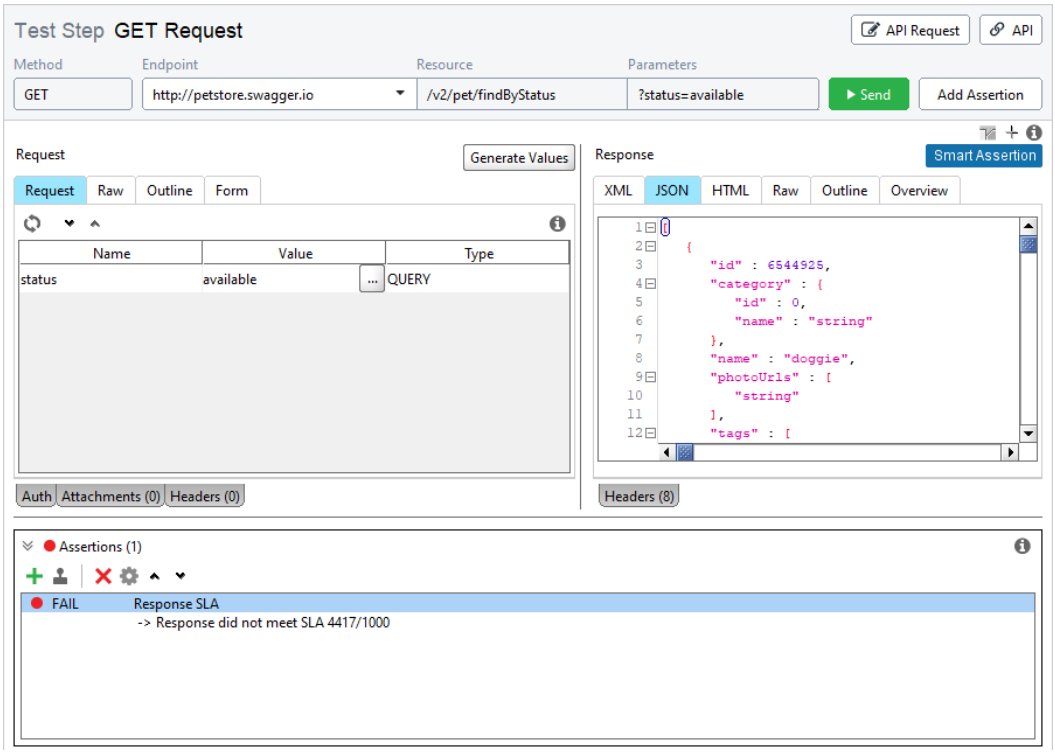


   現在、リクエストの完了に指定されたミリ秒数よりも長い時間がかかると、アサーションがトリガーされ、テストは失敗します。 実行時間が指定した値以下の場合、チェックはパスします。

新しいアサーションは、**Assertions** パネルで利用できます。

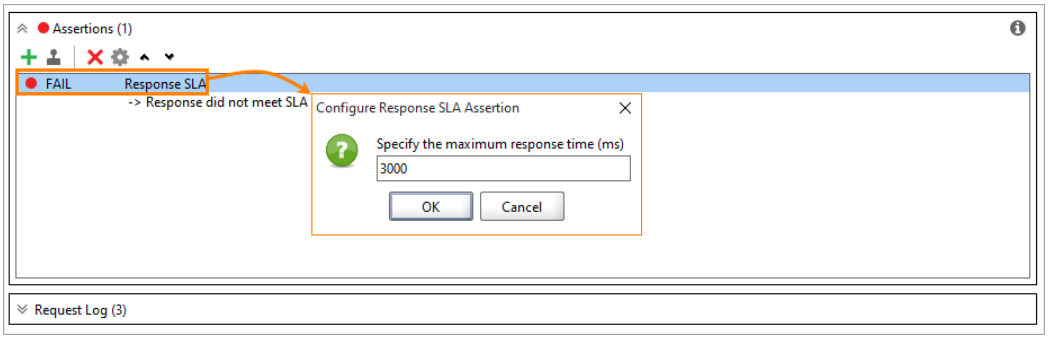


リクエストエディターにレスポンスデータがある場合、アサーションはすぐに適用されます。 今回の場合、これが持っているものです。以前にリクエストを実行し、今までにいくつかのレスポンスを持っています。



ReadyAPI は、より新しいレスポンスデータを取得するたびにアサーションを再適用します。

ご覧のとおり、このケースでは、リクエストが 1 秒よりも長く実行されたため、アサーションは失敗しました。 時間制限を変更するには、[Assertions]ページでアサーションをダブルクリックし、後続のダイアログでより大きな値を入力します。



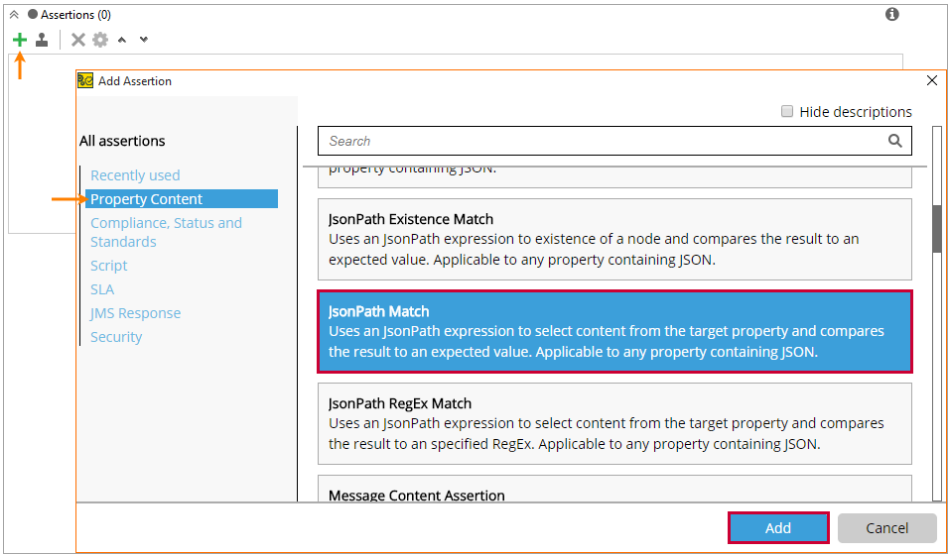
ヒント: New Functional Test ウィザードを使用して、Response SLA およびその他のアサーションをリクエストに追加できます。

例 2 - レスポンス内容を確認

次に、レスポンスデータを確認する方法を見てみましょう。 サンプルのレスポンスボディには JSON データ形式があるため、JSON データのアサーションを作成します。

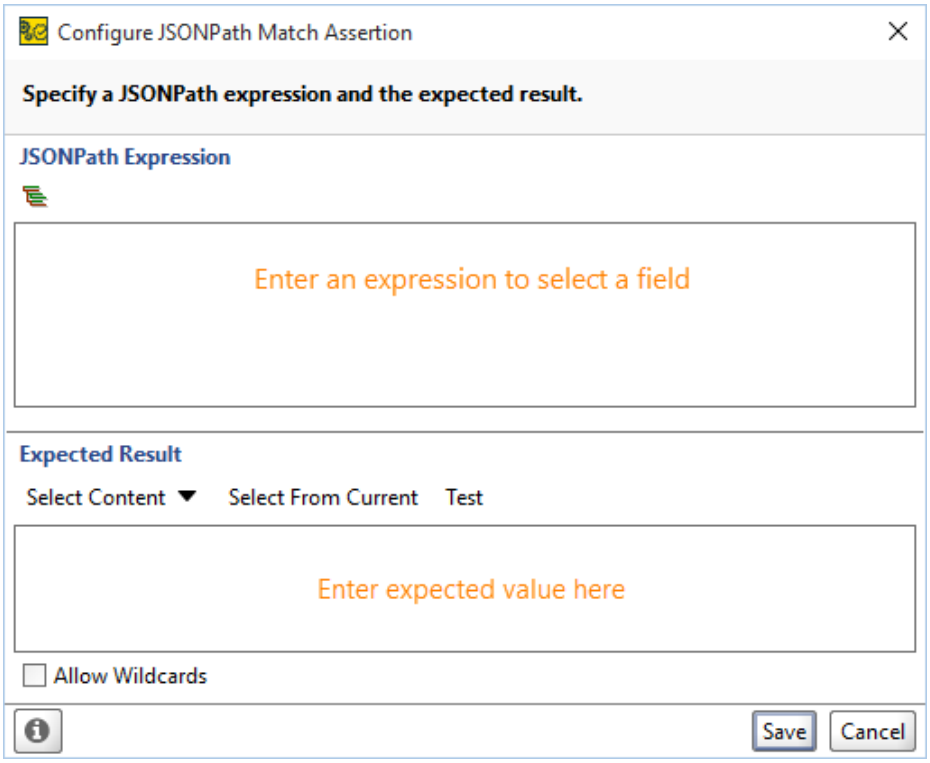
1. リクエストにレスポンスがあることを確認します。 ない場合は、 Send をクリックしてリクエストを送信し、レスポンスを取得します。「個別のリクエストを実行する」をご覧ください。
2. [Assertions]パネルで、 + をクリックして別のアサーションを作成します。

   後続のダイアログで、左側で[**Property Content**]カテゴリを選択し、右側で **JsonPath Match** アサーションを選択して、[**Add**]をクリックします。

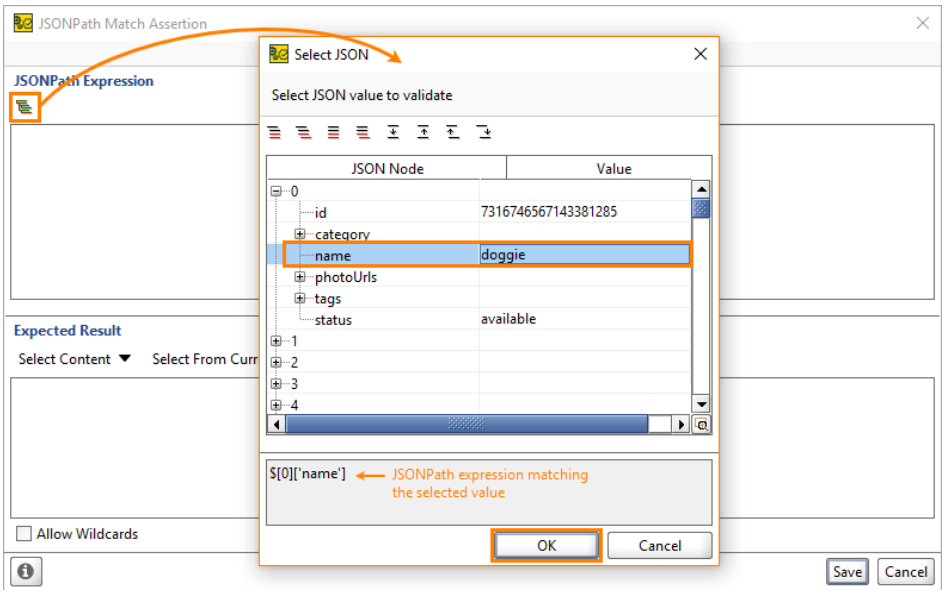


これにより、アサーションプロパティ ダイアログが呼び出されます。

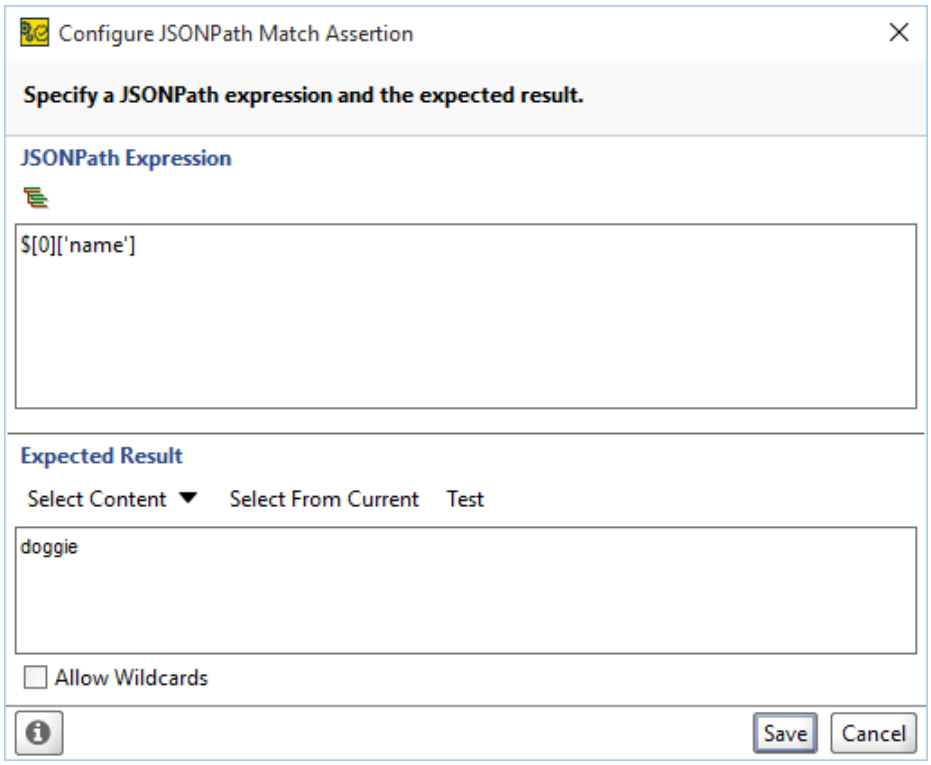
3. ダイアログで、レスポンスボディからいくつかのフィールドと、このフィールドの期待値を抽出する JSONPath 式を入力する必要があります。



目的の式を入力するか、ツールバーの **Select node** をクリックして、後続のダイアログで視覚的に値を選択できます。 をクリックして、次のダイアログで最初の配列項目の名前フィールドを選択し、[**OK**]をクリックします。

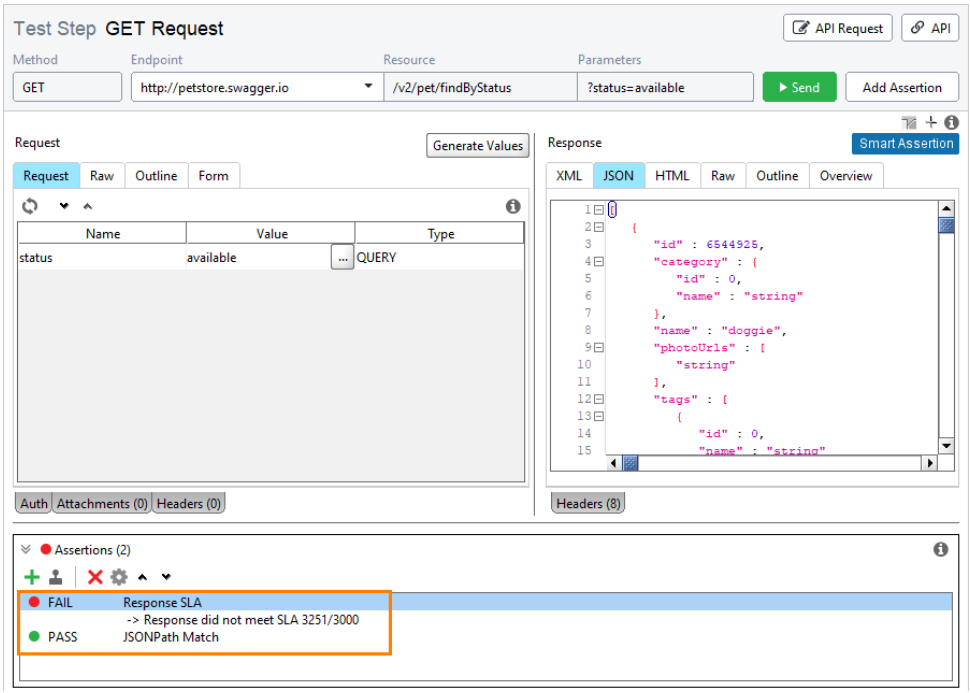


JSONPath Expression フィールドにはセレクターが含まれ、**Expected Result** には現在のレスポンスデータから抽出された値が含まれます。



4. [**Save**]をクリックして、変更を保存します。

   レスポンスデータがあるため、アサーションはすぐに適用され、Assertions ページに結果が表示されます。



これで、レスポンスが変更されると、アサーションがトリガーされ、テストステップが失敗します。

もう一度説明しますが、*Select node* コマンドは、エディターにレスポンスデータがある場合にのみ機能することに注意してください。 つまり、最初にリクエストを実行し、レスポンスを受信する必要があります。 この例では、チュートリアルの前のステップでリクエストを実行したため機能します。

次のステップ

チュートリアルは終了しました。 ReadyAPI SoapUI で Web サービスの機能テストを作成する方法を理解するのに役立つことを願っています。

もちろん、チュートリアルでは基本的な手順のみを説明しました。 これらのリンクに従って、ReadyAPI を使用した Web サービステストの詳細をご覧ください。

参考文献

- Data-Driven Testing
  SoapUI でデータ駆動型テストを作成する方法について説明します。
- Using Properties
  １つのリクエストから別のリクエストに値を転送する方法を説明します。
- Property Expansions
  プロパティ値を動的に挿入する方法を説明します。
- Authorization
  SoapUI テストで承認設定を構成する方法について説明します。
- Verifying Results
  レスポンスの検証に使用できるアサーションについて説明します。
- API Discovery
  Web サービスのリクエストとレスポンスを記録する方法を説明します。

その他のチュートリアル

製品の使用を開始するのに役立つチュートリアルがいくつかあります。

- Data-Driven Functional Tests
  Web サービスのデータ駆動型機能テストを作成する方法について説明します。
  ヒント：さまざまなデータソースからデータを読み取る方法については、他のデータ駆動型のチュートリアルをご覧ください。
- Creating Your First Virtual Service
  仮想サービスの作成方法について説明します。
- Creating Your First Load Test
  Web サービスの負荷テストを作成して、複数のクライアントでどのように機能するかを

確認する方法について説明します。

Getting Started With Security Tests

Web サービスのセキュリティテストを作成する方法について説明します。

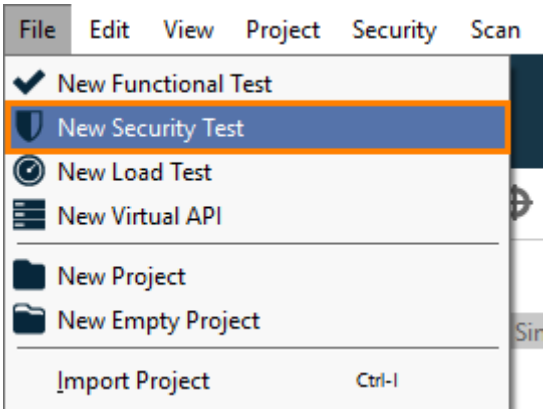
第 2 章: セキュリティ テスト

このチュートリアルでは、ReadyAPI SoapUI ツールを使ったセキュリティ テストの作成方法について説明します。

1. セキュリティ テストの作成および実行

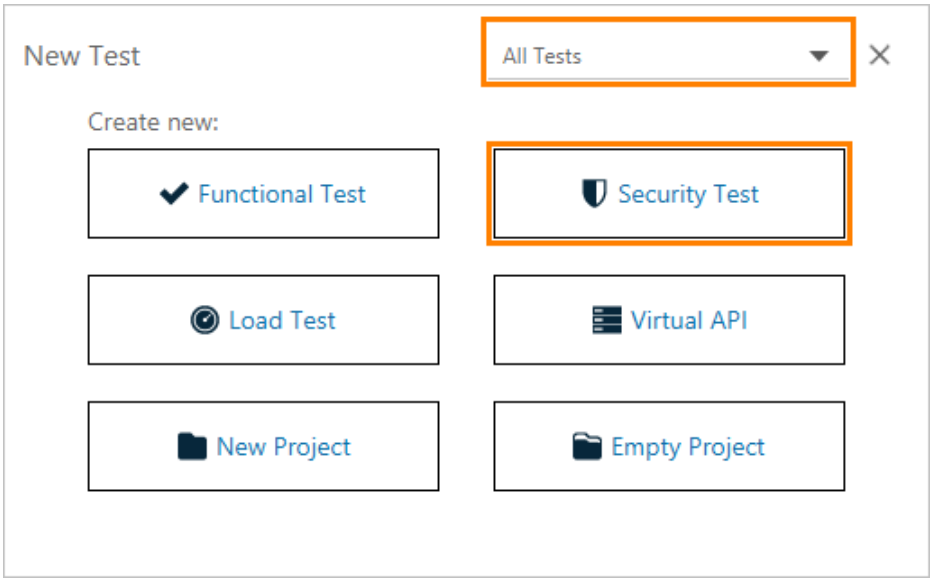
新規のセキュリティ テストを作成するには:

1. **[File] – [New Security Test]** を選択します。



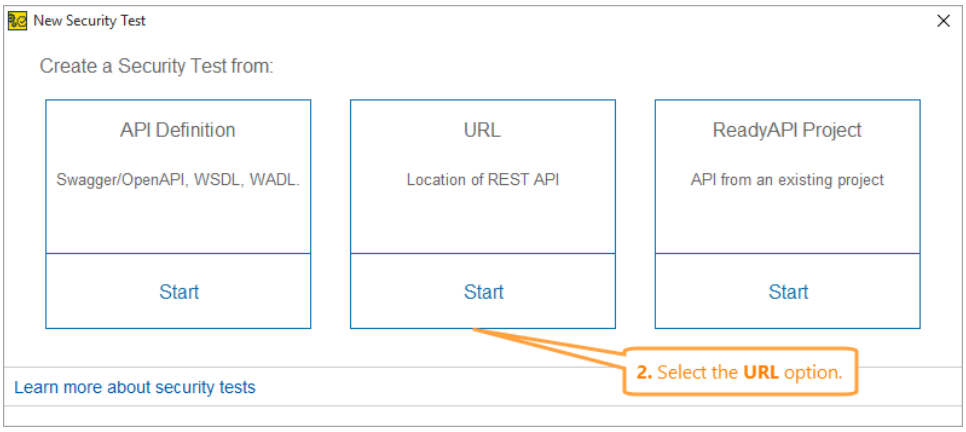
– または –

ダッシュボードの **[New Test]** タイル内で、**[All Tests]** と **[Security Test]** をクリックします。



注意: 画面に表示される New Test タイルは、タイルの内容がお持ちの ReadyAPI ライセンスに依存するため、上記のタイルと異なる場合があります。 詳細については、タイルの説明を参照してください。

2. **[URL]** オプションを選択します。



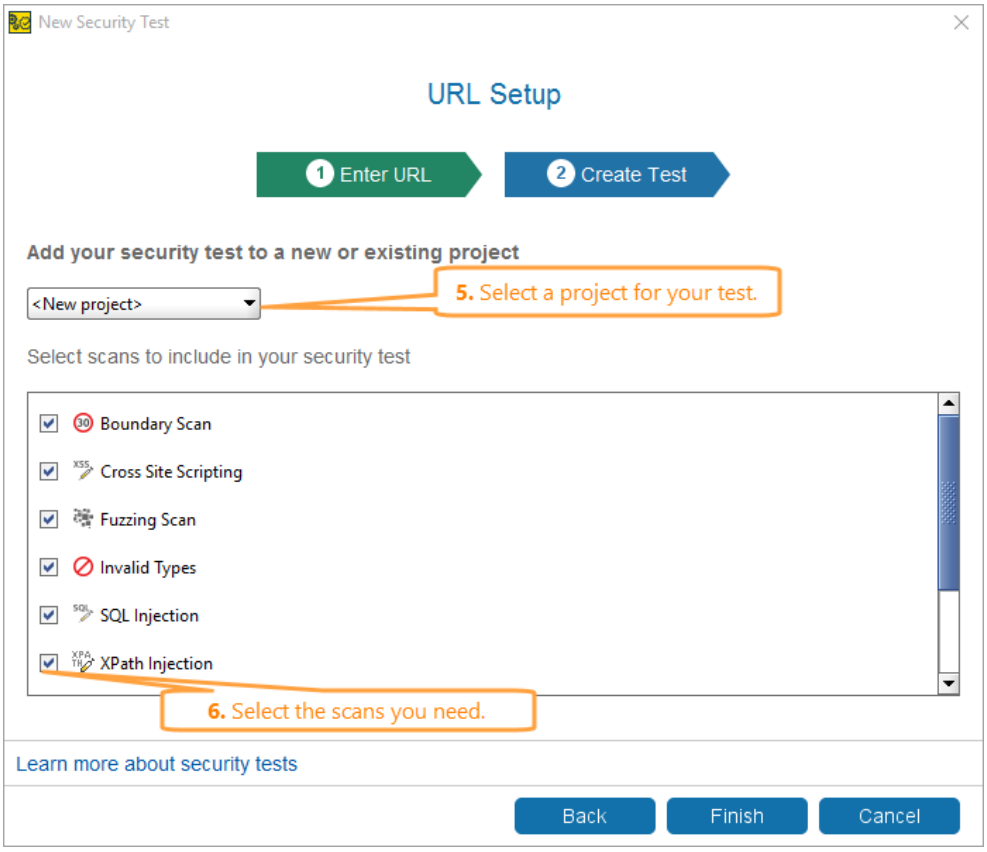
3. 下記の URL を入力し、**[Next]** をクリックします。

```
http://www.datasciencetoolkit.org/maps/api/geocode/xml?
address=1600+Amphitheatre+Parkway,+Mountain+View,+CA&sensor=false
```

ヒント: 複数の URL を同時に指定することができます。別の URL を追加するには、**[Add URL]** をクリックします。

4. ドロップダウン リストから **[HTTP]** メソッドを選択し **[GET]** を選びます。

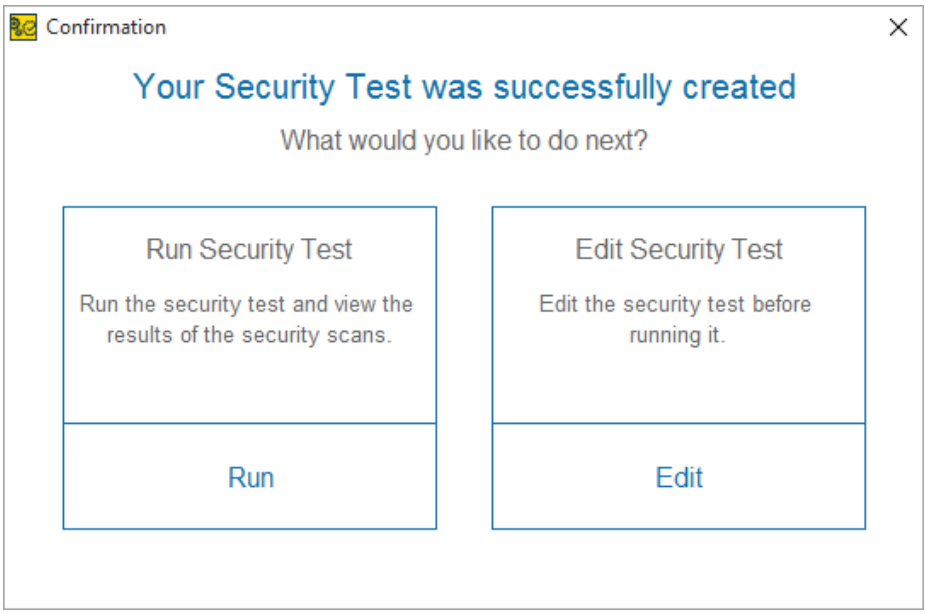
5. テストのためのプロジェクトを選択します。このチュートリアルでは、ドロップダウンリストから **[New Project]** をクリックし新規のプロジェクトを作成します。



6. 必要なスキャンを選択し、**[Finish]** をクリックします。

7. これで、セキュリティ テストの作成が終了しました。セキュリティ テストを直ちに実行するか、または実行する前に変更するかを選択します。今回は、何も設定する必要はありませんので、**[Run Security Test]** オプションを選択します。

   テストの変更を選択した場合は、あとでテストの実行を開始するには **[▶]** をクリックします。



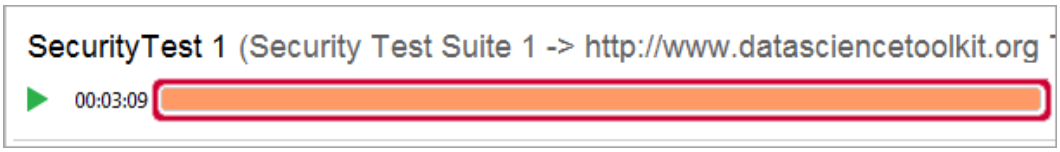
8. ReadyAPI Secure が更新されたリクエストをサービスに送信し、レスポンスを評価します。テストの実行終了後、Secure はテストの実行結果を表示します。 **[View Full Report]** をクリックし、より詳しいレポートを開きます。後でテスト レポートを開く場合は、Report をクリックします。

9. レポートを閉じます。

次のステップでは、セキュリティ テストの結果について説明します。

2. テスト結果の表示

テスト実行中にスキャンに失敗すると、進行状況バーがオレンジ色に変わりますが、スキャンが終了するまでテストは続行します。

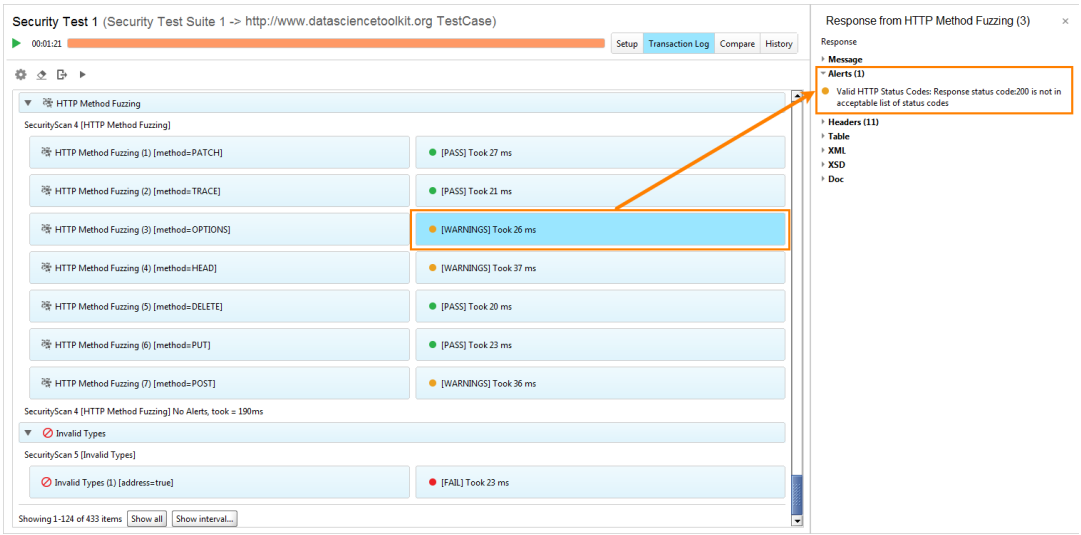


[Transaction log] ページに切り替えます。このページには、セキュリティ テスト中に送信されたリクエストのログがすべて表示されます。

左の列には、サービスに送信されたリクエストの情報が含まれます。
右の列には、サービスからのレスポンスと、リクエストのステータス情報が含まれます。

アサーションが失敗すると、**[FAIL]** というステータスが表示されます。
リクエストが予期しない結果を受け取ると、**[WARNINGS]** ステータスになりますが、失敗にはなりません。

[FAILED] もしくは **[WARNING]** ステータスが表示されているスキャンのレスポンスを選択すると、プロパティ エディターでアラート テストが表示されます。



[Message] セクションを展開すると、このリクエストに対するサービスのレスポンスが表示されます。

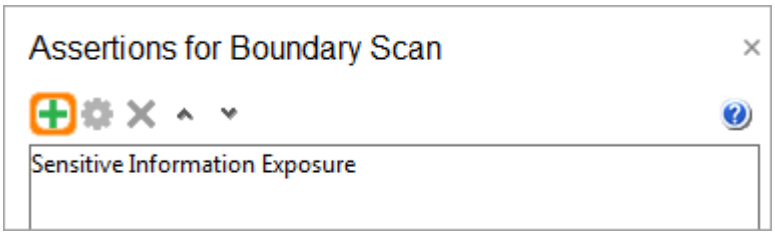
次のステップでは、さらにアサーションを追加し、サービスが SLA 内で返信することを確認します。

3. セキュリティ スキャンへのアサーションの追加

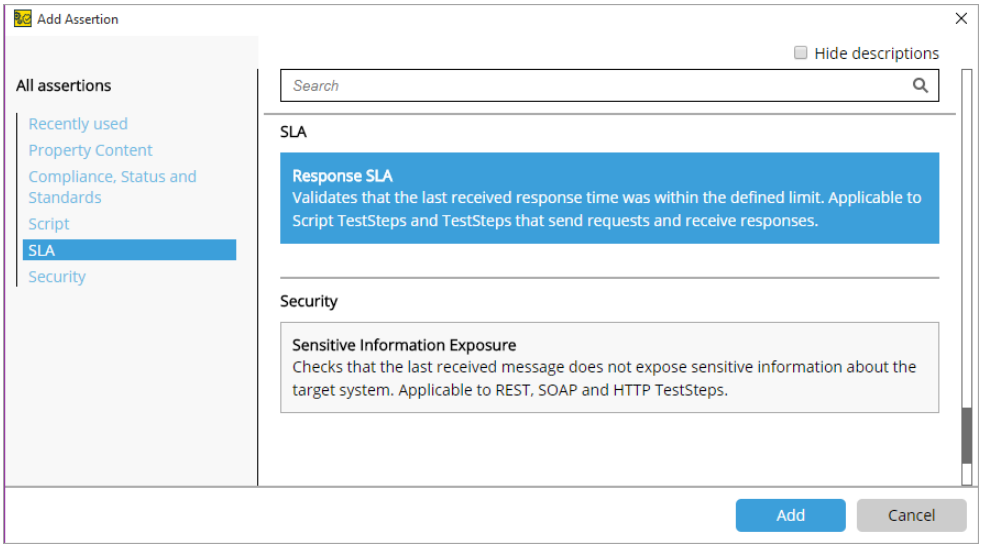
セキュリティ スキャンの終了後、予期しない入力によってサービスのレスポンスに時間がかかりすぎていないことを確認する必要があります。

1. **[Setup]** タブに切り替えます。
2. **[Boundary Scan]** の **[Response]** をクリックします。

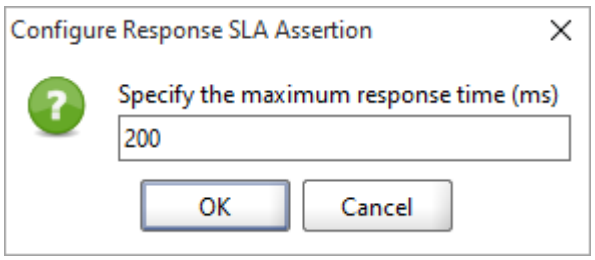
3. アサーション エディターで **[+]** をクリックします。



4. **Add Assertion** ダイアログで **[SLA] – [Response SLA]** を選択し、**[Add]** をクリックします。



5. アサーションがトリガーされるまでのレスポンス タイムを指定し、**[OK]** をクリックします。



これで、スキャン中のレスポンスが 200 ミリ秒以上の時間を要した場合、アサーションがトリガーされてセキュリティ テストは失敗となります。

次のステップ

Secure のチュートリアルはこれで終了です。いくつかのヘルプ トピックを下に示します。

Security Scans

使用可能なセキュリティ スキャンについて説明します。

Security Assertions

使用可能なアサーションについて説明します。

Command-Line Runner

セキュリティ テストを自動化する方法を説明します。

Reports

セキュリティ テストの結果に関するレポートの作成方法を説明します。

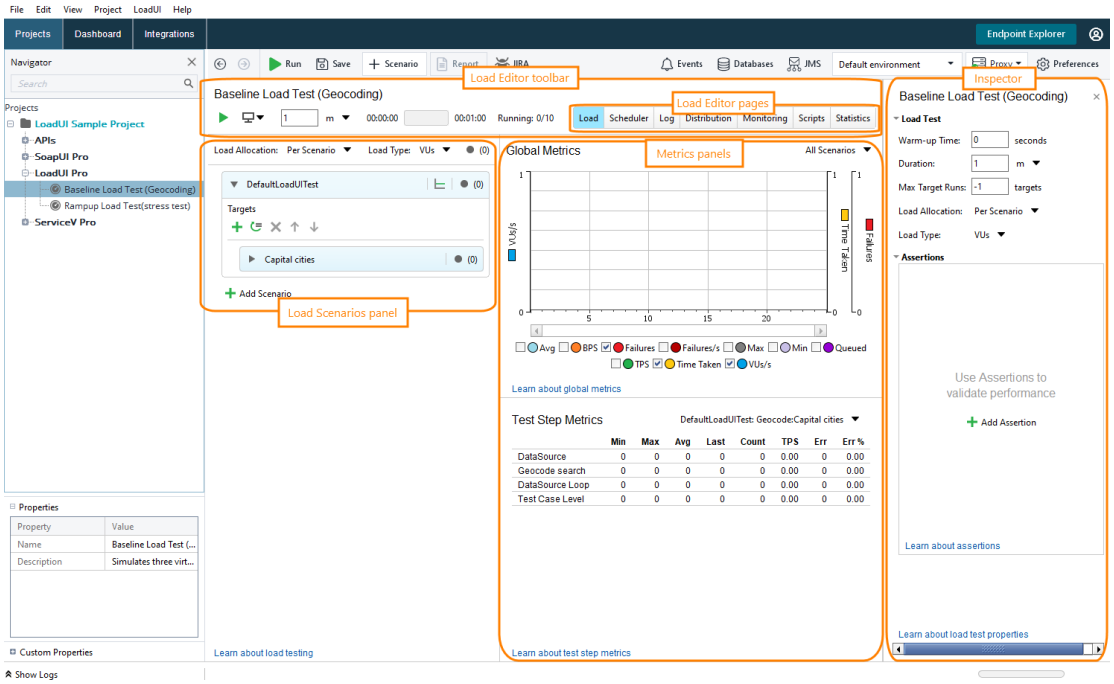
第 3 章: LoadUI スタートガイド

このチュートリアルでは、 Load Test エディターのインターフェイスについて説明し、LoadUI を使った負荷テストの作成方法を示します。このチュートリアルのセクションは、ReadyAPI での負荷プロジェクトの作成方法、簡単なテストの実行 (run a simple test)、およびテスト結果の解析 (analyze the results) に関する説明をします。テストはサンプルの SoapUI テストをエミュレートします。

このチュートリアルは、templates を使って新規の Load Test を作成します。これには、LoadUL Pro または trial ライセンスが必要です。これらを持っていない場合は、URL より新しいテストを手動で作成します。

負荷テスト エディター インターフェイス

LoadUI では、 Load Test エディターを使って負荷テストを表示、編集、実行できます。 Load Test エディターのサンプル表示です。



Load Test Editor ツール バー

エディターのツール バーでは、テストの継続時間 (test duration) を設定します。 LoadUI は、対象となる Web サーバーへのリクエストを指定したタイムアウトが発生するまで、またはその他の理由でテストが停止するまでシミュレートします。タイムアウトが指定されていない場合は、テストを手動で停止する必要があります。

エディターの上部にあるプログレスバー (進行状況バー) は、テストの進行状況を表示します。

Running フィールドには、使用中の仮想ユーザーおよび使用可能な仮想ユーザーの最大数が表示されます。

Load Test Editor ページ

エディターにはいくつかのページがあります。

ページ	説明
Load	このページでは、テストの負荷タイプ (load type) および負荷プロファイル (load profile) の選択、テスト結果を検証するアサーション (assertions) の作成などの負荷テストの設定を行います。テスト設定に関する詳細については「Setting Up Load Tests」セクションを参照してください。
Scheduler	負荷テスト中にシナリオを使用する時間を設定します。詳細については「Load Scheduler」を参照してください。
Log	特定の条件に応じてリクエストを記録します。詳細については「Transaction Log Page」を参照してください。
Distribution	リモート コンピューターでテストを実行するためのリモート エージェントを設定します。詳細については「About Distributed Load Testing」を参照してください。
Monitoring	テスト実行時におけるサーバーのモニタリング設定をします。 収集データは他のテスト結果と一緒に保存されます。詳細については「About Server Monitoring」を参照してください。
Statistics	対象となる Web サービスにおける、テスト間のパフォーマンス メトリックの変化を確認できます。テスト実行の統計情報に関する詳細は、「Test Results」を参照してください。
Scripts	テストの開始時および終了時に使用されるスクリプトを設定します。詳細については「Setup and Teardown Scripts」を参照してください。

LoadUI 基本機能では、**Load** ページのみにアクセスできます。その他のページを使用するには、LoadUI Pro ライセンスが必要になります。

Load Scenarios パネル

このパネルでは、負荷タイプを選択し、負荷テストのシナリオを設定できます。

[Load Type] ドロップダウン リストで、負荷テストで使用する load type を選択します。

- **[VUs]** - LoadUI が、対象となる Web サービスにリクエストを送信する仮想ユーザーをシミュレートします。
- **[Rate]** - LoadUI が、対象となる Web サービスに指定した割合でリクエストを送信します。

シミュレートした負荷を編集するには、編集する負荷シナリオをクリックし、**Editor** パネルの **[Scenario]** セクションで値を編集します。

アサーションを作成するには、アサーションを作成するシナリオまたは TestStep をクリックし、**Editor** パネルで作成します。

Metrics パネル

Global Metrics パネルでは、全体的なテスト結果のグラフが表示されます。リクエストの処理にかかる最大時間を確認できます。 これらのグラフに表示されるメトリックのテストアサーションも確認できます。

Test Steps Metrics パネルは、各 TestStep の詳細メトリックを表示します。たとえば、ステップを完了するのに要する最短および最長時間を確認できます。

また、ドロップダウン リストを使って、テスト実行中に使用した別のシナリオのメトリックに切り替えることもできます。

> これらのパネルに表示される結果は、進行中のテストのみです。過去のテスト情報、またはその他のメトリックを参照するには、Statistics ページのグラフを使用します。

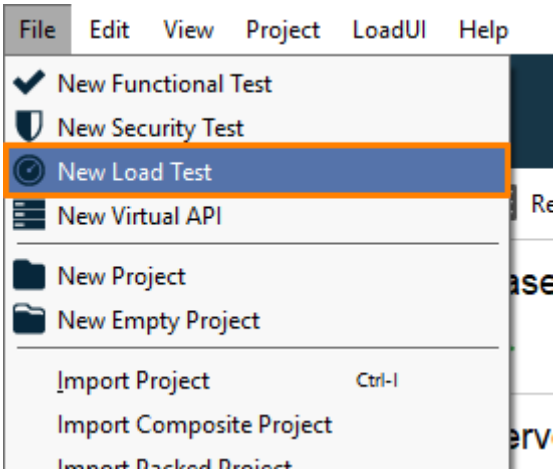
Inspector パネル

シナリオ、TestCase、または TestStep レスポンスをクリックすると、Inspector パネルが表示されます。このパネルでは、選択したオブジェクトを編集できます。たとえば、負荷シナリオの負荷プロファイルを設定したり、TestStep レスポンスにアサーションを追加したりできます。

1. 新規の負荷テストの作成

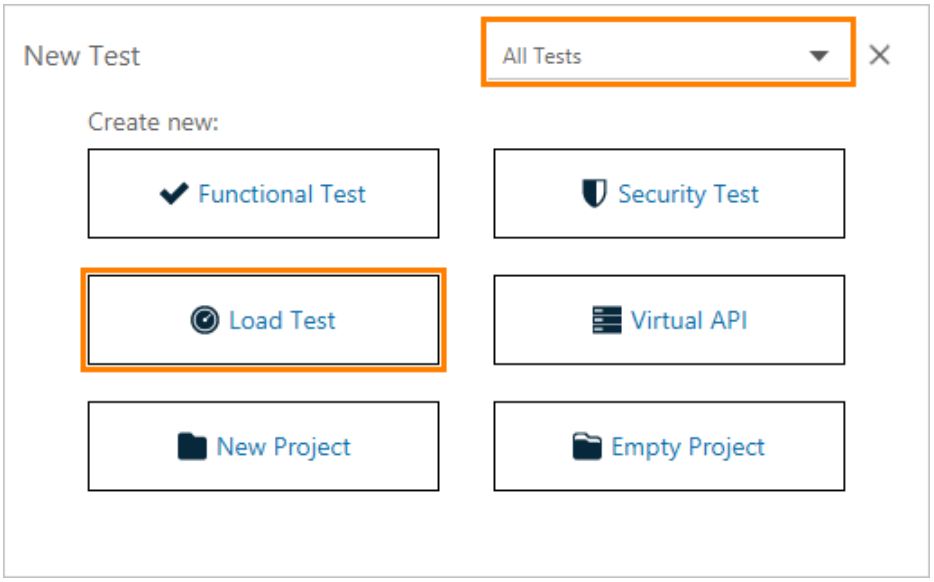
負荷テストを LoadUI で作成するには:

1. **[File] – [New] – [Load Test]** を選択します。



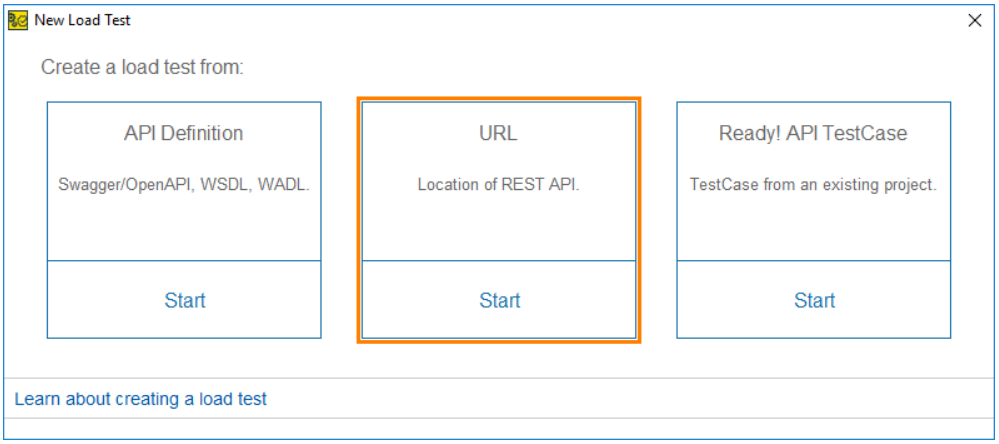
– または –

ダッシュボードで、**[New Test]** タイル内の **[All Tests]** を選択し、**[Load Test]** をクリックします。



注意: 画面に表示される **New Test** タイルは、タイルの内容がお持ちの ReadyAPI ライセンスに依存するため、上記のタイルと異なる場合があります。 詳細については、タイルの説明を参照してください。

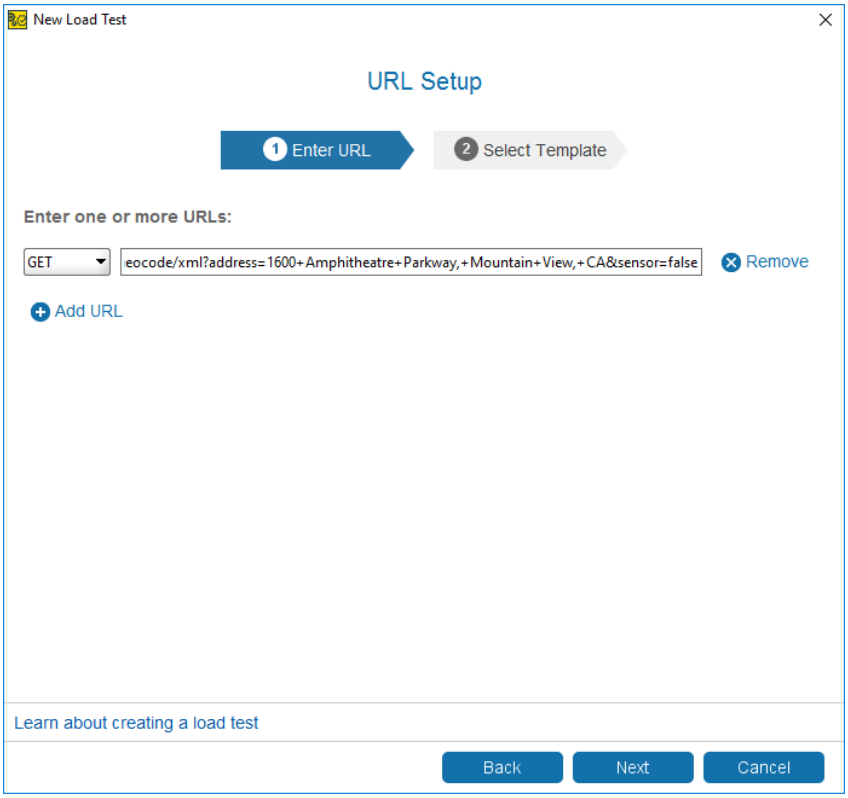
2. **[URL]** を選択します。



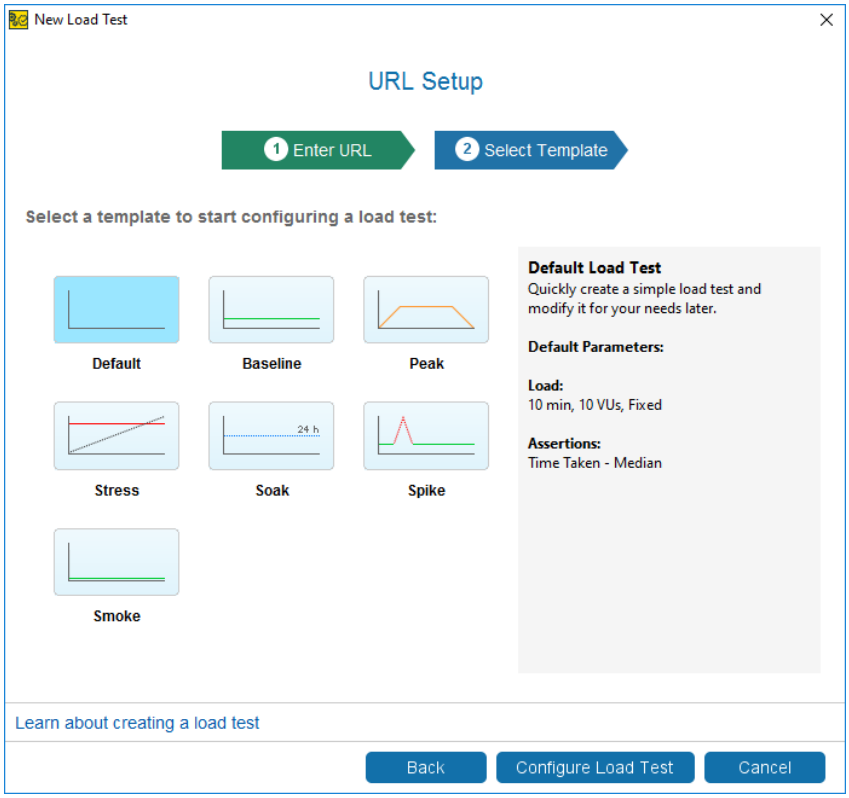
ヒント: 現在のワークスペースに他のプロジェクトがある場合、ReadyAP は既存のプロジェクトのテストケースからテストを作成することを提案します。 このチュートリアルでは、URL から負荷テストを作成する必要があります。

3. 下記の URL を入力し、**[Next]** をクリックします。

```
http://www.datasciencetoolkit.org/maps/api/geocode/xml?
address=1600+Amphitheatre+Parkway,+Mountain+View,+CA&sensor=false
```

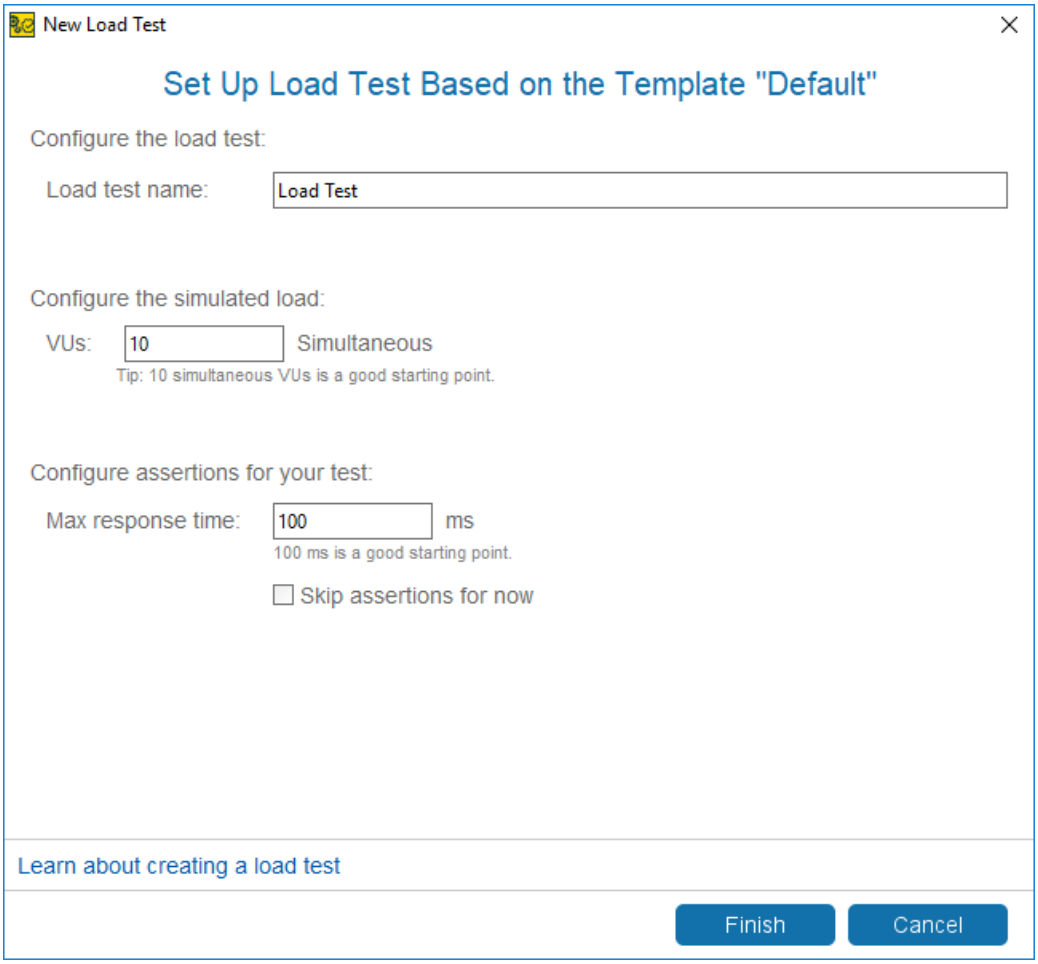


4. **[Default]** テンプレートを選択し、**[Configure Load Test]** をクリックします。

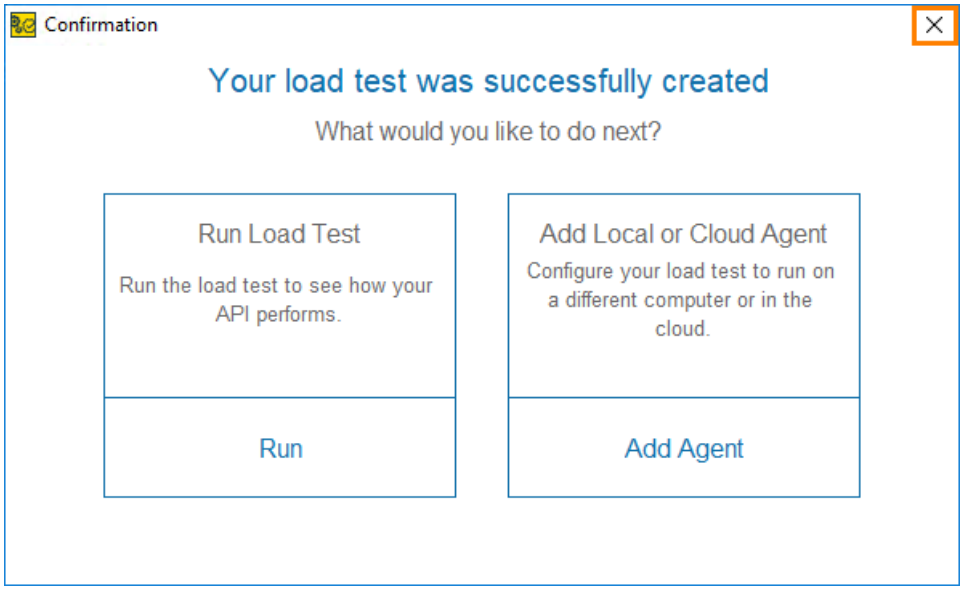


他のテンプレートを使用する場合は、LoadUI Pro の評価版または製品ライセンスが必要です。

5. 今回は、テンプレートをデフォルト値のままにし、**[Finish]** をクリックします。



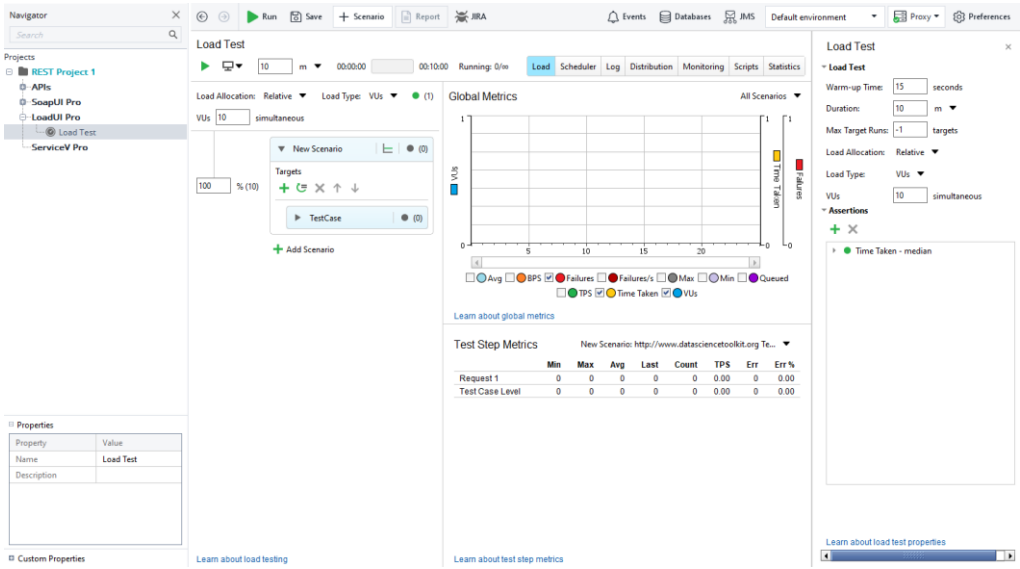
6. LoadUI が負荷テストを作成し、**[Run Load Test]** または **[Add Local or Cloud Agent]** を選択する画面が表示されます。テストを実行する前にこのウィンドウを閉じ、負荷テストを変更することができます。このチュートリアルでは、ウィンドウを閉じ、テストを開始する前にテストを変更します。



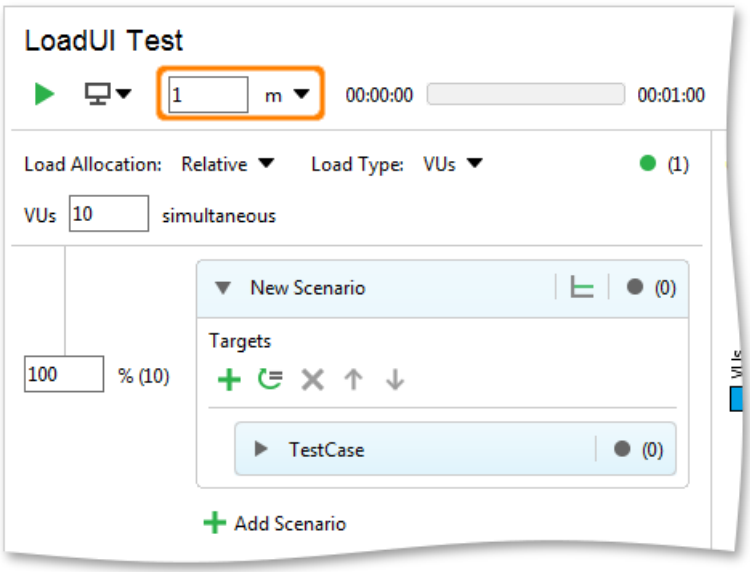
これで、作成した負荷テストの設定または実行ができます。

2. 作成した負荷テストの実行

デフォルトでは、新規作成した負荷テストは選択したテンプレート設定を使用します。



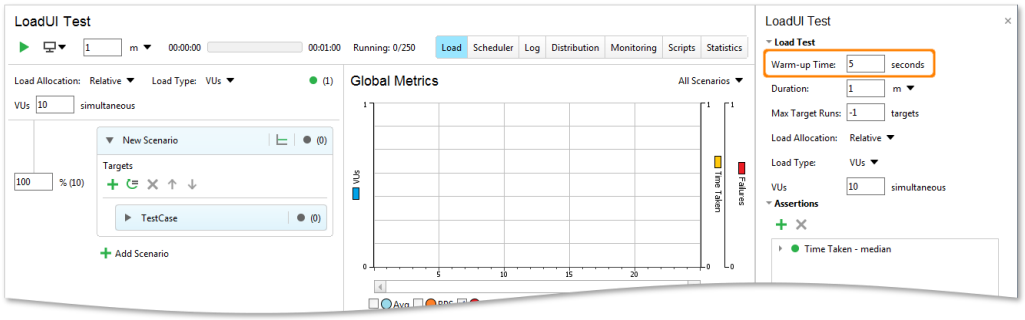
- テスト継続時間を 1 分に変更するため、負荷時間フィールドに **1** を入力します。



- 必要なプロセスをスタートするのに時間がかかるため、最初の数秒間はリクエストに対するサーバーのレスポンスが正しく反映されません。そのような結果を記録することを回避するために、テストのウォーム アップに要する時間を指定します。

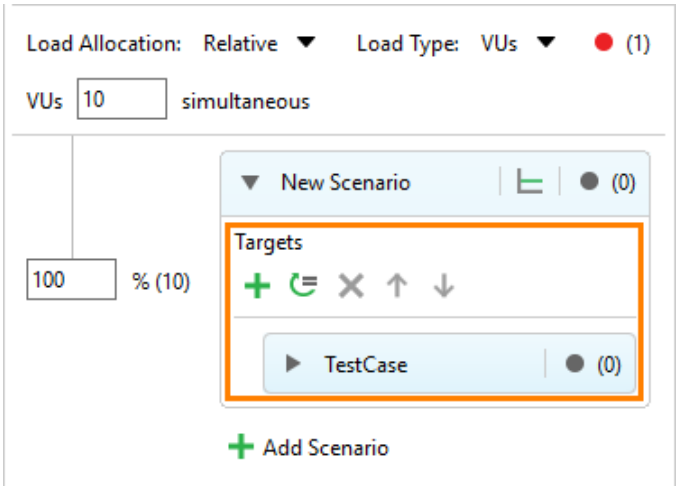
Scenarios パネルの任意の場所をクリックして Load Test エディターを開きます。

[Warm-up Time] フィールドに 5 秒と入力します。

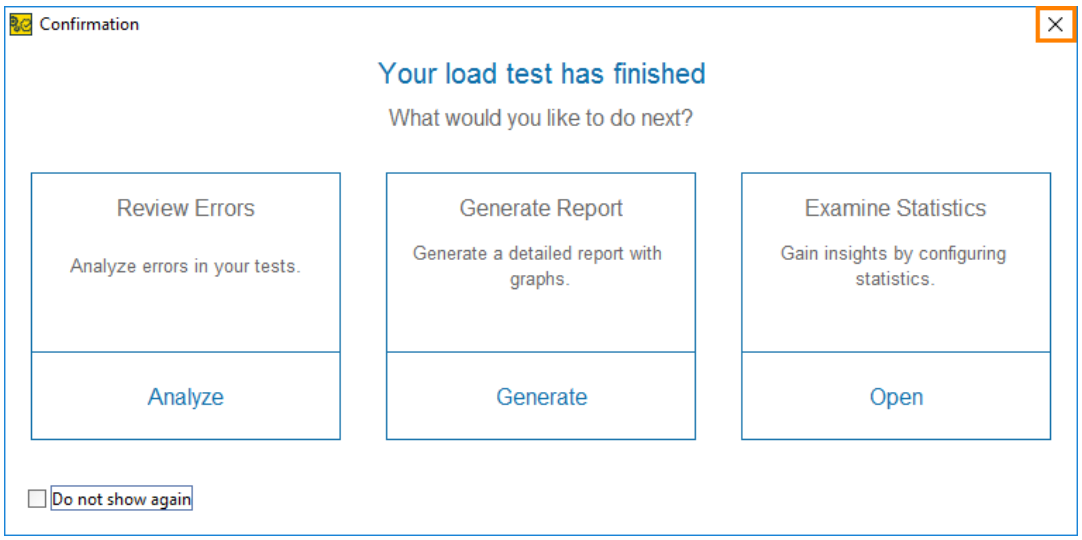


[Run] をクリックしてテストを実行します。

テストの実行には、ユーザーの操作は不要です。LoadUI は、基になる SoapUI の TestCase の TestSteps をシミュレートします (エディターの **[Targets]** セクションで名前を確認できます)。対象となる Web サービスにリクエストを送信し、レスポンスを取得します。テスト開始後には、テスト結果が表示されます。LoadUI では、テスト終了後だけでなく、テスト実行中にも結果が表示されます。

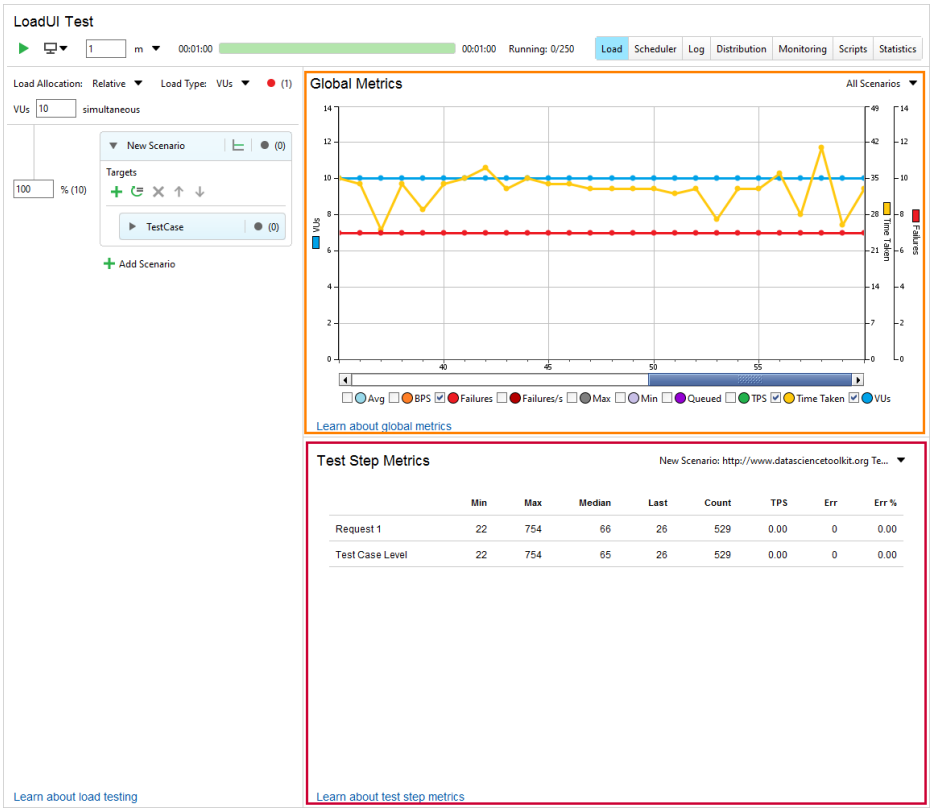


テストの実行が終了すると、LoadUI は **Confirmation** ダイアログを表示し、エラーの確認、レポートの作成、または統計の調査を提案します。 このチュートリアルでは、手動で行うため、ダイアログを閉じます。


Confirmation
Your load test has finished
What would you like to do next?
Review Errors
Analyze errors in your tests.
Analyze
Generate Report
Generate a detailed report with graphs.
Generate
Examine Statistics
Gain insights by configuring statistics.
Open
Do not show again

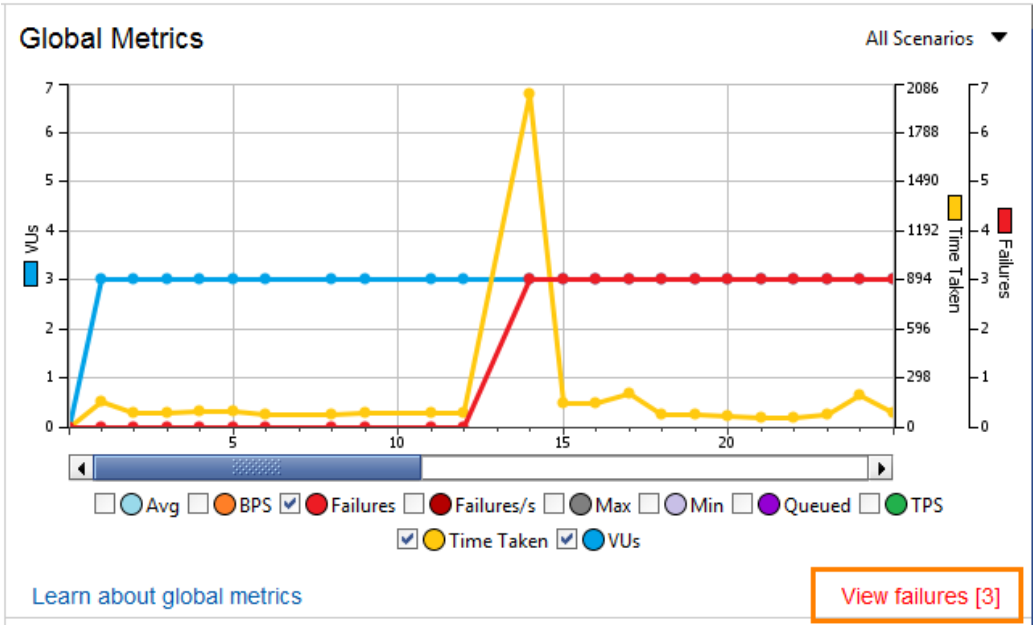
3. テスト結果の表示

LoadUI が負荷テストを実行中に、サービス パフォーマンスのデータが収集されます。収集されたデータは、Global Metrics グラフ、Test Steps Metrics テーブル、および Statistics ページで表示されます。



[Global Metrics] グラフでは、対象となる Web サービスのさまざまなパフォーマンスメトリックの経時的な変化が表示されます。**VUs/s** (1 秒ごとにサーバーに到達するユーザー数)、**Time taken** (各リクエストに要した時間)、**TPS** (1 秒あたりのトランザクション数) および **BPS** (バイト/秒) メトリックを分析し、対象となるサービスのパフォーマンス情報を得られます。

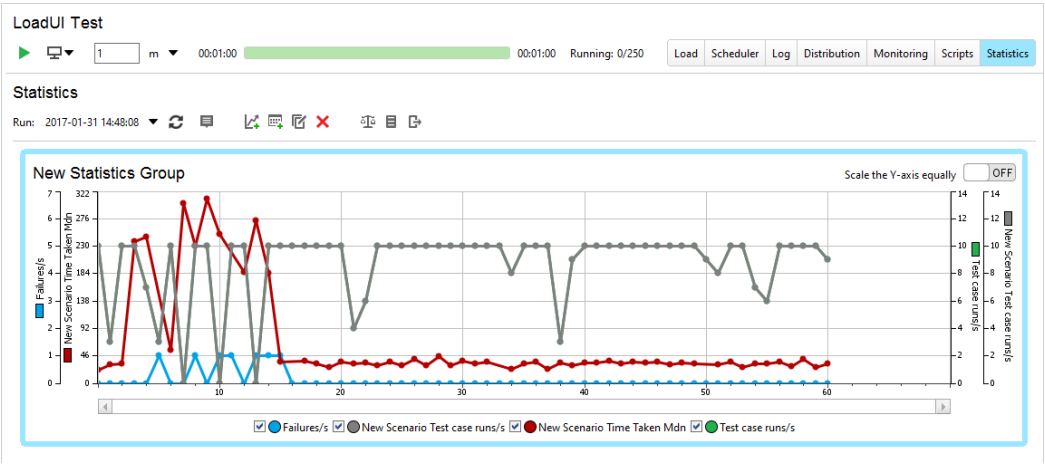
テスト中にエラーが発生すると、LoadUI はグローバル メトリックグラフの下にメッセージを表示します。 メッセージをクリックして、**LoadUI Log** タブでエラーの詳細な説明を表示します。



[TestSteps Metrics] テーブルは、各ステップをシミュレートするのに要した最長および最短時間など、個々の TestSteps におけるパフォーマンス値を表示します。

テストの他の統計を表示、確認するには **Statistics** ページに切り替えます。この操作には LoadUI Pro のライセンスが必要となります。 エクセルソフト株式会社 Web サイトから評価版をお申込みください。

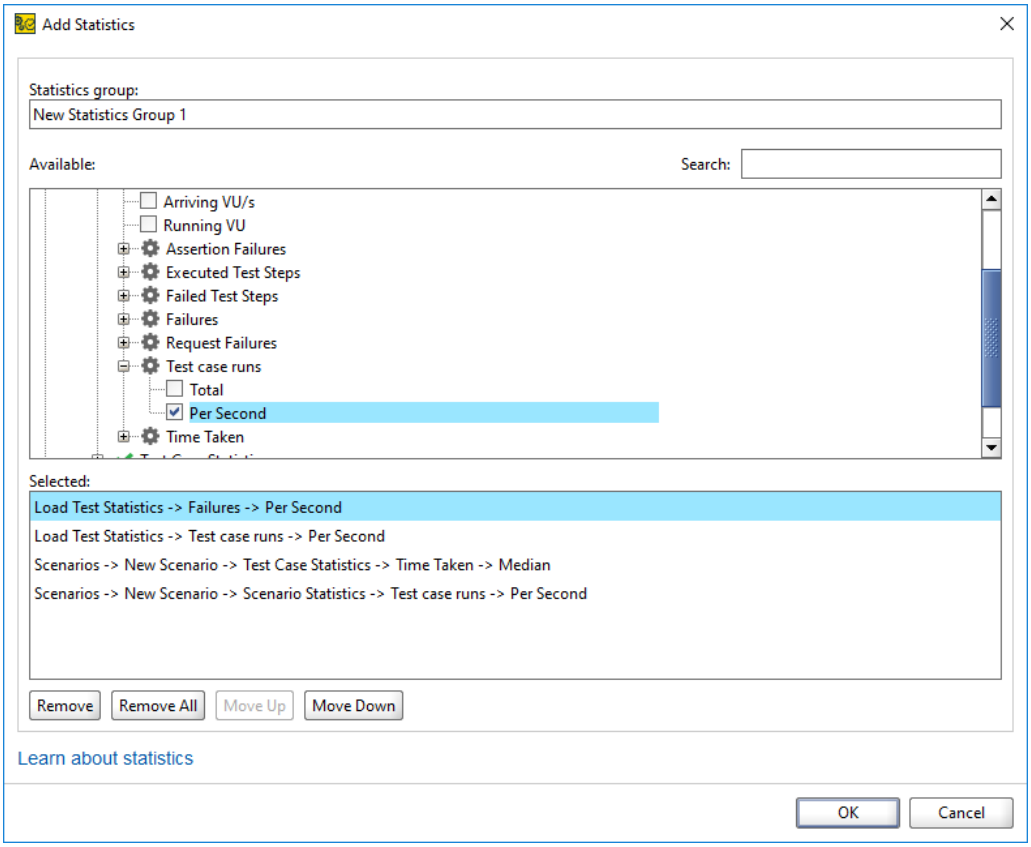
ヒント: このページでは、サーバーのモニターを設定し (configure server monitors)、サーバー メトリックを解析することもできます。



1 秒あたりのリクエスト送信数を表示するグラフを追加しましょう。

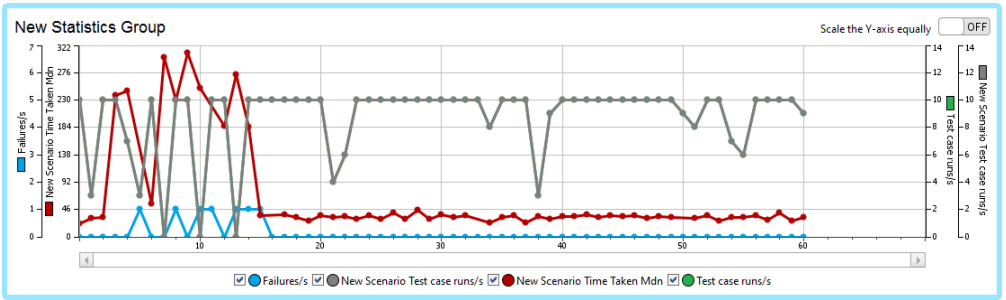
- [ ] をクリックします。

- **Add Statistic** ダイアログで **[Scenarios] - [ScenarioName] - [Scenario Statistics] - [Test case runs] - [Per Second]** を選択し、**[OK]** をクリックします。

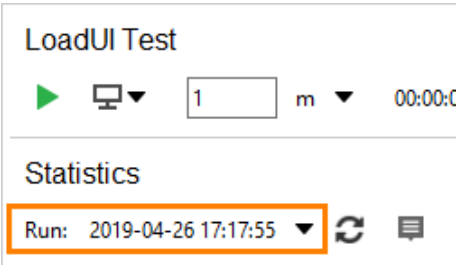


Search フィールドに *Test case runs* と入力すると、迅速に統計情報を探せます。

LoadUI の **Statistics** ページに新しくグラフが追加されます。選択したテストにおいて 1 秒ごとに対象となる Web サービスに送信されるリクエスト数が即座に見られます。



複数のテストを実行した場合、**[Run]** ドロップダウン リストから必要なテストを選択します。デフォルトでは、最後に実行した 5 つのテストが LoadUI 上に保存されます。

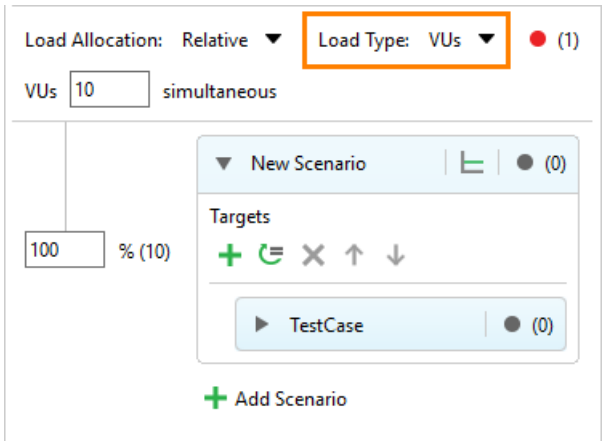


テスト実行中には、統計情報はリアルタイムで更新され、新しくシミュレートしたリクエストが反映されます。

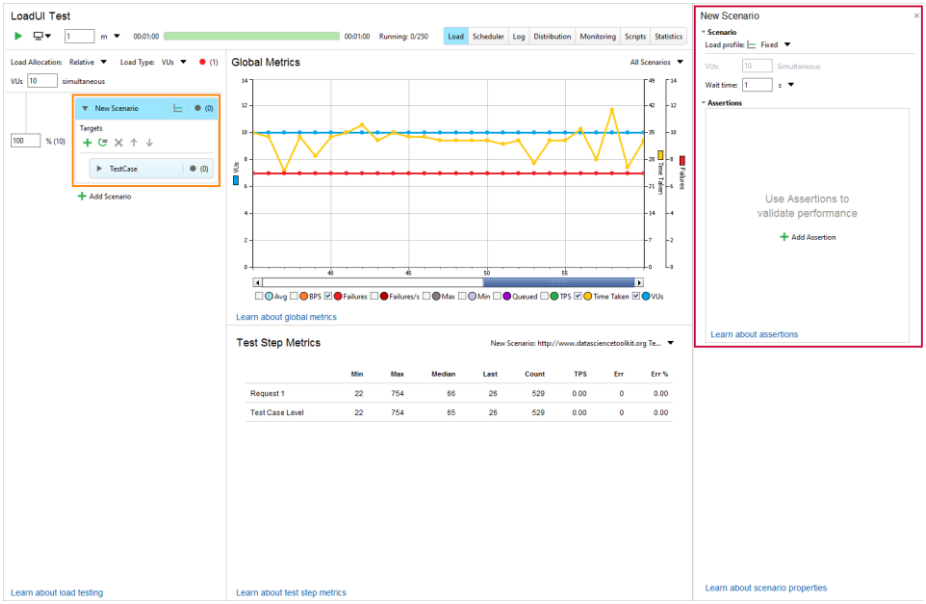
4. 負荷テストの変更

最初の負荷テストの実行に成功しました。いくつかのパラメーターを変更し、異なる負荷プロファイルでテストを再実行してみましょう。

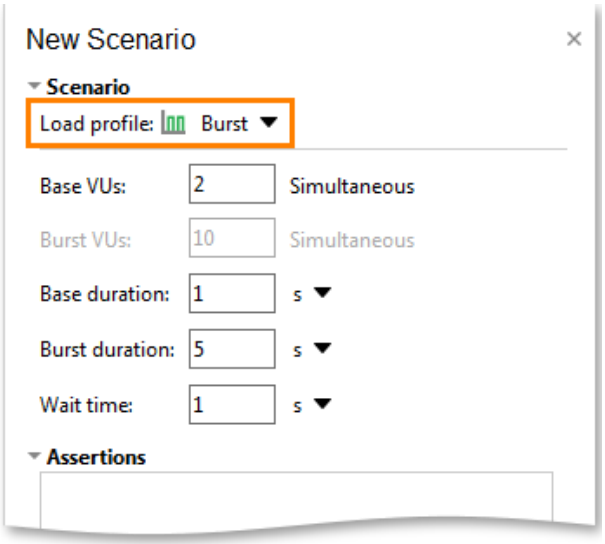
1. **Load Test** エディターの **Load** ページに切り替えます。
2. **[Load Type]** ドロップダウン リストで、**[VUs] (simulate user behavior)** を選択します。この負荷タイプ (load type) を選択すると、負荷プロファイル設定で同時に動作する仮想ユーザーの人数を指定できます。



3. 負荷シナリオをクリックすると、**Editor** パネルが右側に表示されます。

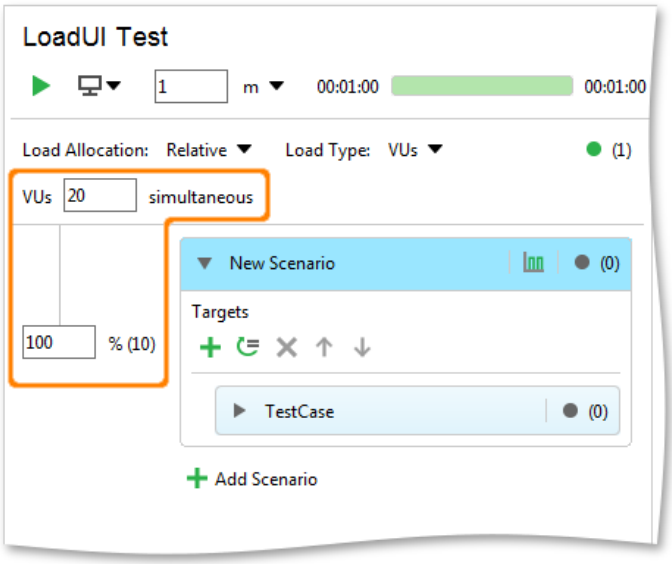


4. **[Load profile]** ドロップダウン リストで、**[Burst]** プロファイルを選択します。プロファイルのアイコンが表示するとおりに、テスト実行中に異なる負荷をシミュレートします。



5. **[Base VUs]** には、たとえば 5 ユーザーのように、シミュレートする仮想ユーザーの最少人数を入力します。

6. **[Burst VUs]** では、シミュレートする仮想ユーザーの最多人数が表示されます。負荷テストの設定から計算されます。このチュートリアルでは、一つのシナリオしかないため、すべての仮想ユーザーをカウントします。たとえば、シミュレートする仮想ユーザーの最多人数を 10 ユーザーと設定します。



LoadUI の基本バージョンでは、最高 10 人までの仮想ユーザーを同時にシミュレートできます。さらに多くの人数をシミュレートするには、LoadUI Pro のライセンスが必要となります。評価版を無料でご利用いただけます。

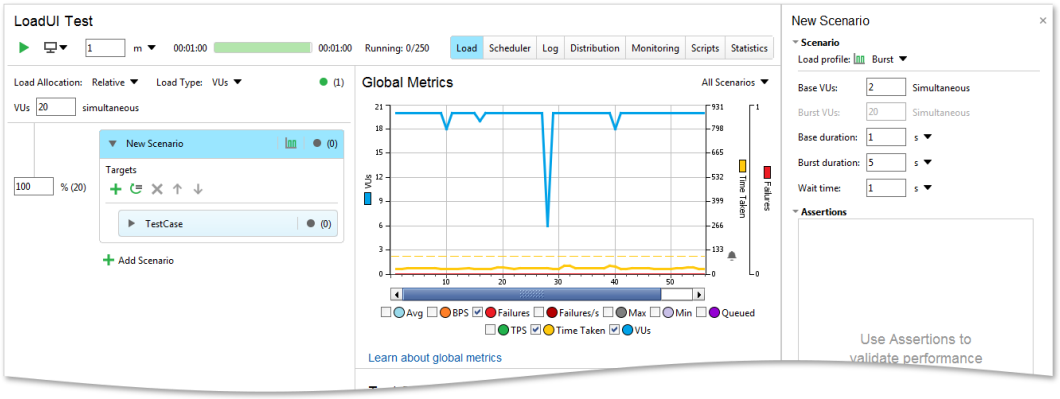
7. **[Base duration]** には、たとえば 1 秒のように、LoadUI が最少人数の仮想ユーザーをシミュレートする時間を入力します。

8. **[Burst duration]** には、たとえば 5 秒のように、LoadUI が最多人数の仮想ユーザーをシミュレートする時間を入力します。

9. **[Wait time]** には、たとえば 1 秒のように、各仮想ユーザーのテストを再実行するまでの LoadUI の待ち時間を入力します。

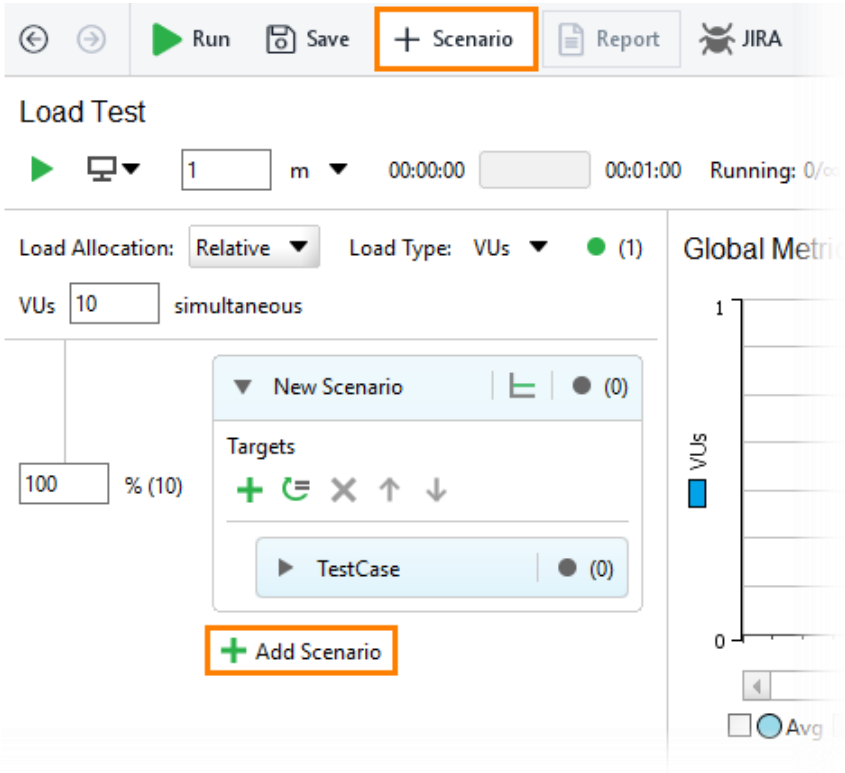
   再実行の必要な理由: 何人かの仮想ユーザーのシミュレーションが終了すると、同時使用ユーザーの人数が減ります。この人数がプロファイルに指定された人数より少ない場合、LoadUI は必要人数に達するまで新規の仮想ユーザーを増やします。

10. テストを実行します。テスト実行中に、LoadUI で指定された最多人数に達するまで仮想ユーザーの人数が増えます。その後、LoadUI はテストが終了するまで仮想ユーザーの人数を増減し続けます。



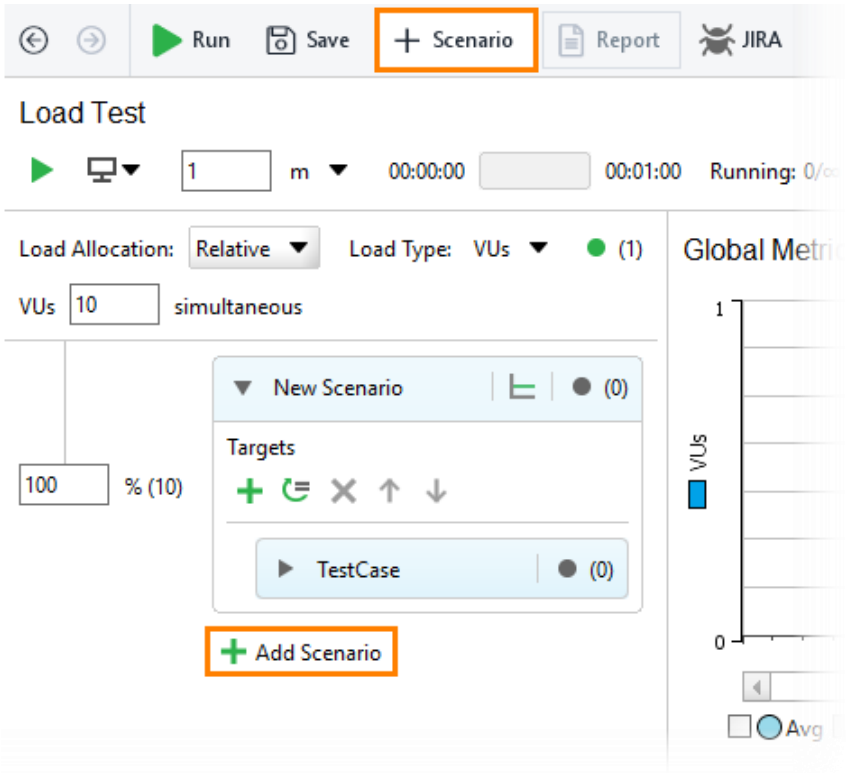
11. テスト結果をもとに、対象となる Web サービスのパフォーマンスの負荷下における変化を確認します。

LoadUI ツール バーの **[Add Scenario]** もしくは **[+ Scenario]** をクリックしてテストにシナリオを追加します。このシナリオは最初のシナリオと並行してシミュレートされるため、複雑な負荷形状を作成できます。上記と同じ方法で設定してください。


Run
Save
Scenario
Report
JIRA
Load Test
1
m
00:00:00
00:01:00
Running:
Load Allocation:
Relative
Load Type:
VUs
(1)
Global Metr
VUs
10
simultaneous
New Scenario
(0)
Targets
100
% (10)
TestCase
(0)
Add Scenario
1
VUs
0
Avg

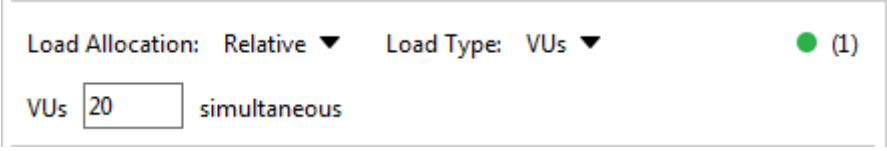
5. 複数のシナリオの実行

いくつかの TestCases を並行してシミュレートするには、負荷テストにシナリオを追加します。**Load Scenarios** パネルの **[Add Scenario]** ボタンか、LoadUI ツール バーの [+ Scenario]をクリックします。「LoadTests の変更」トピックでの説明と同じ要領で新しいシナリオを設定すると、LoadUI でシナリオが並行してシミュレートされます。



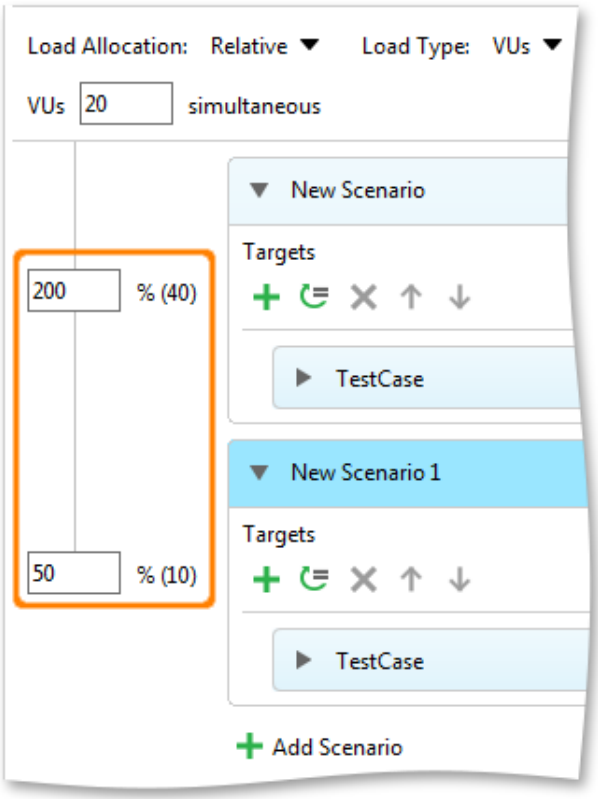
いくつかのシナリオを同時に使用する場合、各シナリオに個別に負荷を割り当てるのではなく、1 つのフィールドから負荷を割り当てることができます。**[Load Allocation]** ドロップダウン リストから **[Relative] (percentage of VUs)** を選択します。

下の新しい VUs フィールドで、たとえば 10 などのように、シナリオが参照するベースとなるユーザー数を指定します。



各シナリオの隣のフィールドに、シミュレートされるベース ユーザーの比率を入力します。

- 最初のシナリオでは、負荷を 200% に設定します。これで、このシナリオの Burst プロファイルは 20 人の仮想ユーザーでシミュレートします。ベースとなる仮想ユーザー数はシナリオで設定されます。
- 2 つめのシナリオでは、負荷を 50% に設定します。これで、このシナリオは 5 人の仮想ユーザーでシミュレートします。

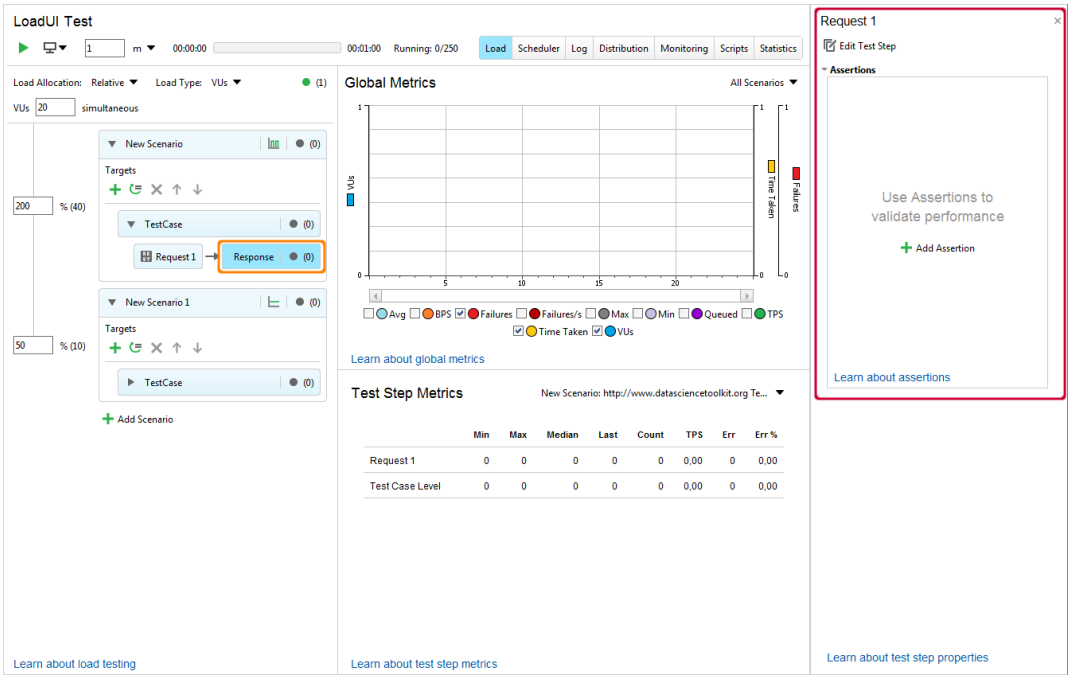


6. 負荷テストへのアサーションの追加

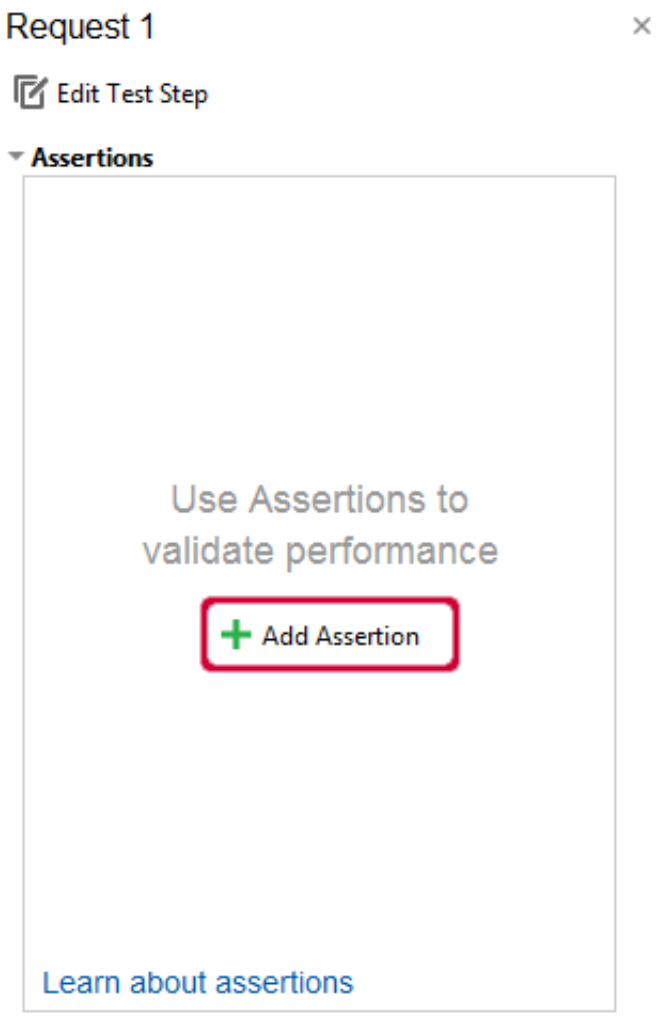
アサーション (Assertions) とは、テスト実行中に対象となる Web サービスの機能を確認するテスト結果に適用される検証規則です。

1 つの TestStep の平均シミュレーション タイムが制限時間を超過しないことを確認するアサーションを作成しましょう。

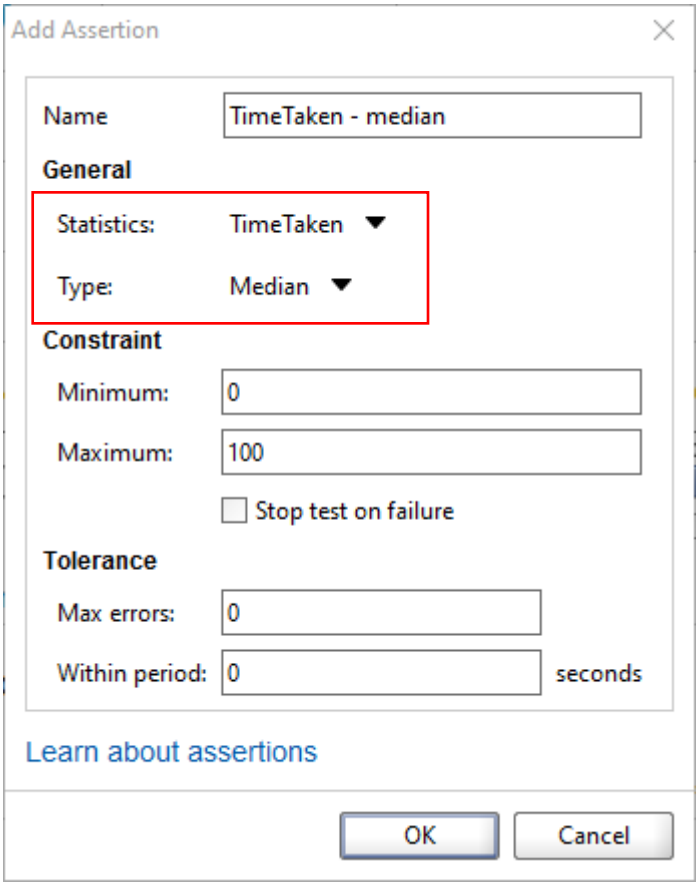
1. TestCase を広げ、アサーションを作成する TestStep の **[Response]** をクリックします。**[Request]** エディターが表示されます。



2. **[Request]** インスペクターの **[Assertions]** セクションで **[+]** をクリックします。



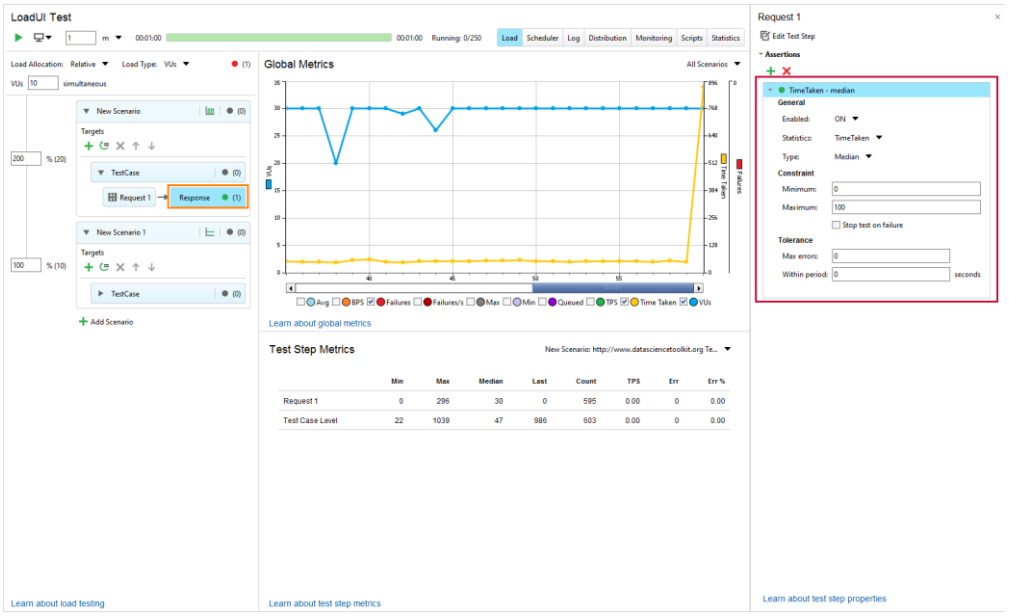
3. 次の **Add Assertion** ダイアログの **[Statistics:]** では **[Time Taken]**、**[Type:]** では **[Median]** をそれぞれのドロップダウン リストから 選択します。



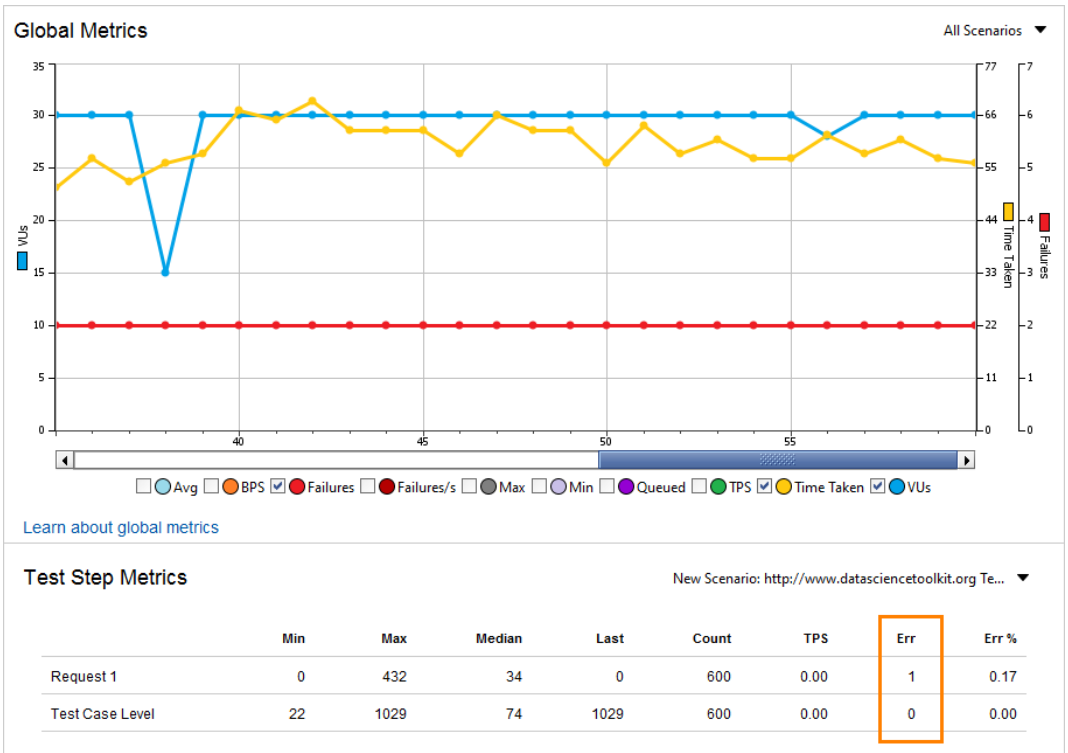
4. その他のアサーション パラメーターを設定します。

   - **[Maximum]** エディット ボックスに TestStep が平均して要する最多回数を入力します。
   - **[Tolerance]** セクションで、エラーをログに記録する前にアサーションがトリガーされる最多回数を設定します。

5. **[OK]** をクリックすると、LoadUI の負荷テストにアサーションが追加されます。



6. 再度テストを実行します。レスポンスに時間がかかりすぎると、LoadUI はエラーをレポートします。アサーション エラーは、**Global Metrics** グラフおよび **TestSteps Metrics** テーブルで表示されます。



	Min	Max	Median	Last	Count	TPS	Err	Err %
Request 1	0	432	34	0	600	0.00	1	0.17
Test Case Level	22	1029	74	1029	600	0.00	0	0.00

次のステップ

LoadUI のチュートリアルはこれで終了です。このスタートガイドが LoadUI の理解に役立つことを願っています。いくつかのヘルプ トピックを下に示します。

Server Monitoring

テスト実行中のサーバーのモニター方法について説明します。

Creating Distributed LoadTests

リモート コンピューターで実行するテストの作成方法を説明します。

Creating Distributed Cloud Tests

クラウド マシンで実行するテストの作成方法を説明します。

Exporting Test Results

LoadUI テストの結果をエクスポートする方法を説明します。

Transaction Log Page

シミュレートされたリクエスト情報のログの仕方について説明します。

Testing Scenarios

負荷シナリオの概要と、ストレス テストの作成など、LoadUI でシミュレートする方法を説明します。

第 4 章: ServiceV スタートガイド

このチュートリアルでは、仮想サービスの作成、設定、実行、および使用方法について説明します。

基本概念の理解

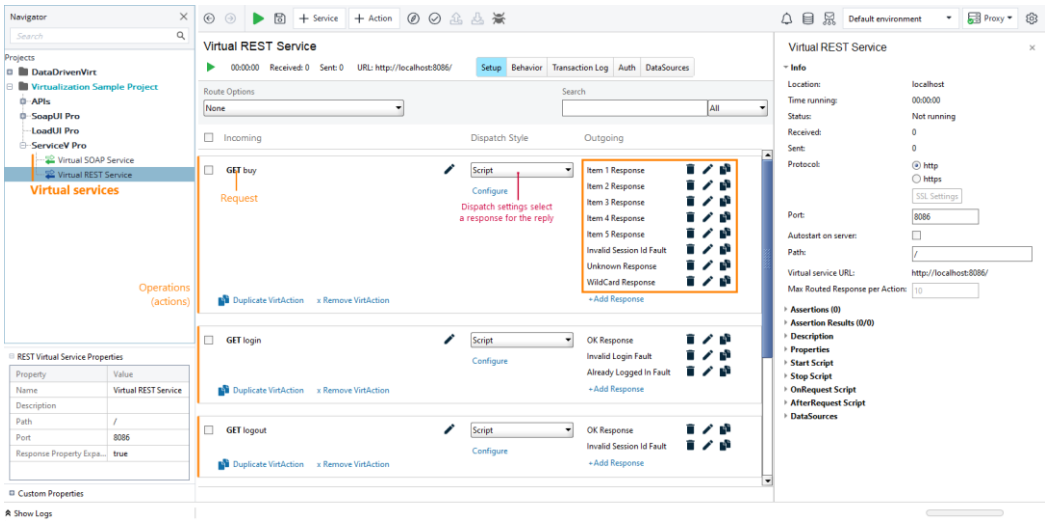
仮想サービス

仮想サービスは、実際の Web サービスのモックです。 サービスが受信するリクエストと、リクエストに応答するためにクライアントに送信するレスポンスを定義します。 仮想サービスは、実際の Web サービスとして機能します。特定の URL とポートで実行され、リクエストを送信でき、レスポンスで応答します。 ただし、仮想サービスはそれ自体を実行することはできず、ReadyAPI または VirtServer (リモートマシンで仮想サービスを実行するには後者が必要) のランナーが必要です。 ServiceV では、仮想サービス**仮想 API** も呼び出します。

プロジェクトには、必要な数の仮想 API を作成できます。 プロジェクトのノードの下のナビゲーターパネルでそれらを表示できます (下の画像を参照)。

オペレーション (アクション)

仮想サービスを作成するとき、サービスが受信する着信メッセージ (リクエスト) とそれらに対するレスポンスを定義します。 これは、仮想サービスエディターで行います。



着信要求と発信応答の組み合わせを**オペレーション**（操作または**アクション**）と呼びます。 ご覧のとおり、リクエストには複数のレスポンスを定義できます。

仮想サービスエディターでオペレーションを手動で指定できます。または、サービスに OpenAPI、Swagger、WADL、または WSDL 仕様がある場合は、ReadyAPI にコマンドを実行して、仮想サービスの作成時にこの仕様からオペレーション定義を読み込むことができます。 サービス操作のリストを取得するもう 1 つの方法は、ライブサービスへのトラフィックを記録することです。ReadyAPI は要求に関する情報を取得し、仮想サービスに操作を作成します。

仕組み

- 仮想サービスがリクエストを受信すると、リクエストに一致する仮想オペレーションを見つけます。 オペレーションを見つけるために、仮想サービスはさまざまなリクエストプロパティを分析します。 正確なアルゴリズムはサービスタイプによって異なります。SOAP 仮想サービスは URL のリクエストされたリソースパスをチェックし、REST 仮想サービスはリクエストタイプ（GET、POST など）と URL パスをチェックし、JMS 仮想サービスは JMS セッションをチェックし、トピック（またはキュー）。詳細については、「仮想サービスの詳細」を参照してください。

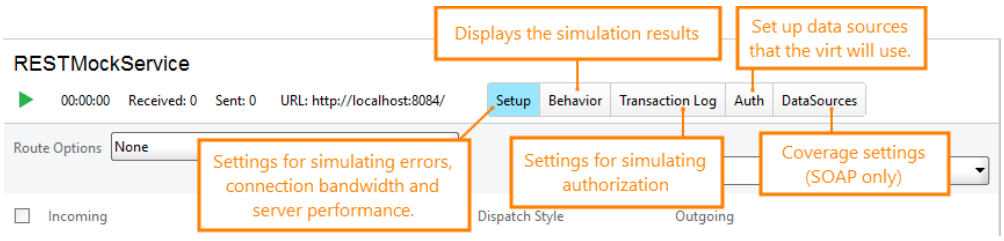
  サービスがオペレーションを見つけると、処理を続行します。 それ以外の場合は、「500 Internal Server Error」エラーを返します。

- 仮想サービスがオペレーションを検出した後、着信要求に対するレスポンスを選択する必要があります。 サービスは、オペレーションに指定した Dispatch 設定に基づいてレスポンスを選択します。 選択は、リクエストパラメータ、リクエストの内容、またはその他の方法に基づきます。 ディスパッチ設定を参照してください。

  仮想サービスは、いくつかのライブ Web サービスにリクエストを転送し、レスポンスを受信してリクエスト送信者に返送できます。 つまり、クライアントとライブサービスの間の透過的なプロキシとして機能できます。 このプロセスは**ルーティング**と呼ばれます。 すべてのスクエストまたは一部のリクエストのみをルーティングするように仮想サービスを構成できます。 このチュートリアルでは、リクエストをルーティングしません。 必要に応じて、仮想サービスに対してこれを実行できます。 詳細については、リクエストのルーティングをご覧ください。

その他

- レスポンスエディターでレスポンスの内容を指定します。 レスポンスには添付ファイルを含めることができます。 プロジェクトやサービスのプロパティを挿入したり、Excel シート、データベース、CSV ファイル、その他のデータソースからデータを読み込んで挿入したりすることもできます。
- サービスエディターページの簡単な説明を次に示します。



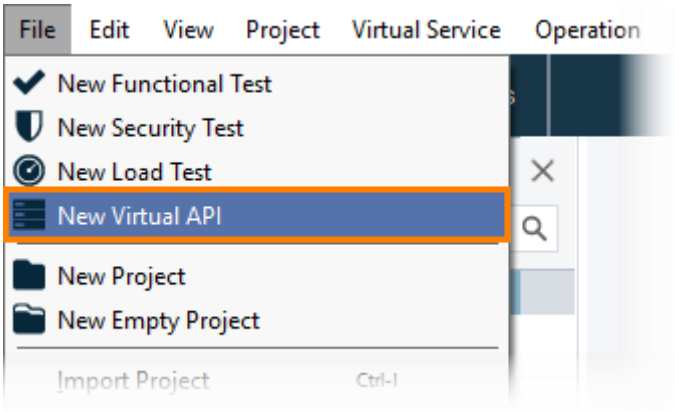
次に、仮想サービスを作成しましょう。

1. 仮想サービスの作成

仮想サービス (virt、あるいは仮想 API) を作成するには:

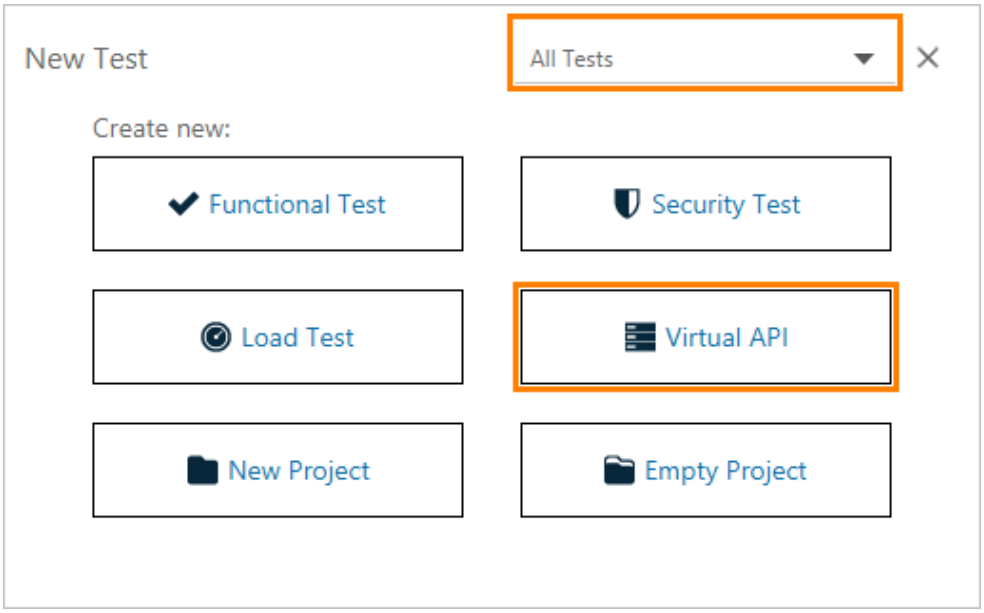
To create a new virtual service (*virtual service* or *virtual API*):

1. メイン メニューから **[File] – [New Virtual API]** を選択します。



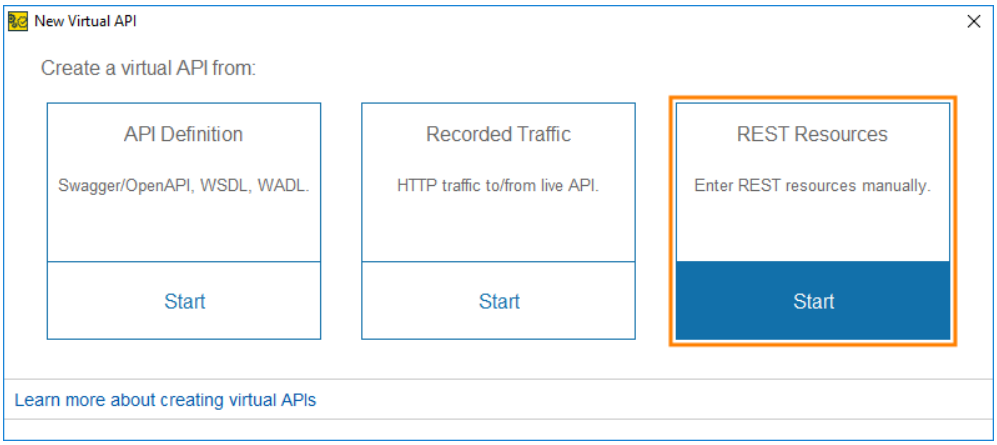
– または –

ダッシュボードの New Test タイルで、**All Tests** を選択して、**Virtual API** をクリックします。



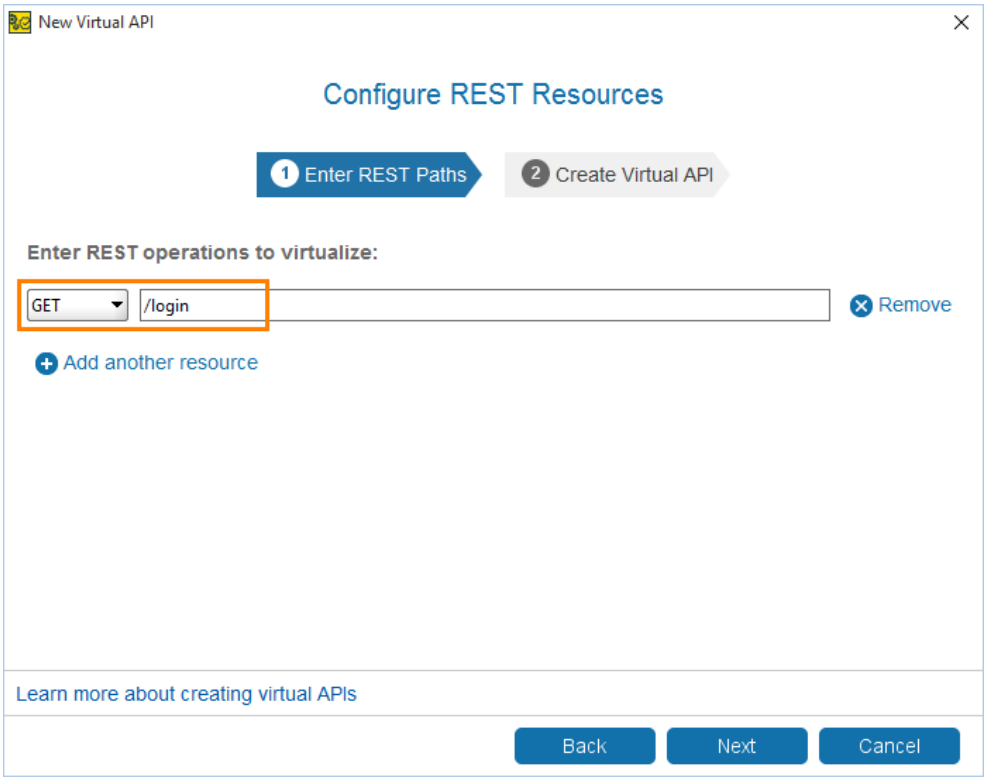
注意: 画面に表示される新しいテストタイルは、タイルの内容がお持ちの ReadyAPI ライセンスに依存するため、上記のものと異なる場合があります。 詳細については、タイルの説明を参照してください。

2. 次の **New Virtual API** ダイアログで、オペレーションを作成する方法を選択します。 **[REST Resources]** の下の **[Start]** をクリックします。これにより、オペレーションを手動で作成します。



API の Swagger、WADL、または WSDL 仕様を持っている場合は、**[API Description]** をクリックします。この場合、ServiceV では仕様から読み込んだ情報に基づきオペレーションを作成します。または、**[Record Traffic]** を選択し、次のウィンドウで現行の API にリクエストを送信します。 ReadyAPI ではトラフィックを記録し、記録したリクエストとレスポンスに基づきオペレーションを作成します。詳細については、「Recording Virtual Services (Discovering)」を参照してください。

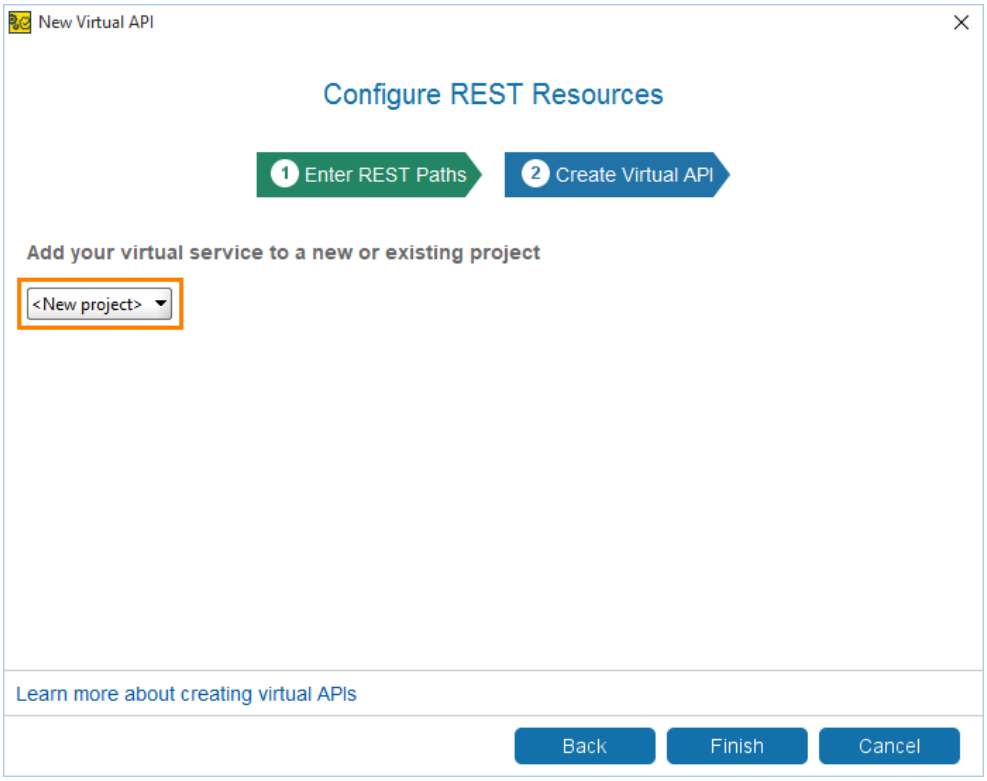
3. 次のダイアログでは、オペレーションを指定し仮想化します。 ドロップダウン リスト内で選択されている **[GET]** オペレーション タイプのデフォルト値をそのままにし、エディット ボックスに **[/login]** と入力します。この際、必ず名前の前にスラッシュ（/）を入れます。



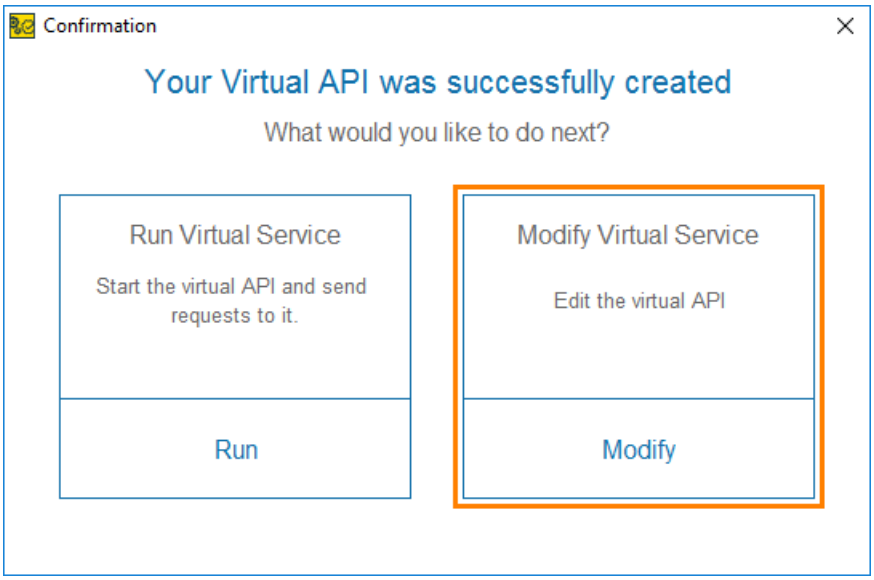
クエリのパラメーターを入力しないでください (これらは、URL パスで ? の後に続きます)。オペレーション パスにはこれらのパラメーターを含めないでください。

仮想サービスエディターではいつでもオペレーションを追加できます。(詳細については、「Creating Requests (Operations)」を参照してください)。 **[Next]** を選択します。

4. 新規の仮想サービスの追加先のプロジェクトを選択します。デフォルトで選択されているドロップダウン リストの **[New Project]** のままにし、**[Finish]** をクリックします。

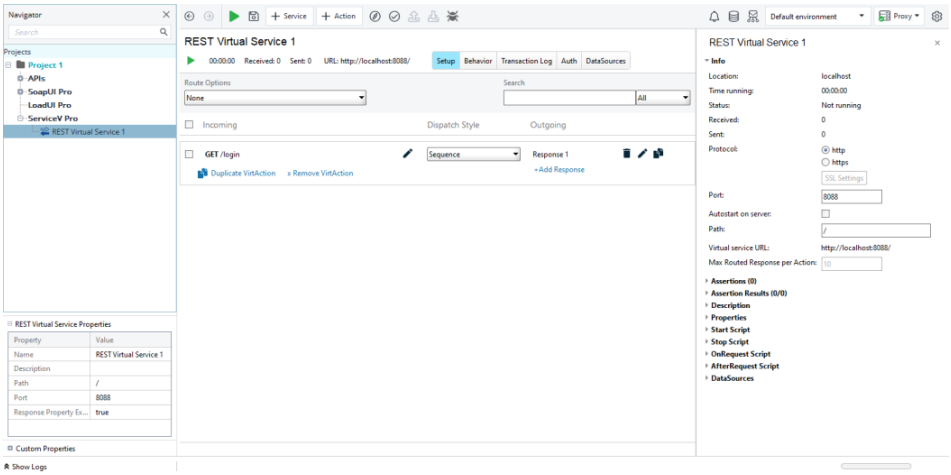


5. ReadyAPI は仮想サービス(virt) を新規作成したプロジェクトに追加します。その後、次の作業の選択を行います。



作成したオペレーションでは、レスポンスが送信されないため、実行する前にサービスを編集します。**[Modify]** をクリックします。

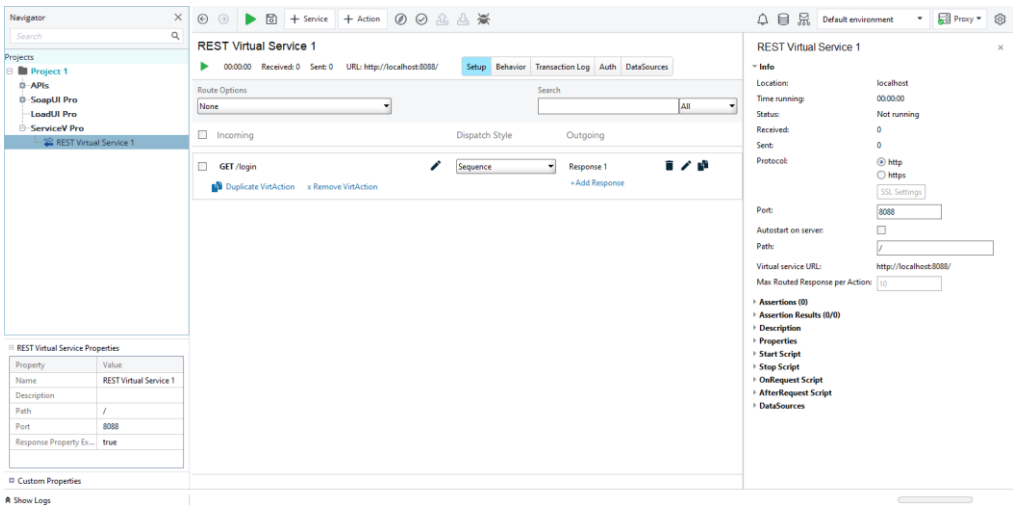
仮想サービスエディターが表示されます。



次のステップでは、仮想サービス機能の設定およびレスポンスの内容を定めます。

2. サービス レスポンスの設定

仮想サービスエディターでは、サービスモックのさまざまな側面を構成できます。 デフォルトでアクティブになっている Setup ページで、オペレーションを作成および削除し、レスポンスを構成できます。 オペレーションごとに、1 つまたは複数の応答を定義できます。仮想サービスは、これらのレスポンスのいずれかを返します。 返されるレスポンスを指定するには、Dispatch 設定を使用します。 次のステップでそれらと協力します。 それでは、`GET /login` オペレーションのレスポンスを作成しましょう。



次のステップ

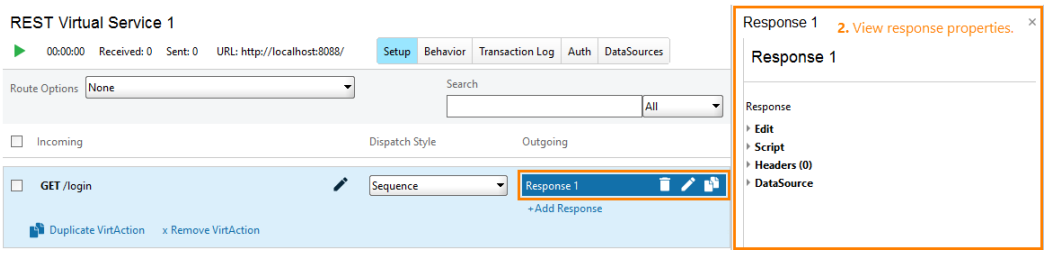
2 つの異なるレスポンスを作成します。一方は、成功した結果を送信し、もう一方はエラー結果を送信します。

次のステップでは、各レスポンスが示す内容を設定します。

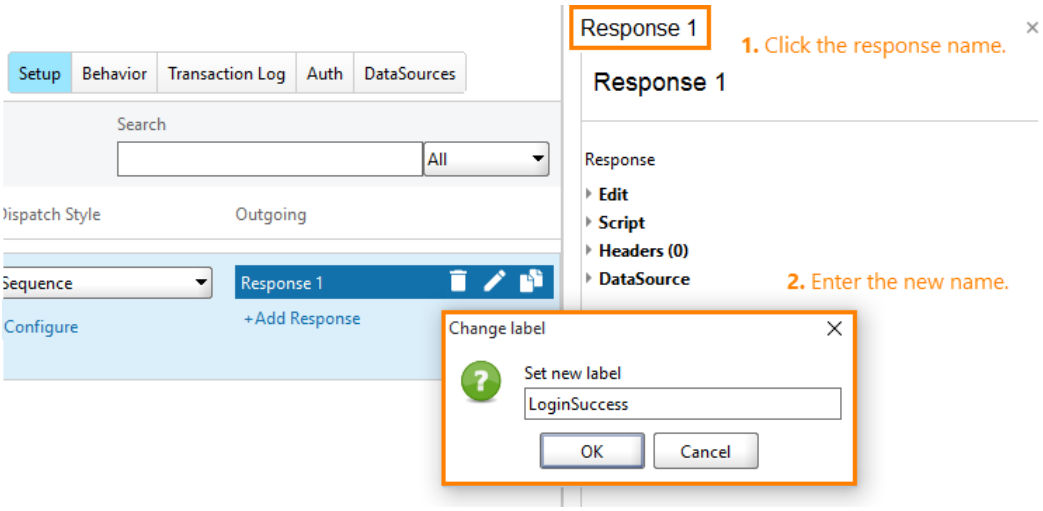
レスポンス設定

新規の仮想 API ダイアログでオペレーションを作成した際に、ReadyAPI では 1 つのレスポンスを作成されます。デフォルトではこのレスポンスは空なので、レスポンスを設定します。

1. エディター内のレスポンスをクリックします。レスポンス プロパティが右側に表示されます。



2. レスポンス名をクリックします。次のダイアログ ボックスで、**LoginSuccess** と入力し、**[OK]** をクリックします。



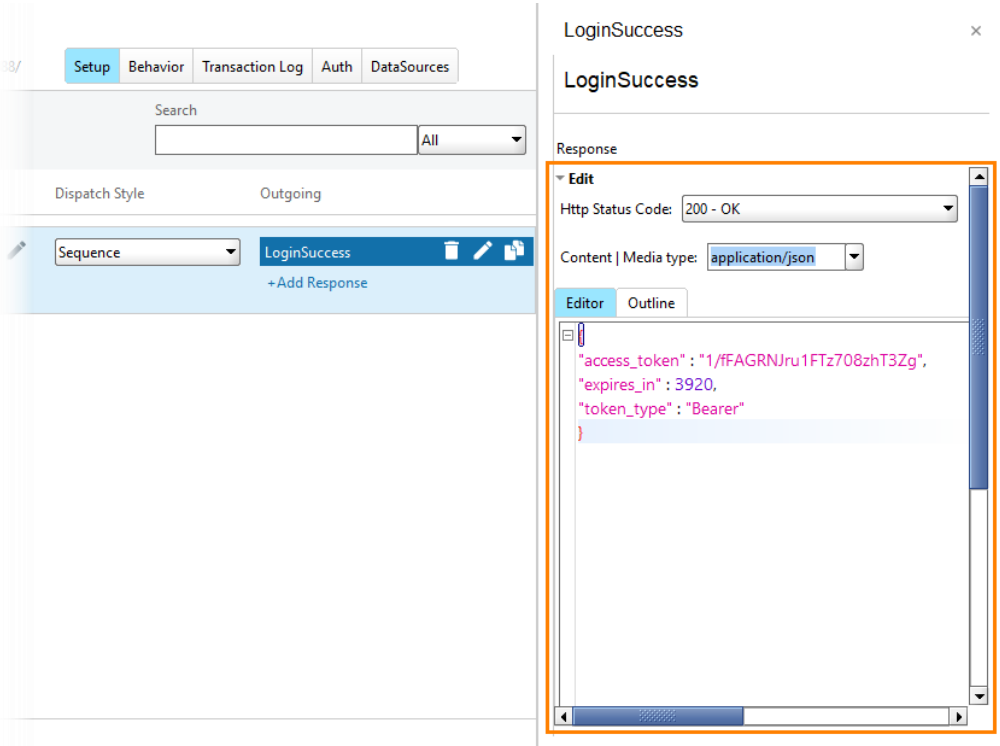
3. **Edit** セクションで、下記のプロパティ値を設定します。

プロパティ	値
Http status code	200 – OK

プロパティ	値
Content | Media type	Application/json

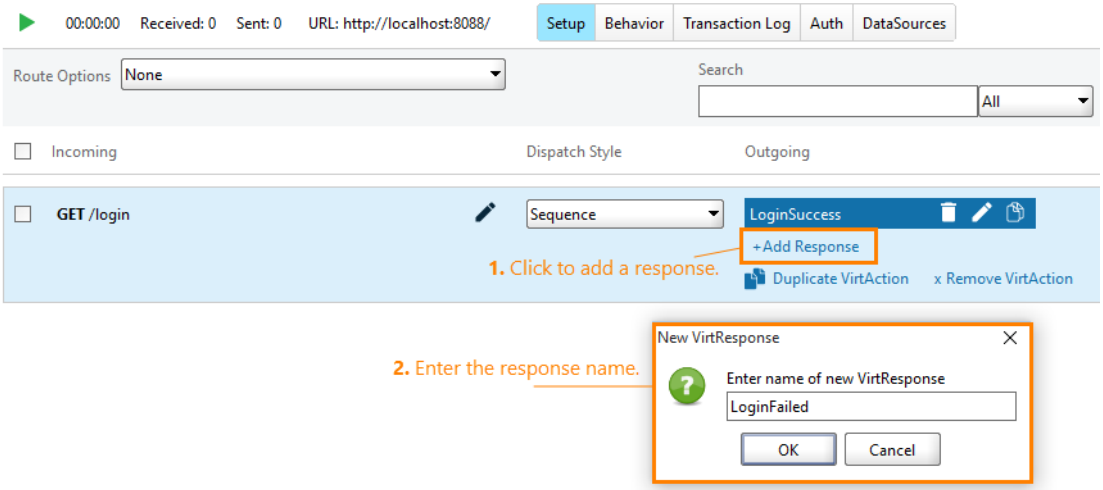
Editor	レスポンス本文の内容となる、下記のテキストを入力します。
	{
	"access_token" : "1/fFAGRNJru1FTz708zhT3Zg",
	"expires_in" : 3920,
	"token_type" : "Bearer"
	}

4. 他のプロパティはデフォルト値のままにします。

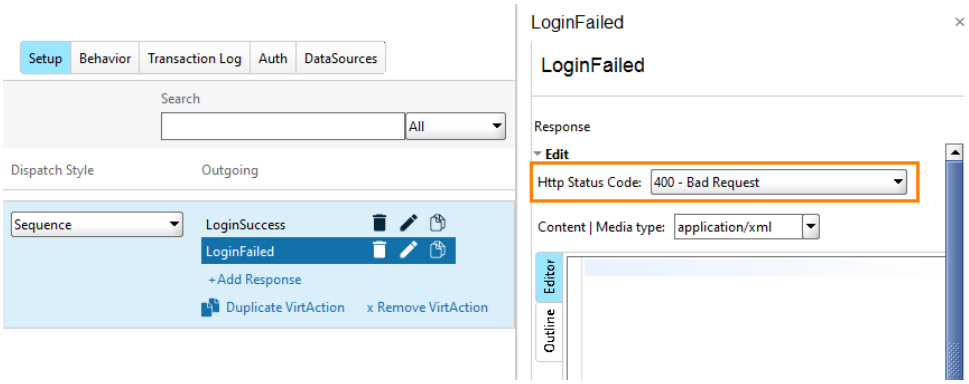


レスポンスの追加

1. **[Add Response]** をクリックし、もう 1 つのレスポンスを作成します。**LoginFailed** という名前を付けます。



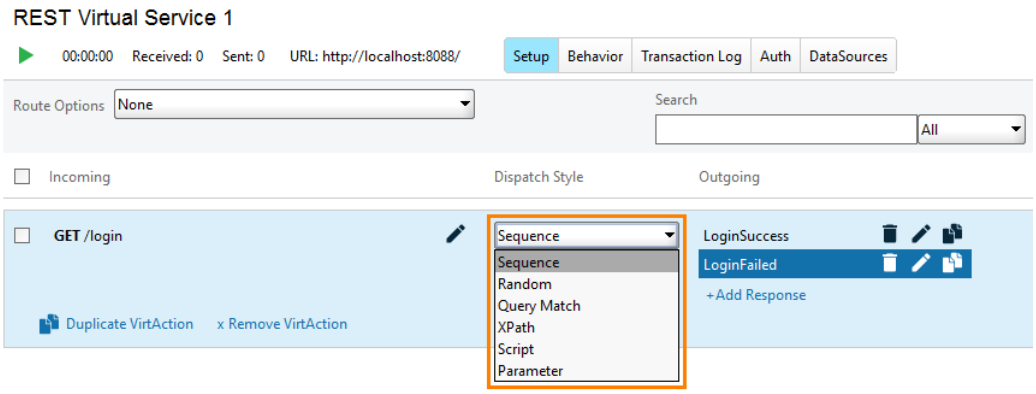
2. **[Http Status Code]** を **[400 – Bad Request]** に設定します。その他のフィールドはデフォルト値のままにします。



これでサービス レスポンスは設定されましたが、レスポンスが送信される条件を定義する必要があり、これをするには、**Dispatch** 設定を定義する必要があります。次のステップでは、**Dispatch** 設定を定義します。

3. レスポンス ディスパッチの設定

Dispatch Style の設定は、着信リクエストに対して仮想サービスが返すレスポンスを指定します。 ReadyAPI は、すべての仮想オペレーションに対して多数のディスパッチストラテジーを提供します。 仮想サービスエディターでディスパッチスタイルを選択できます。



デフォルトでは、新しいオペレーションには Sequence ディスパッチスタイルがあります。これは、オペレーションが一連のレスポンスを返すことを意味します。 Query Match および XPath スタイルを使用すると、リクエストの内容に応じてレスポンスを選択できます。パラメータースタイルは、着信リクエストのパラメーター値に応じてレスポンスをカスタマイズするのに役立ちます。 使用可能なディスパッチストラテジーとそのプロパティの詳細については、「ディスパッチ レスポンス」を参照してください。

次のステップ

パラメーター ディスパッチ ストラテジーを使用し、下記の仮想サービス動作を実行します。

- 着信要求が下記のパラメーターを持つ場合は—

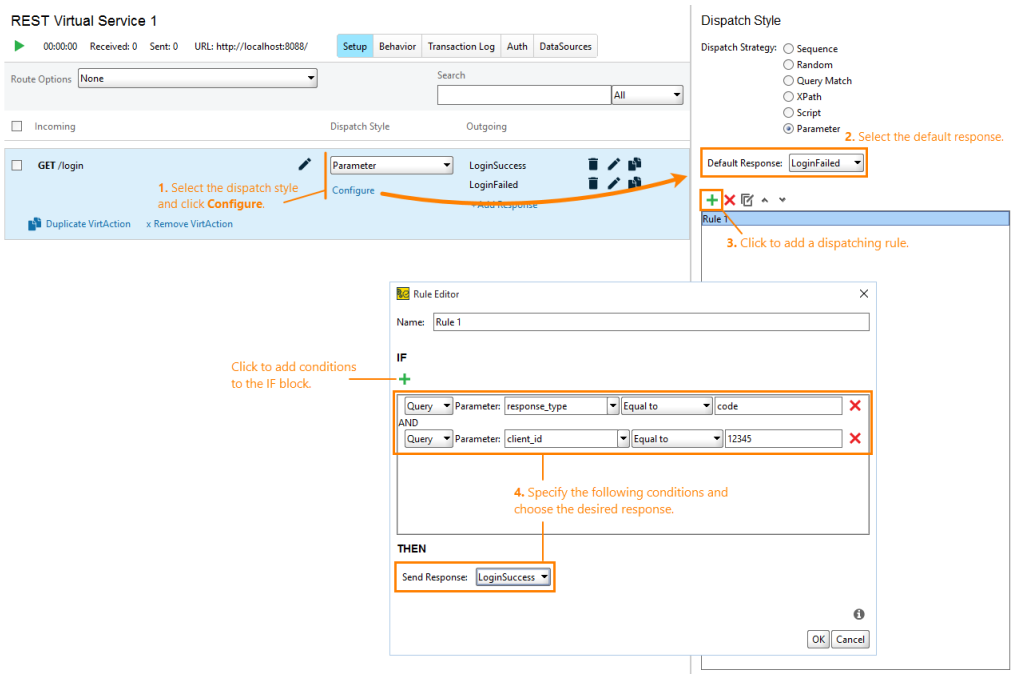
パラメーター	値
response_type	code
client_id	12345

— 仮想サービスは **LoginSuccess** レスポンスを送信します。

- それ以外の場合、サービスは **LoginFailed** レスポンスを送信します。

ディスパッチ プロパティの設定

1. Get /login 仮想オペレーションの場合は、**[Dispatch Style]** ドロップダウン リストから **[Parameter]** を選択し、**[Configure]** をクリックします。
2. **[Default Response]** を **[LoginFailed]** にします。
3. **[+]** をクリックし、新規ルールを追加します。**[Rule Editor]** が表示されます。
4. 下記の値を設定します。



5. **[OK]** を選択し、変更を保存します。

これで、ディスパッチは設定されました。仮想サービスは実行およびテストが可能な状態となりました。

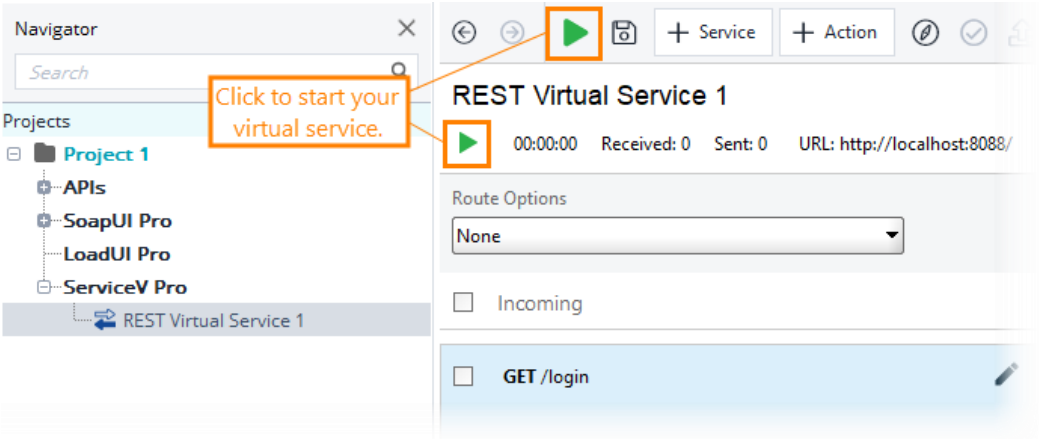
4. 仮想サービスの実行およびリクエストの送信

仮想サービスの設定が完了すると、プロジェクトを実行およびテストできます。

仮想サービスの開始

コンピューターまたはネットワーク上の別のマシンで仮想サービスを実行します。異なるマシン上で実行する場合は、VirtServer をマシン上にインストールします。詳細については、「Running Virtual Services」を参照してください。

このチュートリアルでは、このコンピューター上で仮想サービスを実行します。 仮想サービスエディターで **[Run]** をクリックします。

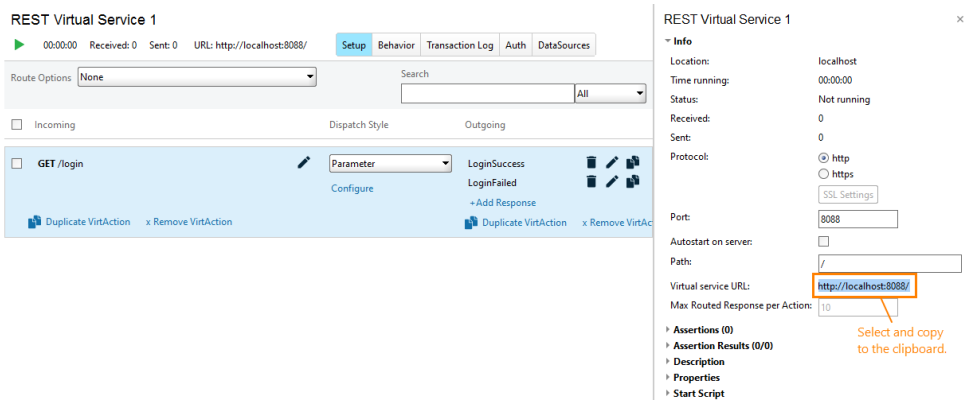


注意:同じポート上で複数の Web サーバーまたはサービスを実行することは禁じられています。 仮想サービスが動作しているポート上に他の Web サービス、Web サーバーがすでに存在するばあい、実行は失敗します。 サービスポートを変更し、サービスを再起動してください。また、ReadyAPI では、リクエスト ルーティングを使用する仮想サービスを一度に 1 つしか実行できません。Virtserver がインストールされたリモート コンピューターでは、リクエスト ルーティングを使用する仮想サービスを複数実行できます。

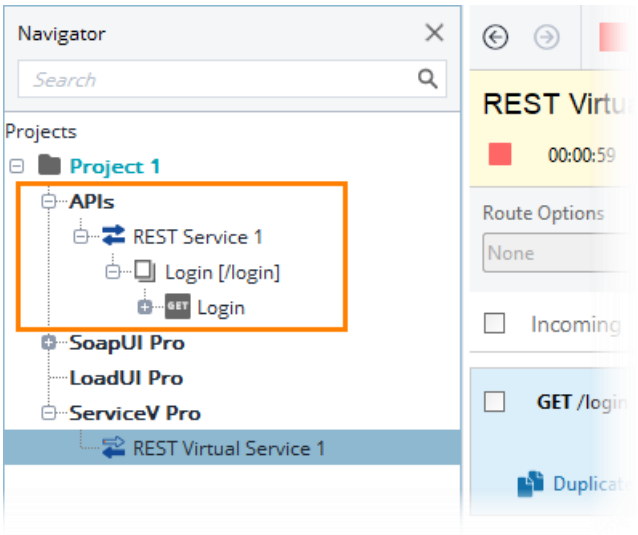
テスト リクエストの作成

仮想サービスの起動後、リクエストを送信します。

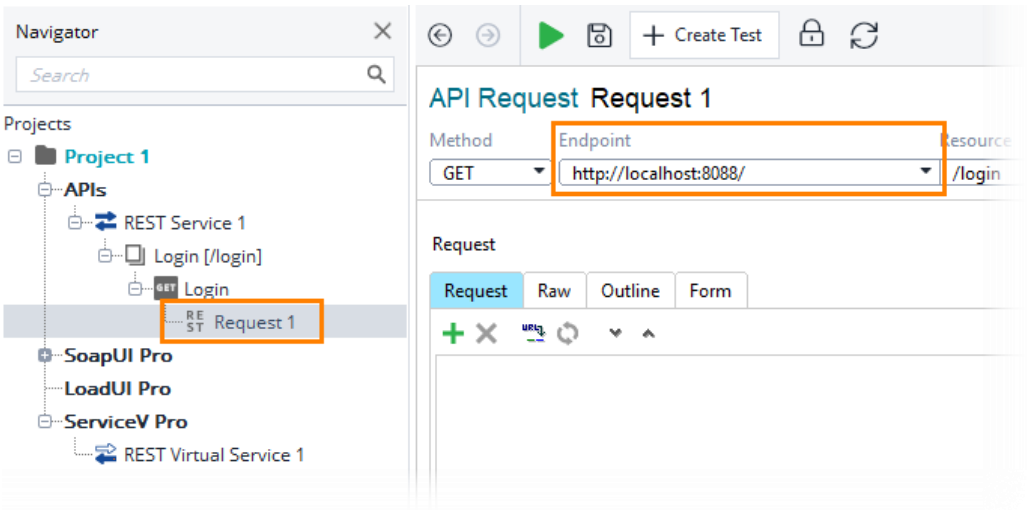
1. **Navigator** パネル内の仮想サービスを選択します。 仮想サービスプロパティを確認し、**[Virtual service URL]** フィールドの値をコピーします。



2. [Navigator]パネルで[API]ノードを展開します。 仮想サービスにオペレーションを追加すると、ReadyAPI は仮想サービスの API 仕様を自動的に生成し、その仕様にオペレーションを追加しました。



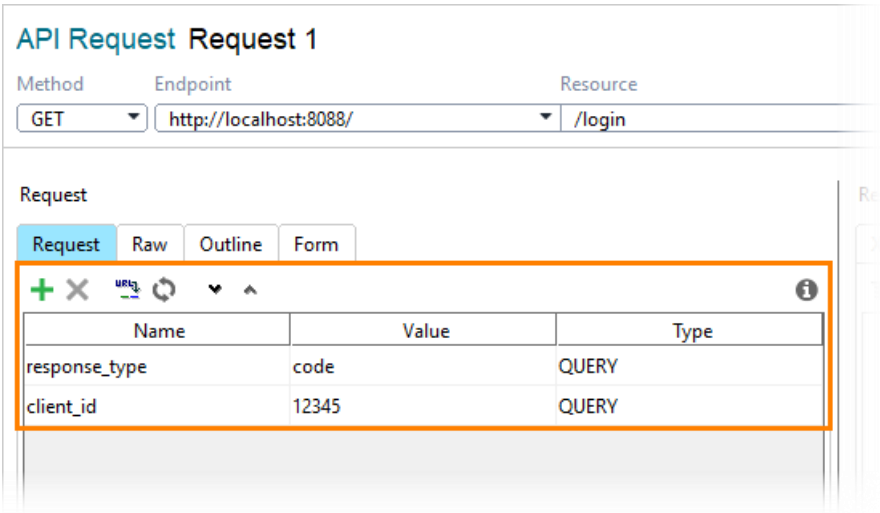
3. リクエスト エディターを開き、仮想サービスが使用するエンドポイントを指定します（下記の画像では、`http://localhost:8088/`）。



4. リクエスト パラメーターを指定します。[+] をクリックし、表にデータを入力します。以下の 2 つのパラメーターを追加します。

名前	値	スタイル
response_type	code	QUERY
client_id	12345	QUERY

エディターには、以下のように表示されます。

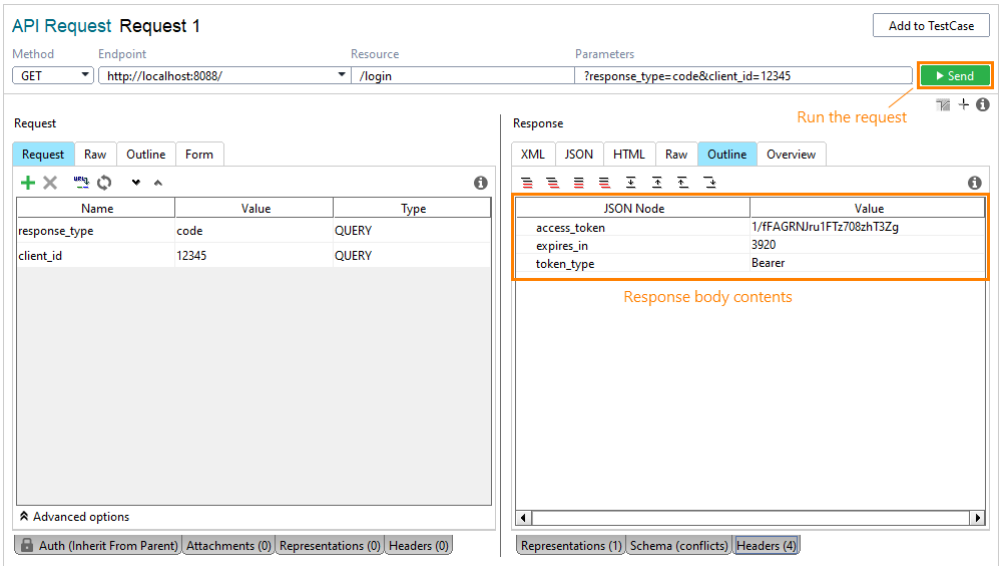


テスト リクエストの実行

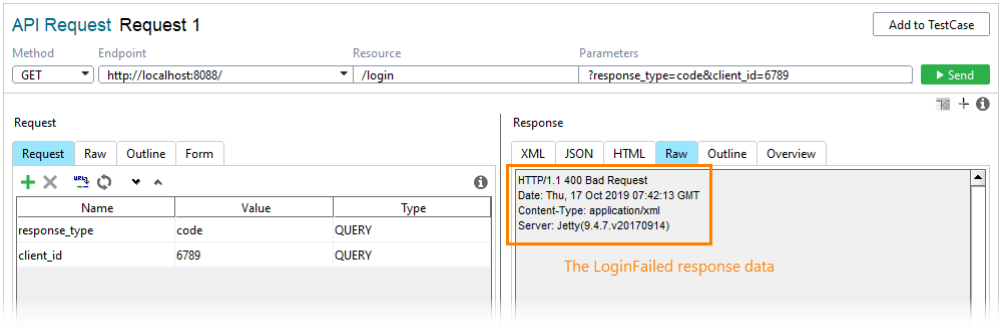
1. リクエストをシミュレートするには、リクエスト エディターで 

をクリックします。

   サービスがリクエストを処理し、LoginSuccess レスポンス内容がエディターに表示されます。

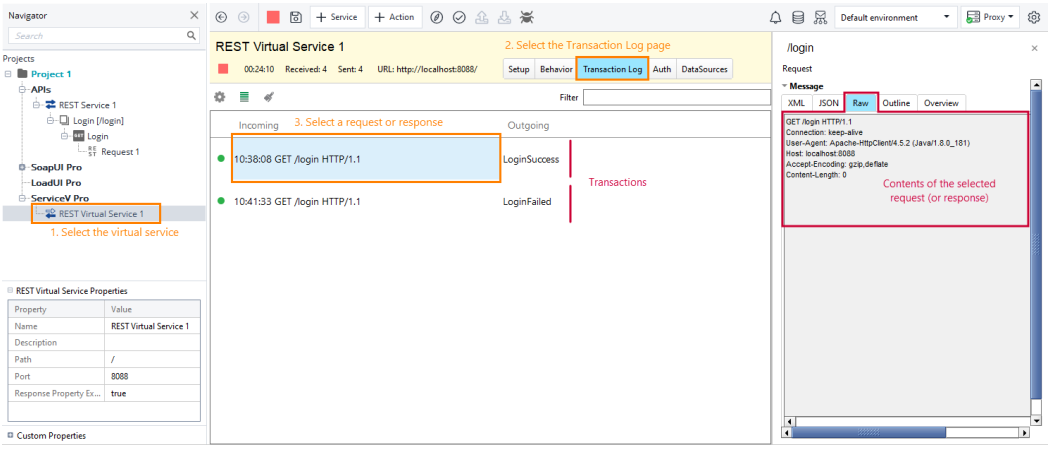


2. *client_id* パラメーターを異なる値に変更し、リクエストをもう一度実行します。仮想サービス ディスパッチャーによって **[LoginFailed]** レスポンスが選択されます。



リクエスト エディター内では、リクエスト パラメーターとレスポンス内容を見て、仮想サービスが正常に動作しているかを確認します。さらに、ServiceV の **Transaction Log** ページで、リクエストとエディターを確認することもできます。

- **[ServiceV]** に移動して **Navigator** パネルの仮想サービスを選択し、仮想サービスエディターの **Transaction Log** ページに切り替えます。



- ページ上のそれぞれの列はトランザクション、すなわち リクエスト/レスポンス ペアに対応しています。

  特定のリクエストまたはレスポンスを選択すると、それらの内容を確認することができます。右側に表示されます（上記画像をご確認ください）。

リクエストまたは結果確認後、サービスエディター上の **[■ Stop]** をクリックし 仮想サービスを終了します。

次のステップ

これで、チュートリアルは終了です。この段階で、仮想 API は完全に設定されました。新規にオペレーションを追加し、機能テスト、データ ドリブン テストまたは負荷テストを作成できます。以下のトピックが準備されています。

仮想サービスの設定

仮想サービスの作成や設定に関する情報が含まれています。

VirtServer のチュートリアル

リモートコンピューター上で virt を実行するための VirtServer の使用方法について説明します。

機能テストの作成

SoapUI での API 機能テストの作成方法について説明します。

第 5 章: VirtServer のチュートリアル

このセクションのトピックでは、VirtServer の使い方、仮想 API サービス (仮想サービス) の展開、および VirtServer での実行方法などについて説明します。

1. VirtServer のインストールと実行

VirtServer を使用するには、VirtServer をインストールして、少なくとも 1 人のユーザー登録をし、ライセンスをアクティベートします。

1. VirtServer のインストール

VirtServer をインストールするには、インストーラーを実行して画面の指示に従います。「VirtServer Installation」を参照してください。

2. VirtServer の実行

VirtServer のプロセスを開始するには、次のファイルを実行します。

- Windows: `<VirtServer directory>/bin/virtserver.bat`
- Linux または macOS: `<VirtServer directory>/bin/virtserver.sh`

> ルート権限で VirtServer をインストールした場合、ルート権限で実行するか、コマンドラインで VirtServer settings ファイル(*virt-server.yml*) の場所を手動で指定します。

3. ユーザーの作成

ServiceV から VirtServer を操作するには、ユーザー名とパスワードが必要です。ユーザーの作成は初回のみで、VirtServer を起動するたびに作成する必要はありません。

バージョン 1.6.0 以降の場合

- VirtServer の**インストール後初めて**起動する際に、ユーザー名とパスワードを指定する必要があります。画面の指示に従ってください。
- ユーザー登録をすると、さらにユーザーを追加できます。VirtServer を実行してコマンド ラインの `–a` 引数を指定します:
  - Windows:
    ```
    <VirtServer directory>/bin/virtserver.bat -a userName:password
    ```


- Linux または macOS:
  `<VirtServer directory>/bin/virtserver.sh -a userName:password`

ヒント: パスワードを指定しない場合、VirtServer はユーザーを作成する前にパスワードを要求します。

VirtServer を操作するユーザーごとのユーザー設定をすることを推奨します。

バージョン 1.5.0 以前の場合

VirtServer 1.5.0 以前のバージョンでは、デフォルトでユーザーが作成されます:

User name: `defaultUser`

Password: `svp4ever`

パスワードは、VirtServer settings file (*virt-server.yml*) ファイルで設定できます。

4. VirtServer ライセンスのアクティベーション

VirtServer を初めて実行する場合、ライセンス ファイルを指定してライセンスをアクティベートする必要があります。画面の指示に従ってください。「VirtServer License Activation」も参照してください。

2. 仮想サービスの展開

1. 仮想サービスの作成

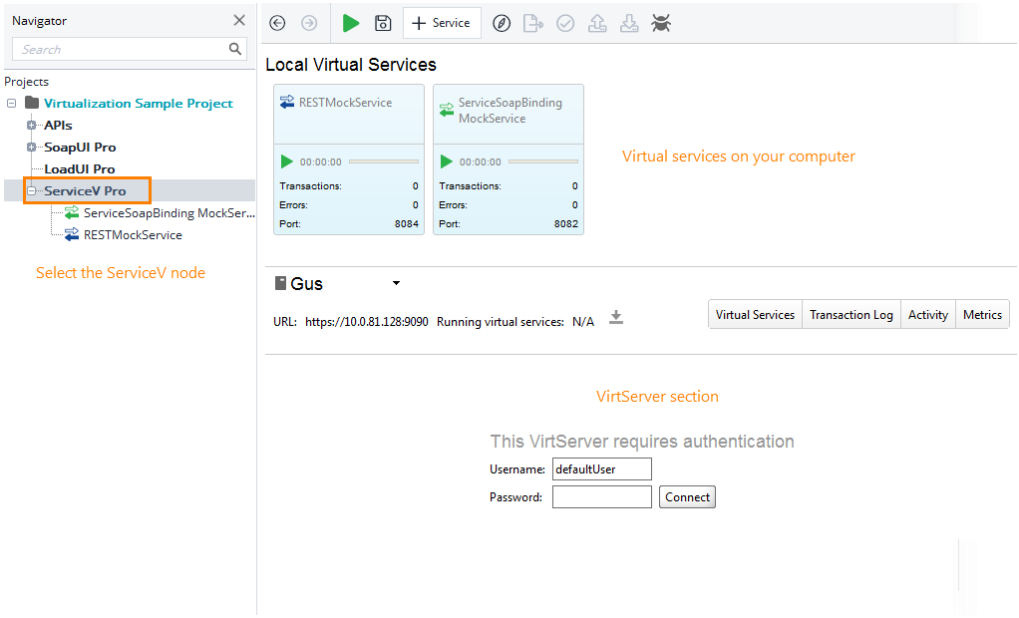
仮想サービスを VirtServer で実行する前に、仮想サービスを作成する必要があります。仮想サービスは、ReadyAPI の ServiceV で作成します。仮想サービスの作成方法については、「第 4 章 ServiceV スタートガイド」を参照してください。

このチュートリアルでは、Sample-Virtualization-project の仮想サービスを使用します。このプロジェクトは製品と一緒にインストールされており、`<ReadyAPI>¥tutorials` というディレクトリにあります。ReadyAPI でこのプロジェクトを開き、**[ServiceV]** に切り替えます。**[ServiceV]** パネルの **[Local Virtual services]** セクションに 2 つの仮想サービスが表示されます (次に示す画像を参照してください)。

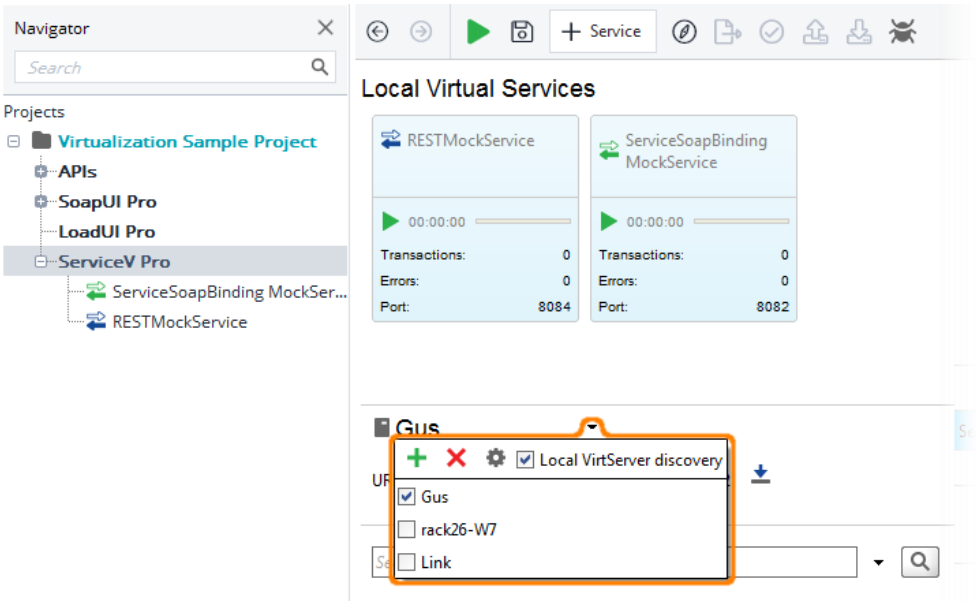
2. 仮想サービスの展開

仮想サービスを作成すると（またはファイルから開くと）、VirtServer で展開 (アップロード) できます。

1. ReadyAPI を開いて **[ServiceV]** に切り替え、左のプロジェクト ツリーでプロジェクト ノードを選択します。
   製品ウィンドウの中央部に VirtServer パネルが表示されます。



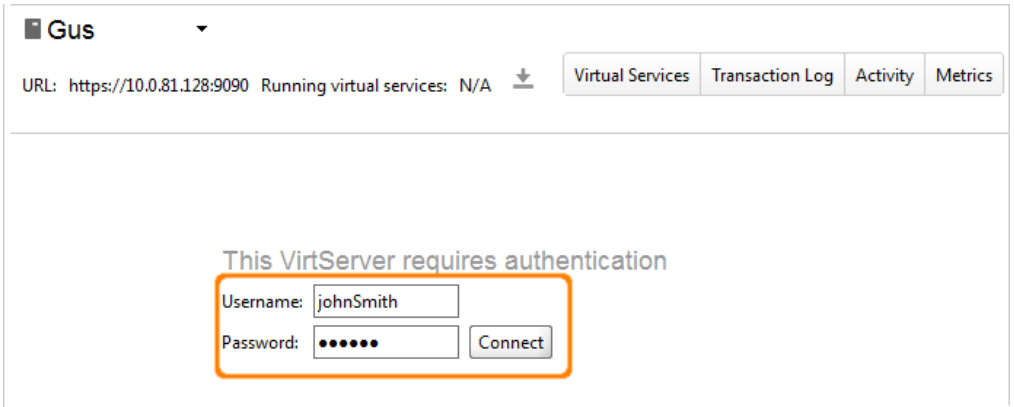
2. VirtServer を選択して接続します。ドロップ ダウン矢印をクリックして使用する VirtServer を選択します。



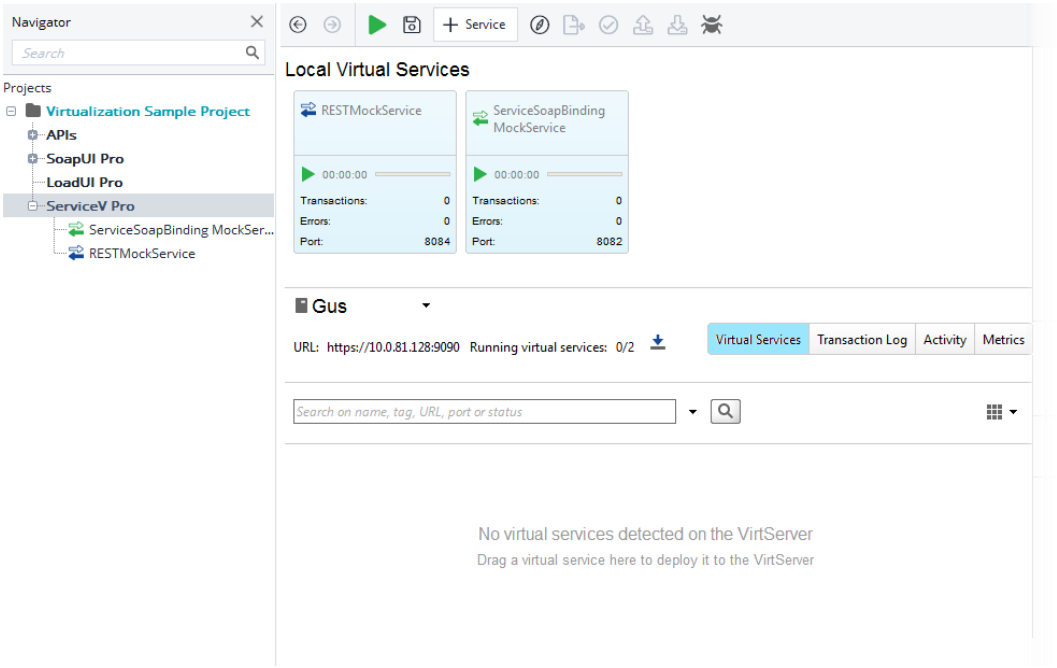
[Local VirtServer discovery] チェック ボックスが選択されていると、ServiceV がローカル ネットワークで使用可能な VirtServers を自動的に検索します。

このチェック ボックスが選択されていない場合、もしくはリストに VirtServer が存在しない場合、**[+]** ボタンをクリックして次のダイアログで VirtServer の動作プロトコル、IP アドレス、およびポートを指定します。

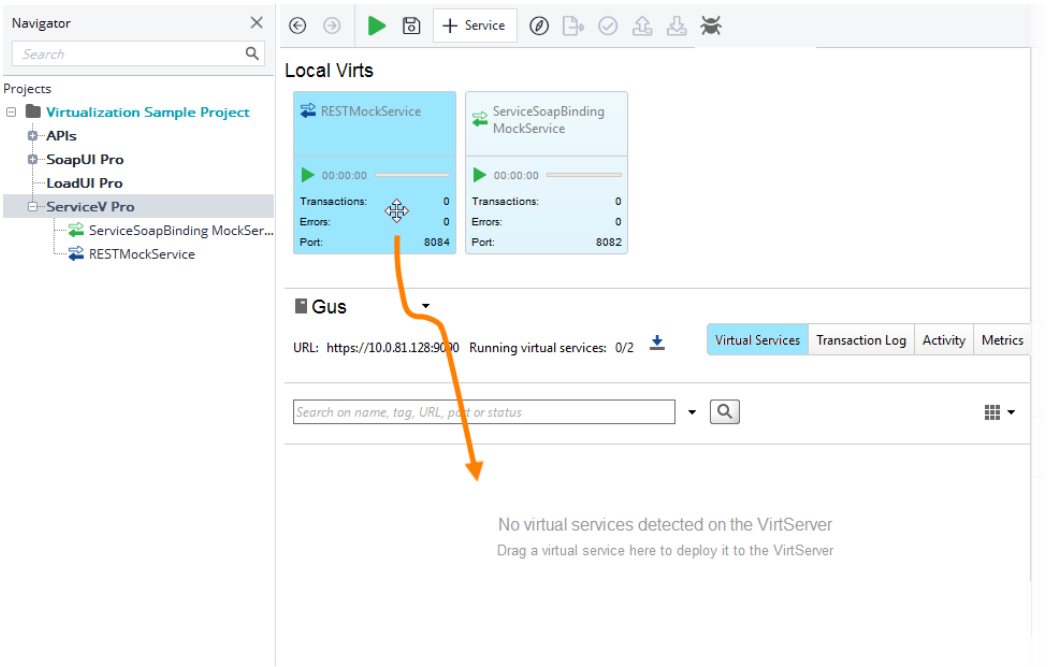
3. VirtServer に接続します。このチュートリアルの Step 1 で作成したユーザー名とパスワードを指定し、**[Connect]** をクリックします。



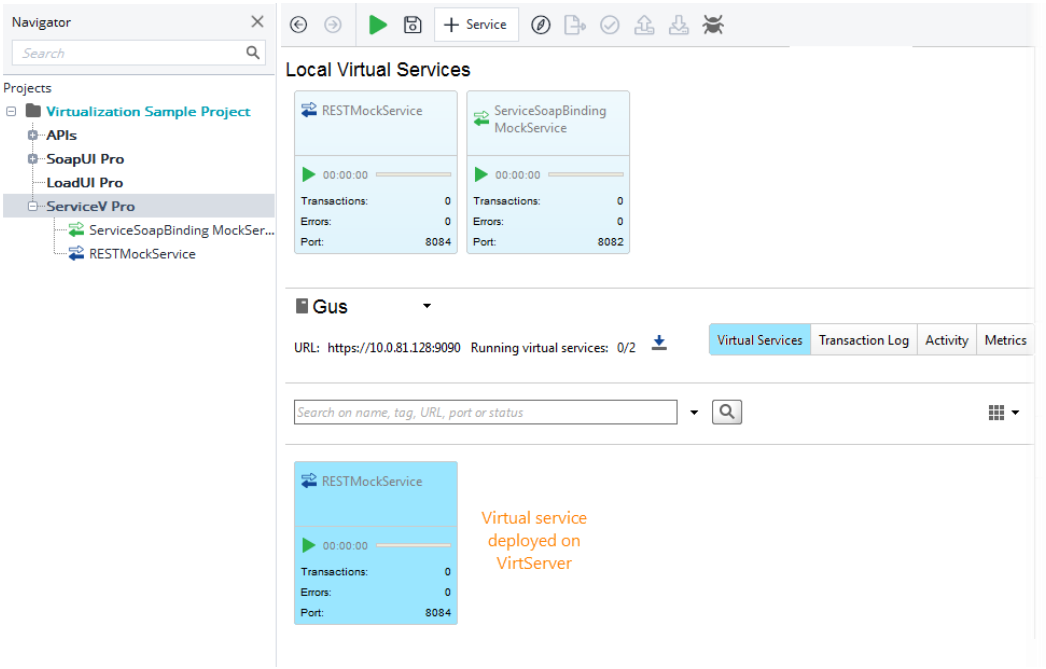
選択した VirtServer で仮想サービスが実行されます。この例では、VirtServer には仮想サービスがまだ存在しません。



4. エディターの上部にある **[Local Virtual Services]** セクションから **[VirtServer]** セクションに仮想サービスをドラッグします。



VirtServer セクションで仮想サービスが表示されます。



仮想サービスの展開中に NullPointerException エラーが発生した場合には、ReadyAPI および VirtServer のバージョンをアップデートしてください。この問題は 1.5.0 以前のバージョンで発生する可能性がありますが、1.6.0 では修正されています。

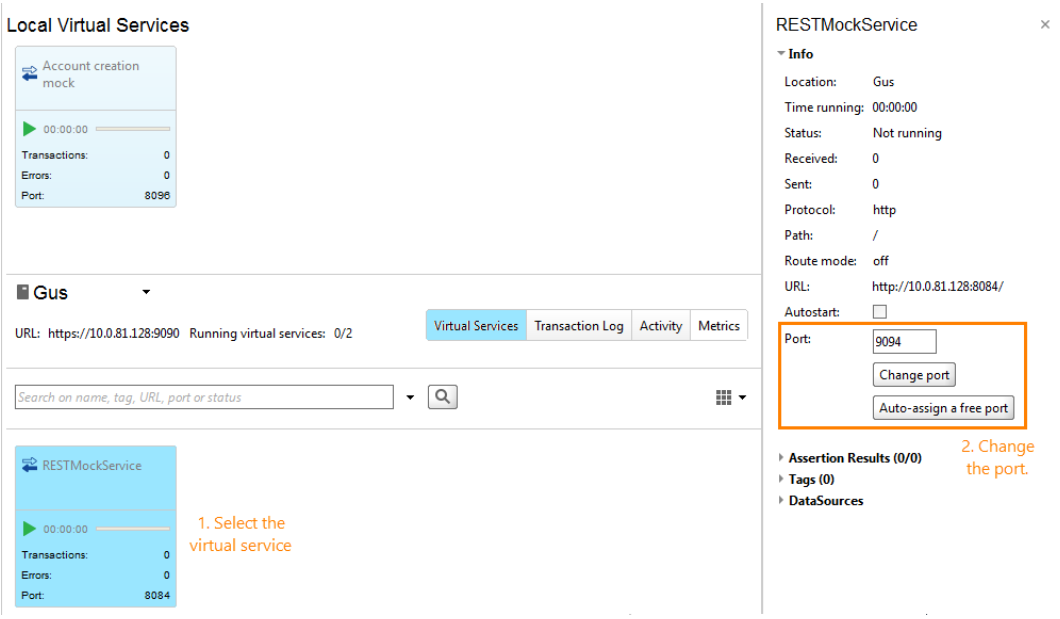
3. 仮想サービスのポート番号の変更

仮想サービスはポートを使用します。ポートは、エディターの右側にある仮想サービスのプロパティで指定できます(次の画像を参照)。

VirtServer で仮想サービスを展開したのち、ポート番号の変更が必要な場合があります。仮想サービスに使用するポートが、別の仮想サービスまたはプロセスにより VirtServer で使用されている場合などに変更が必要となります。

VirtServer で仮想サービスのポートを変更するには、次の手順に従ってください。

1. エディターの VirtServer セクションで仮想サービスを選択します。
2. 誤って仮想サービスを実行してしまった場合には、実行を停止します。
3. エディターの右側にある サービス プロパティのポート番号を変更し、**[Change port]** をクリックします。もしくは、**[Auto-assign a free port]** をクリックして VirtServer が仮想サービスにポートを自動的に選択するようにします。



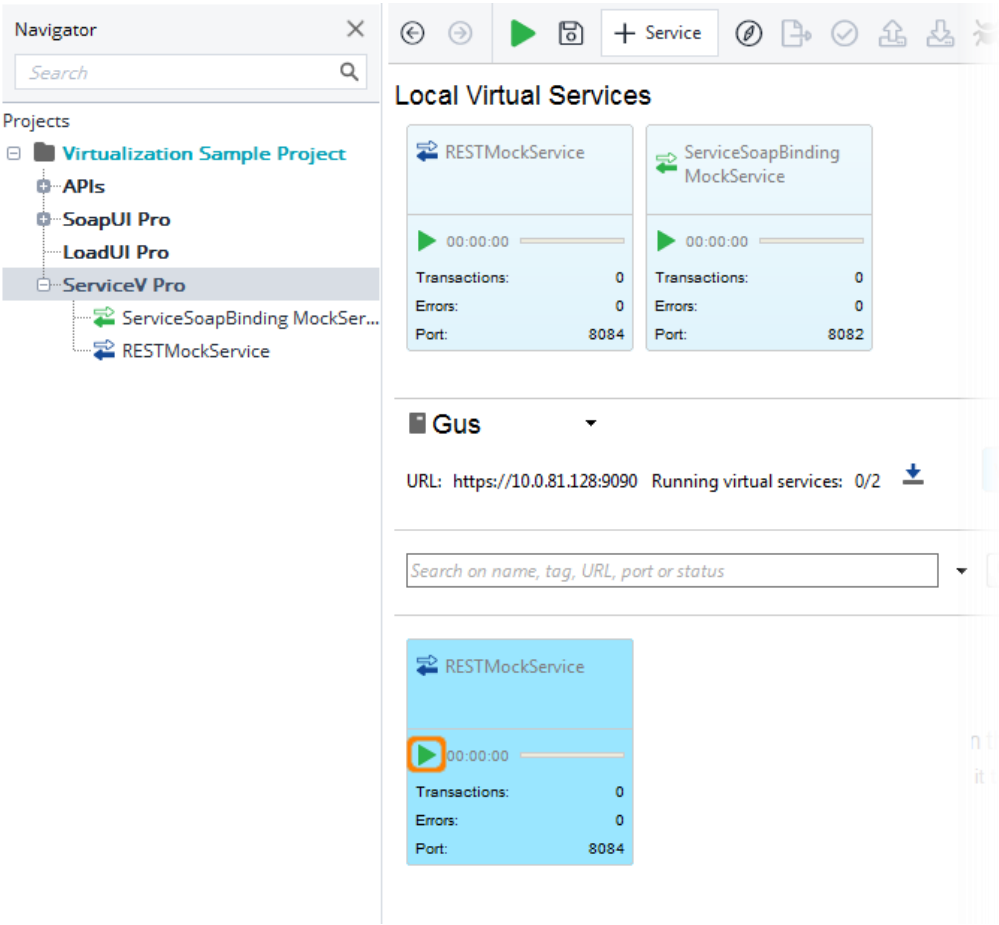
これで、仮想サービスをスタートしてテストを実行する準備が整いました。

3. 仮想サービスの実行とテスト

1. 仮想サービスの実行

VirtServer で仮想サービスを展開して設定を終えると、仮想サービスを実行できます。

仮想サービスを実行するには、[ ] ボタンをクリックします。



VirtServer は、指定したポート番号で仮想サービスを実行します。

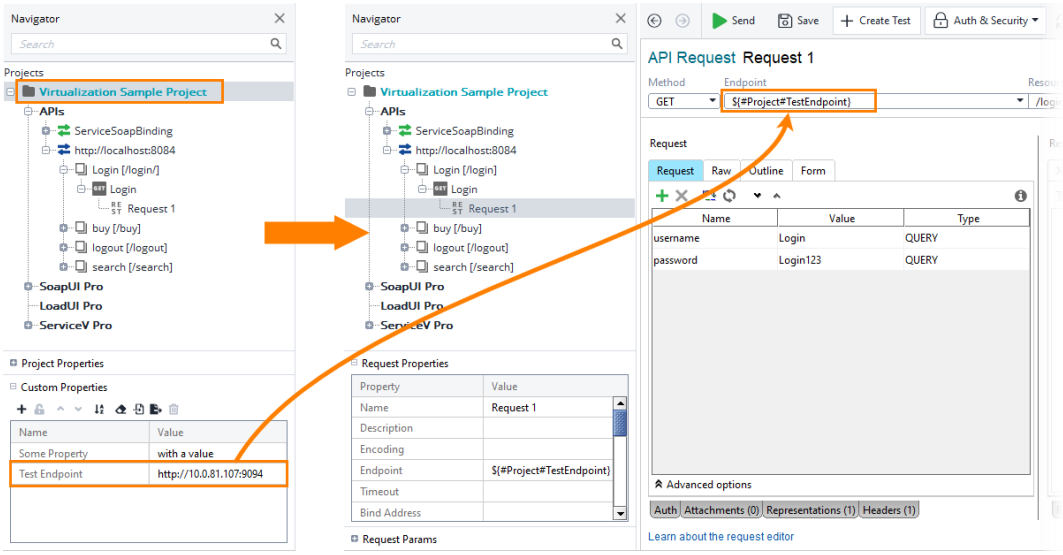
ヒント: 「VirtServer Web Interface」を使用することにより、仮想サービスをブラウザーから始めることもできます。

2. 仮想サービスのテスト

仮想サービスの実行後、ReadyAPI のテスト プロジェクトからテスト リクエストを送信できます。他の API で行うのと同じやりかたで行います。

ヒント:

仮想サービスのテストを作成する際に、VirtServer が使用するポートが不明の場合があります。テストにポート番号をハードコード化した場合、仮想サービスの展開後にポート番号を変更する必要があります。回避策として、仮想サービスのアドレスおよびポート値を保存するプロジェクト レベルもしくはテスト スイート レベルのプロパティを作成し、この変数をテスト リクエストの Endpoint プロパティとして指定することができます。ポート番号が変更した場合、すべてのリクエストの Endpoint プロパティをアップデートするのではなく、プロジェクト レベル (もしくはテスト スイート レベル) のプロパティを変更する必要があります。



次のステップ

VirtServer のチュートリアルはこれで終了です。下のトピックでは、よく使うタスクの実行方法を説明しています:

VirtServer の実行

Different Ways to Run VirtServer

Running VirtServer as Service

仮想サービスの設定

Updating Virt Ports On VirtServer

Editing Deployed Virts

Using Virts With DataSources on VirtServer

VirtServer の設定

Managing VirtServer Users

VirtServer Settings

VirtServer Command-Line Arguments

サポート

お問合せ先

エクセルソフト株式会社

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

SmartBear

https://support.smartbear.com/product-list/