

GeneXus™
by Globant

UIテスト

GeneXus[™]
© Genesys

UIテストの有用性

GeneXus™
by Globant



GeneXusの開発者やテスターは、アプリケーションのデプロイ後にインターフェーステストを行う必要があり、これらのリグレッションテストを多くの環境（QA、プリプロダクション、その他）で実行しなければならない場合もあります。

UIテストは、各デプロイメント後に即座にフィードバックを提供することを可能にし、バグを見つけて迅速に修正することを可能にします。

UIテストは、EndToEndのテストを作成することができます。

例えば、アプリケーションがAmazonの場合、EndToEndのテストは、アプリケーションへのログイン、商品の検索、ショッピングカートへの追加、配送先の追加、購入の支払い、ログアウトを行うことが可能です。

これらのテストは、異なるアプリケーションモジュール間の統合を検証します。そのため、この種のテストは統合テストやユーザー受入テストに価値があるものとなります。

これらのテストは、各デプロイメントの後だけでなく、GeneXusのアップグレード移行、データベースの新バージョン、およびサーバーの変更後にも実行されます。

GeneXusでのUIテスト





Web Panels



Web component

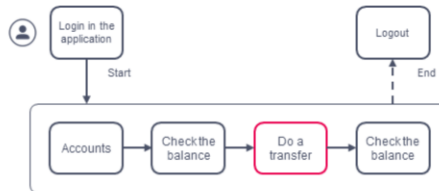


Transaction

GeneXusでは、UIを持つオブジェクトは、Web UIテストでテストされます。
これらのオブジェクトは、アプリケーションがブラウザで実行されている間にテストされる
ので、テストを実行するためにはアプリケーションをデプロイする必要があります。

UIテストのアプローチは、実行中のアプリケーション上でのユーザーインタラクションのシ
ミュレーションを可能にし、
自動的なインターフェイスの検証を実装し、リグレッションテストとして含めることが目標
となります。

e-banking E2E flow



アプリケーションの最も重要なEndToEndのフロー、つまりビジネスにとってリスクの高いフローを自動化することが目標です。

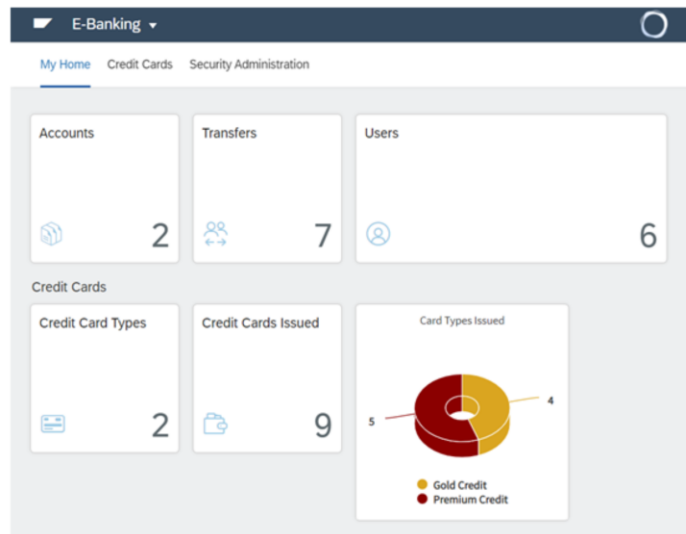
また、バグが頻繁に発生するフローや、顧客に新バージョンをリリースする際にうまく機能するはずのフローも自動化します。

今回の例であるEBankのアプリケーションでは、以下の重要なEndToEndの流れがあります。

- ・ アプリケーションにログインする
- ・ 口座残高を確認する
- ・ 振替をする
- ・ 最終残高を確認する
- ・ ログアウト

UIテストを作成

GeneXus™
by Globant



GX Server: <http://samples.genexusserver.com/v17> | KB Name: DemoEbankingGXtest

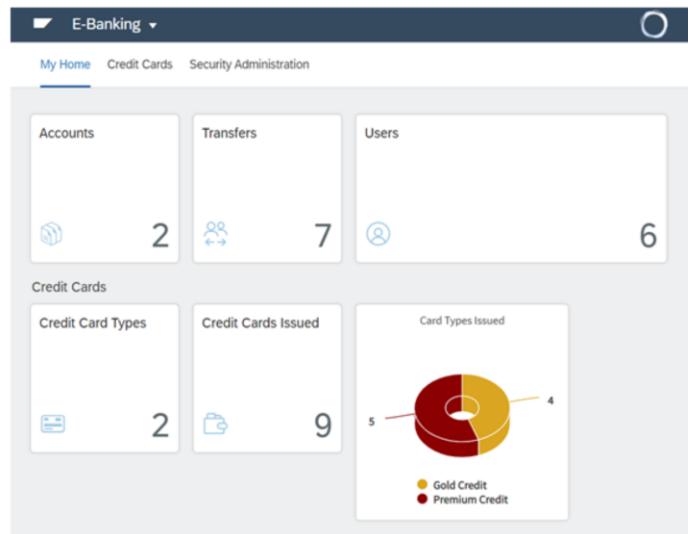
そこで、私たちは同じEBankのアプリケーションを作成いたしました。

前述の通り、ここでは口座、送金、クレジットカードが管理されています。

そして、リグレッションテストに含まれるべき、ビジネスの最も重要な機能を自動化する必要があります。

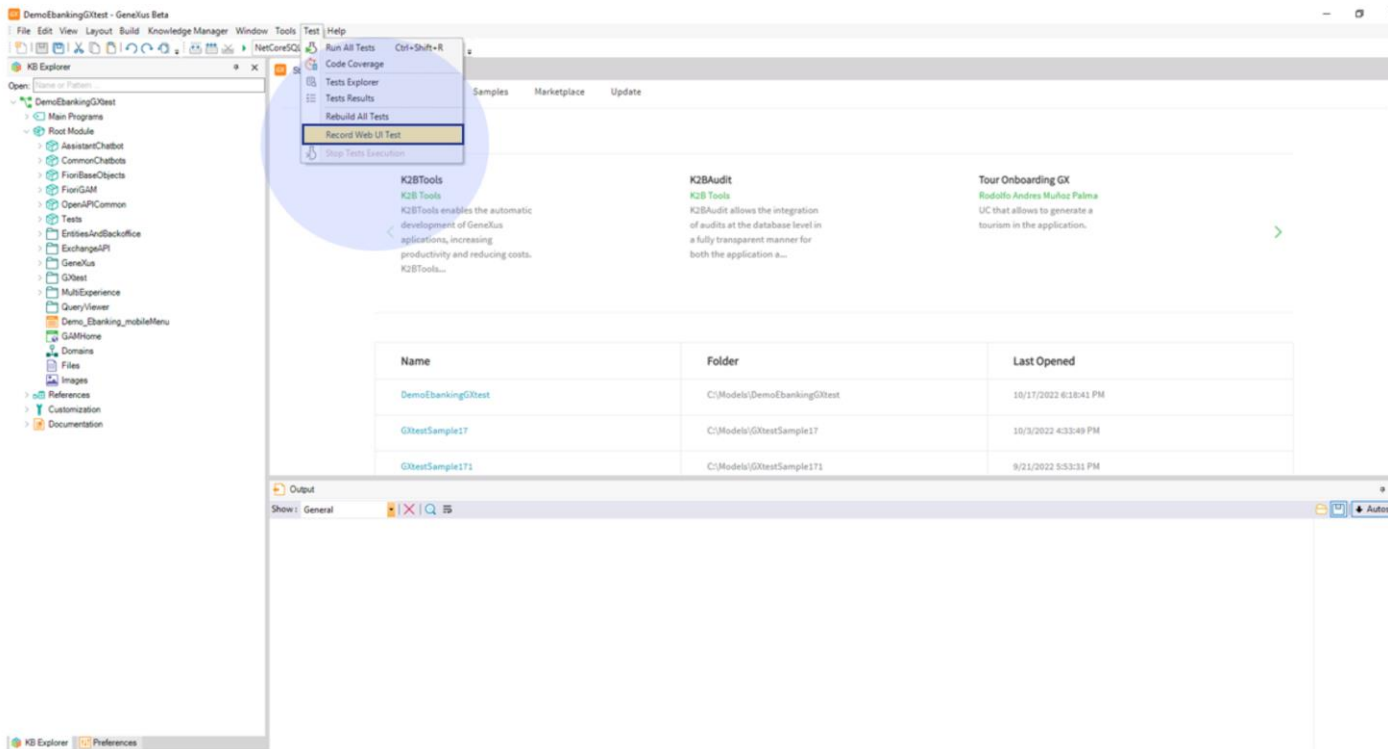
この場合、リスクの高いフローは、次のステップを持つEndToEndのテストです：

ログインし、新しい振替を実行し、関係する口座の残高をチェックします。

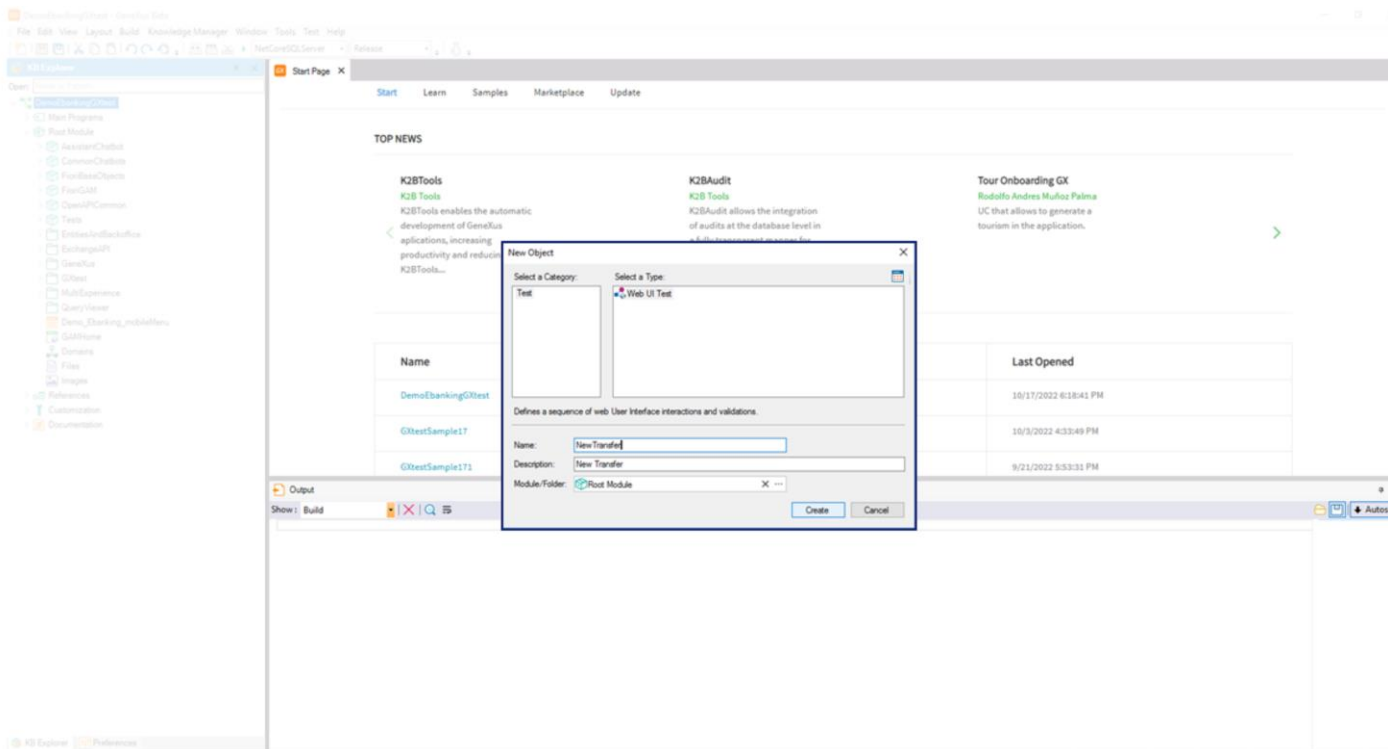


GX Server: <http://samples.genexusserver.com/v17> | KB Name: DemoEbankingGXtest

まず、新しい Web UI Test オブジェクトを作成します。
このオブジェクトによって、ブラウザを制御し、アクションを実行し、UI の検証を追加することができます。
さまざまなアクションを利用することができます。

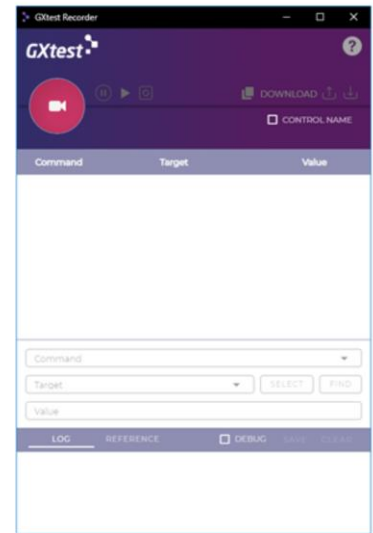


Web UI Testオブジェクトを作成するには、ツールバー → テスト → Web UIテストを記録をクリックします。



テスト名を入力し「作成」をクリックします。
この場合、オブジェクト名は「NewTransfer」としています。

この操作により新しいWeb UIテストが作成されます。

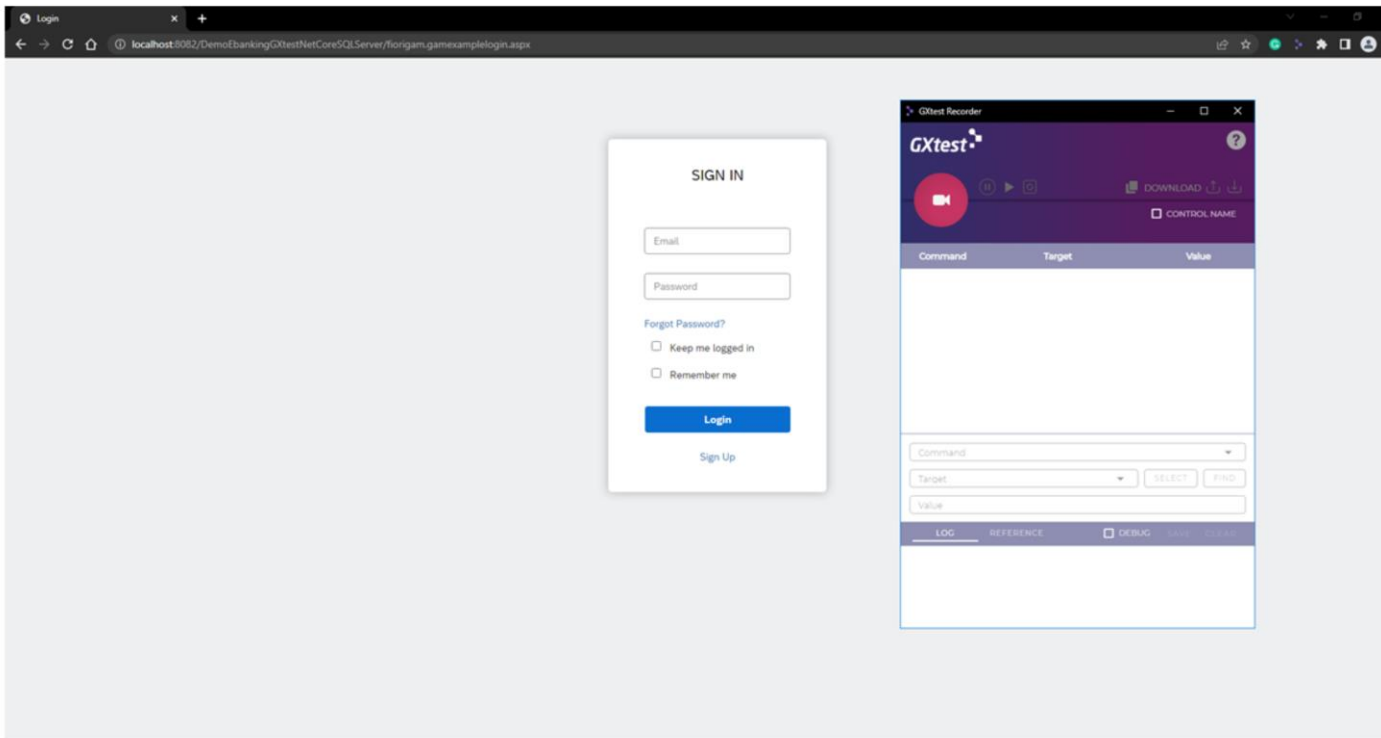


「作成」をクリックすると、自動的にChromeブラウザが開くので、そこに検証したいアプリケーションのホームページURLを入力します。

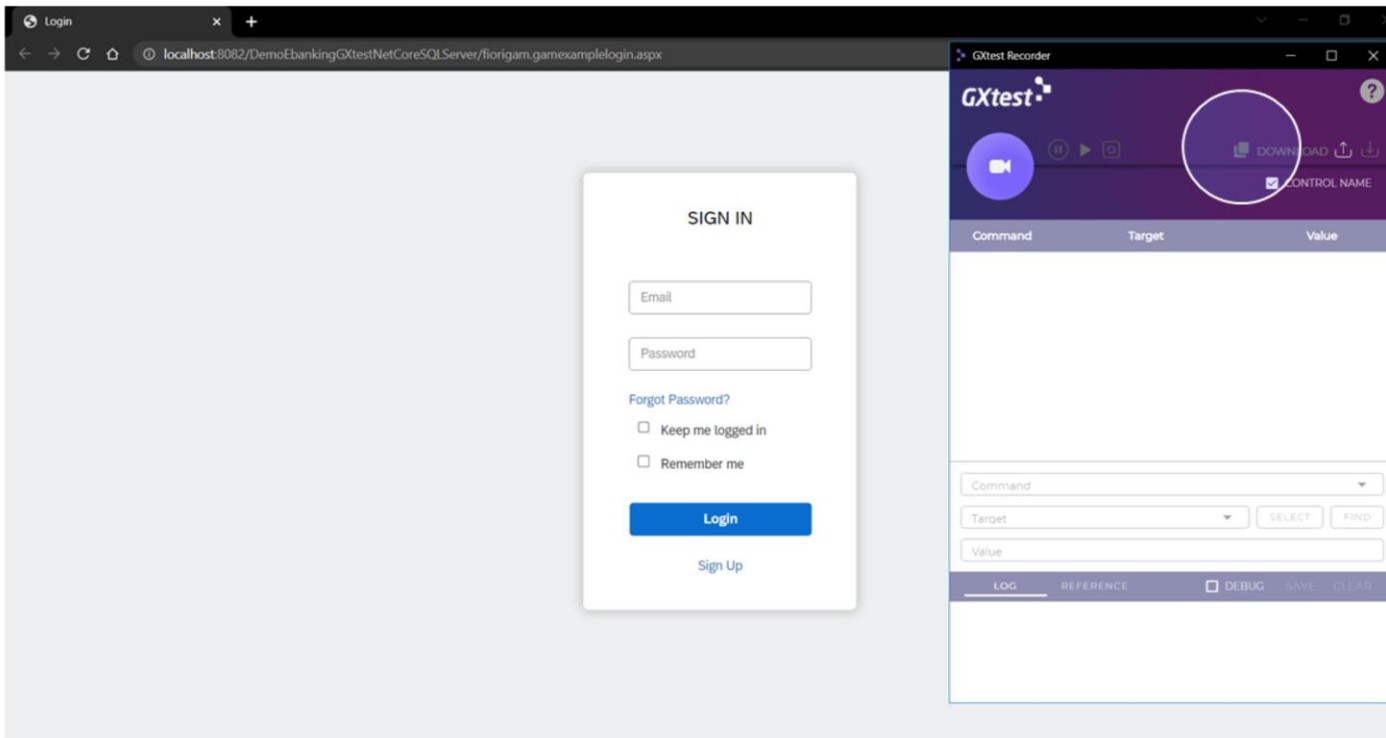
また、GXtest RecorderというChromeの拡張機能が自動的に開きます。

GXtest Recorderは、ユーザー（開発者またはテスター）がアプリケーション上で行ったアクションを記録するChrome拡張機能です。

ユーザーはアプリケーション上でワークフローを実行するだけで、GXtest RecorderがGeneXusコードを生成し、KB内のWeb UIテストに素早くインポートすることができます。

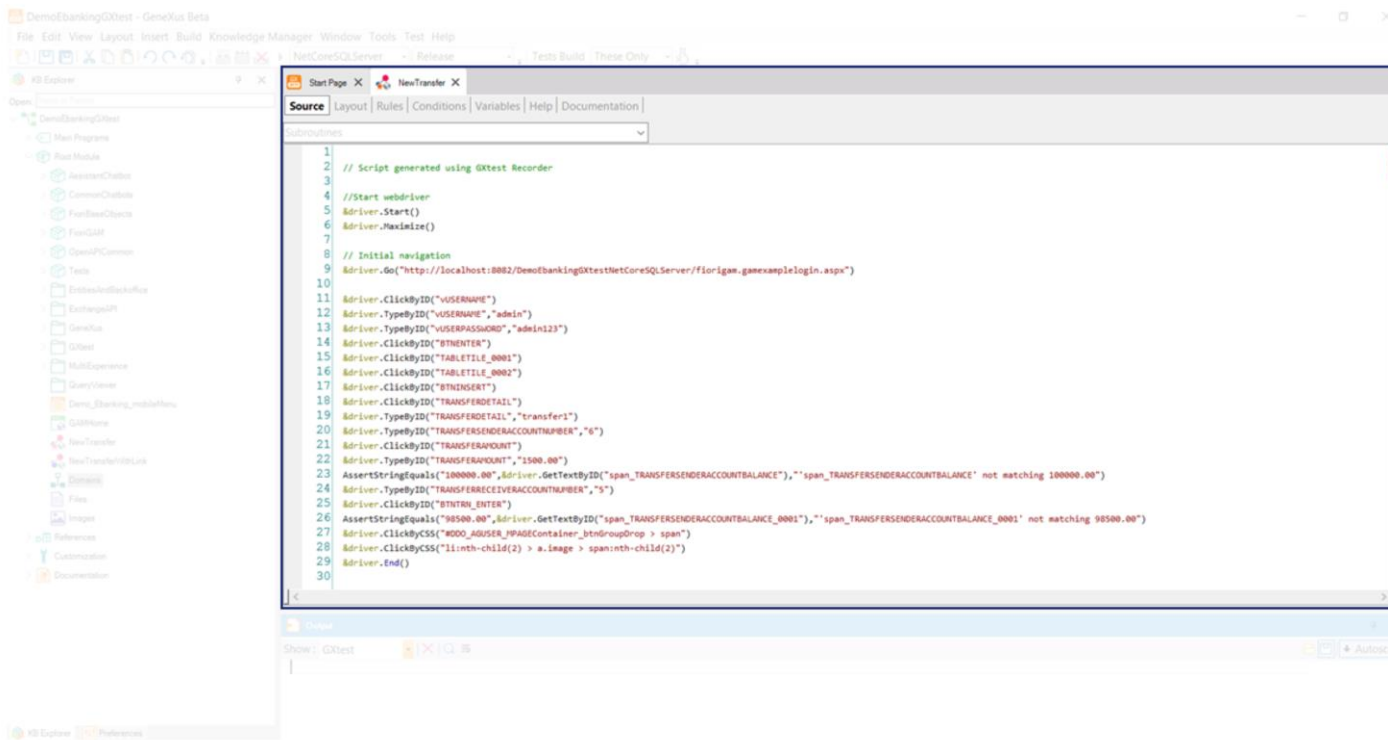


今回は、EbankのアプリケーションのURLを入力し、自動化するためのワークフローの記録を開始します。



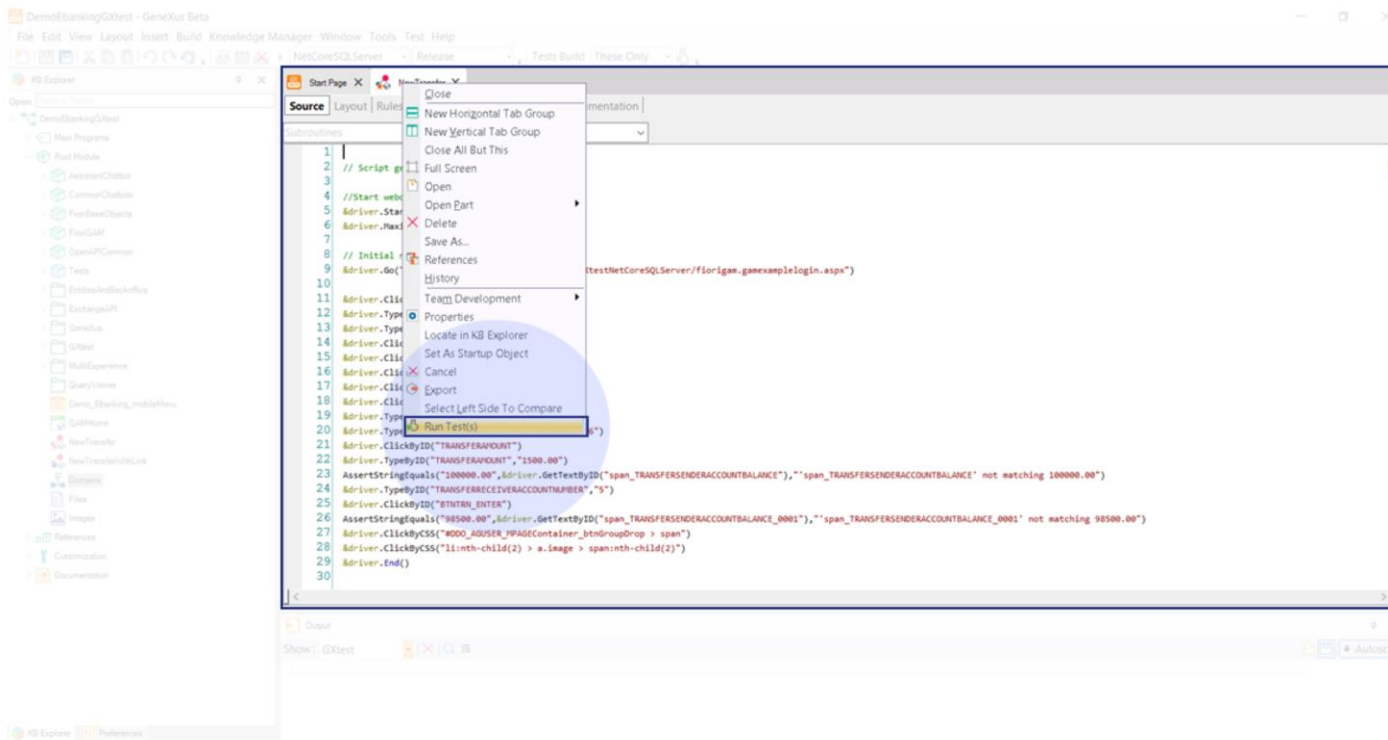
ワークフローの記録を完了すると、行われた操作はGXtest Recorderによってhtml参照のコマンドのように保存されます。このコマンドは、html参照を行うレコーダによって作成されていることに注意してください。つまり、コントロールへの参照はid、name、css、xpathによって行われます。

そして、GXtest RecorderのCopy to clipboardアイコンをクリックし、GeneXus IDEで作成したWeb UI Testオブジェクトに移動すると、Web UIテストのGeneXusコードが自動的にペーストされます。

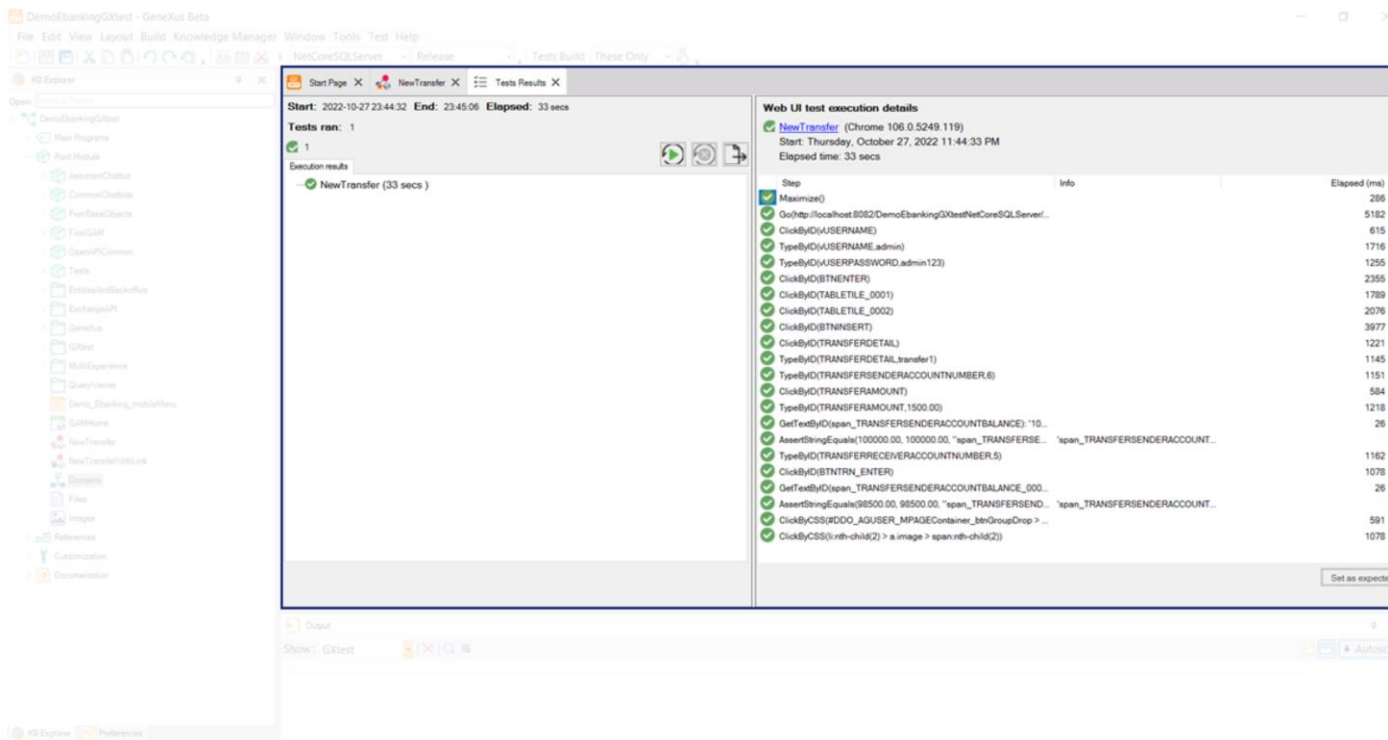


コードを見ていただければわかるように、Web UI Testオブジェクトは、ブラウザでアクションを実行するドライバ変数を持っており、アプリケーションコンポーネントはHTMLで参照されます。

これで、GeneXus IDEとパイプラインで実行するWeb UIテストの準備が整いました。

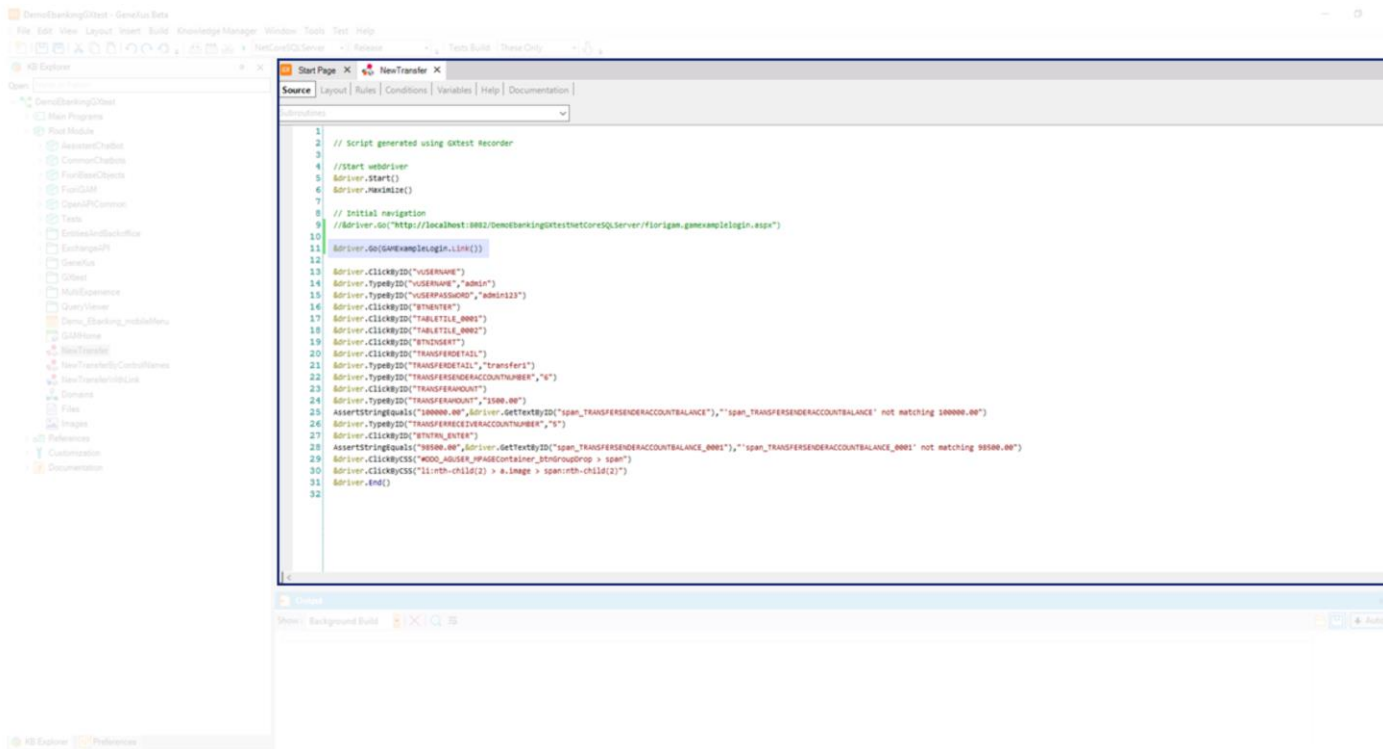


開いているWeb UI Testオブジェクトのタブを右クリックし、"テストの実行"オプションを選択すると、このテストを実行できます。



テストの実行後、テスト結果で実行日時、速度、ブラウザの情報を
 見ることができます。

また、コマンドの詳細が表示されるので、各テストケースの期待値
 と取得値を可視化することができます。



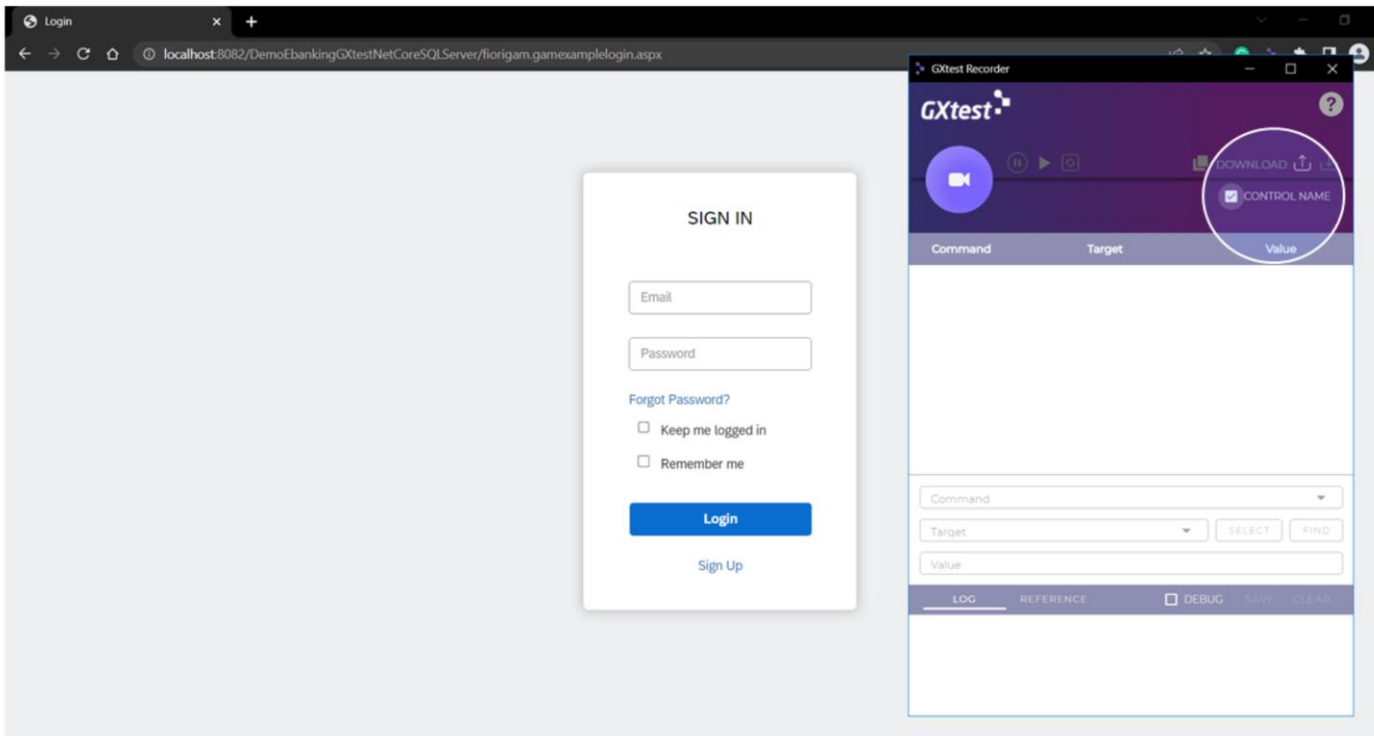
異なる環境で実行したい場合、WebPanelオブジェクトのLink()関数
 を利用し、起動するアプリケーションのホームURLを変更すること
 で対応できます。

コントロール名を用いたコマンド

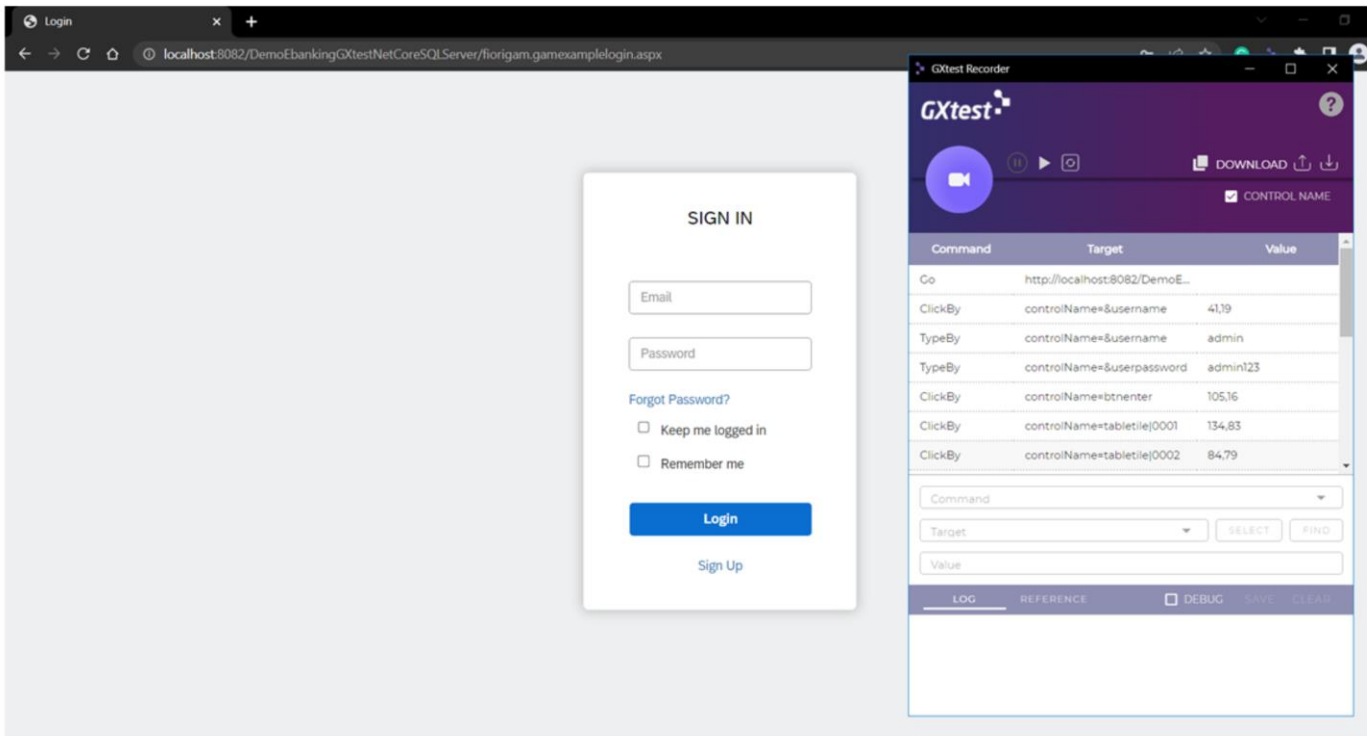


テストコマンドの要素をコントロール名によって参照してテストを作成する機能も
ございます。

これによりレコーディングをより高いレベルで抽象化し、より堅牢なテストを作成
することが可能となります。

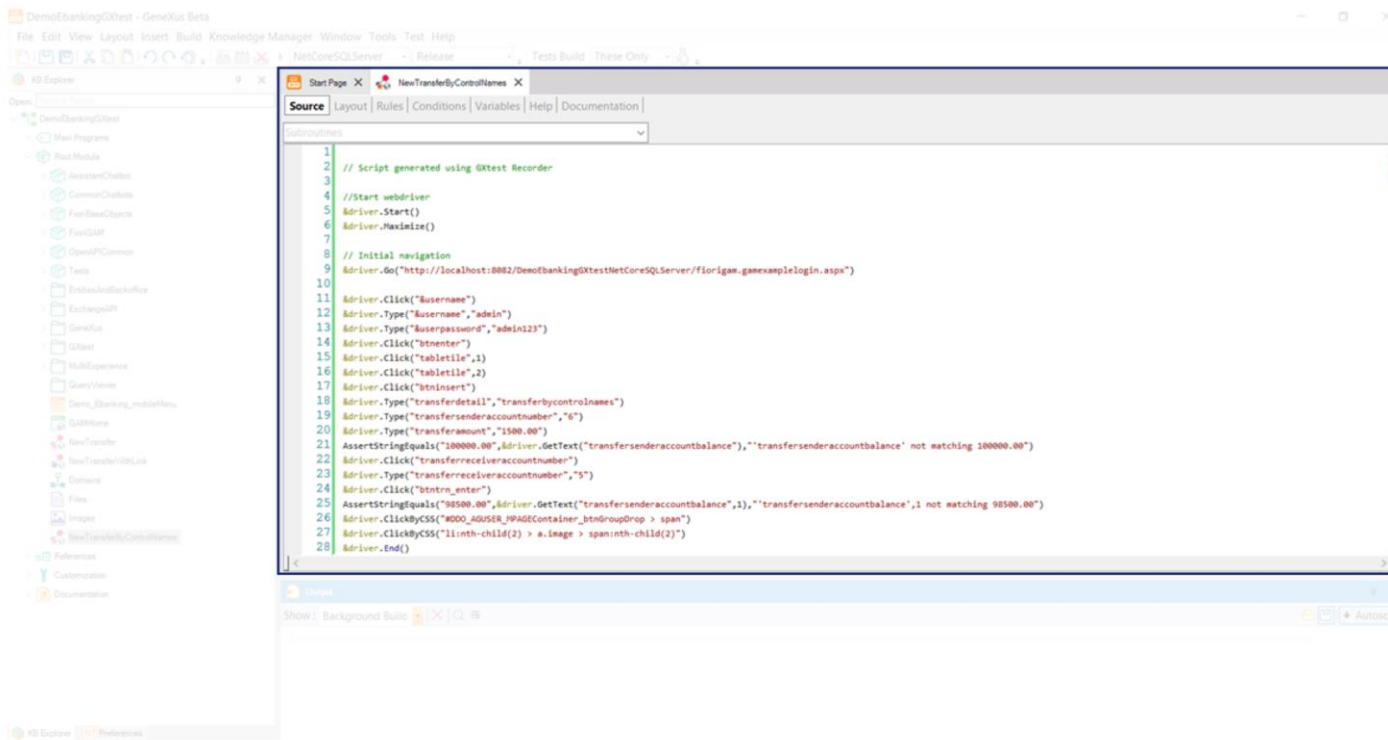


“CONTROL NAME”チェックボックスにチェックした場合、コントロール名で操作を記録することができます。



この記録では、GXtest Recorderはコマンドと同じアクションを保存しますが、要素のセレクトをHTMLで保存するのではなく、GeneXusのControl Nameセレクトで保存します。

記録された各アクションの各要素のcontrolNameセレクトを確認することができます。



GeneXusのコードでは、コントロール名を参照し、コマンドが定義されていることを確認できます。
今回の場合は「Click」と「Type」のコマンドです。

同じワークフロー上で、もしその要素がコントロール名を持っていないければ、最後のコマンド「ClickByCSS」のように、HTML参照で記録されます。

The screenshot displays the GeneXus IDE interface with the 'Tests Results' window open. The test 'NewTransferByControlNames' has completed successfully. The 'Web UI test execution details' pane on the right lists the steps taken during the test, including maximizing the browser, navigating to the application, logging in, and performing a transfer. The 'Output' pane at the bottom shows the command execution and the final result: 'OK. Elapsed: 29942 ms'.

Start: 2022-10-28 00:37:38 **End:** 00:38:08 **Elapsed:** 30 secs

Tests ran: 1

Execution results:

- 1 NewTransferByControlNames (29 secs)

Web UI test execution details

✓ NewTransferByControlNames (Chrome 106.0.5249.119)
Start: Friday, October 28, 2022 12:37:38 AM
Elapsed time: 29 secs

Step	Info	Elapsed (ms)
Maximize()		150
GoToUrl('localhost:8082/DemoEbankingGXTestNetCoreSQLServer...')		4166
Click('#username')		637
Type('#username', 'admin')		1776
Type('#password', 'admin123')		1265
Click('#enter')		3308
Click('#table1_1')		2161
Click('#table1_2')		2084
Click('#insert')		1847
Type('#transferdetail', 'transferbycontrolnames')		1313
Type('#transferamount', '1500.00')		1246
GetText('#transferamount', '100000.00')		118
AssertStringEquals('100000.00', '100000.00', 'transferamount not ...')		1327
Click('#transferreceiveraccountnumber')		621
Type('#transferreceiveraccountnumber', '5')		1169
Click('#btn_enter')		1118
GetText('#transferreceiveraccountbalance', '1')		61
AssertStringEquals('98500.00', '98500.00', 'transferreceiveraccountb...')		569
ClickByCSS('#XDOO_AJUSER_MPADEContainer_btnGroupDrop > ...')		1175
ClickByCSS('#nth-child(2) > a.image > span:nth-child(2)')		

Output

```
Show: GXTest
Command ClickByCSS('#XDOO_AJUSER_MPADEContainer_btnGroupDrop > span') OK took 569ms
Command ClickByCSS('#nth-child(2) > a.image > span:nth-child(2)') started
Command ClickByCSS('#nth-child(2) > a.image > span:nth-child(2)') OK took 1175ms
ENDED Web UI test NewTransferByControlNames. Result: OK. Elapsed: 29942 ms
*****
Execution ended successfully
Coverage data file for this execution was saved in 'C:\Models\DemoEbankingGXTest\NetCoreSQLServer004\web\gxtestTraceFile_20221028_123737.gxd'.
Success - Run Tests
```

テスト実行後、テスト結果で結果の詳細を見ることができます。

GeneXus[™]
by **Globant**

training.genexus.com
wiki.genexus.com