CloupiaScript インタープリタ

CloupiaScript インタープリタについて

CloupiaScript インタープリタは、ライブラリと API が組み込まれた JavaScript インタープリタです。CloupiaScript インタープリタを使用して、ワークフロー タスクを作成および実行することなく、CloupiaScript コードをテストすることができます。

CloupiaScript インタープリタには、次の組み込み関数が用意されています。

- `PrintObj()`：オブジェクトを引数として取り、オブジェクト内のすべてのプロパティとメソッドを出力します。出力された結果には、オブジェクト内の変数の名前と値、およびオブジェクトのすべての関数の名前が示されます。その後、いずれかのメソッド名で `toString()` を呼び出し、メソッド シグネチャを調べることができます。
- `Upload()`：ファイル名を引数として取り、ファイルの内容を CloupiaScript インタープリタにアップロードします。

CloupiaScript インタープリタの開始

CloupiaScript インタープリタを開くには、次の手順に従います。

ステップ 1 メニュー バーで、[ポリシー（Policies）] > [オーケストレーション（Orchestration）] を選択します。
ステップ 2 [カスタムワークフロータスク（Custom Workflow Tasks）] タブをクリックします。
ステップ 3 [インタープリタの起動（Launch Interpreter）] をクリックします。

[Cloupiaスクリプトインタープリタ（Cloupia Script Interpreter）] ダイアログボックスが表示されます。

ステップ 4 インタープリタ ダイアログの下部のテキスト入力フィールドをクリックします。

ステップ 5 JavaScript コードの行を入力して、Enter を押します。
コードが実行され、結果が表示されます。コードに構文エラーがある場合、そのエラーが表示されます。

コンテキストによる CloupiaScript インタープリタの開始

特定のカスタム タスクのコンテキストで JavaScript を評価できます。これを行うには、カスタム タスクを選択し、そのカスタム タスクを実行するために定義されたすべてのコンテキスト変数を使用して CloupiaScript インタープリタを起動します。

インタープリタを起動すると、インタープリタはユーザにカスタム タスクの入力フィールドに対する値を要求し、タスクの入力オブジェクトを入力します。実際にカスタム タスクを実行した場合に利用可能であったすべての変数が使用可能になります。

使用可能なコンテキストで CloupiaScript インタープリタを開くには、次の手順を実行します。

ステップ 1 メニュー バーで、[ポリシー（Policies）] > [オーケストレーション（Orchestration）] を選択します。

ステップ 2 [カスタムワークフロータスク（Custom Workflow Tasks）] タブをクリックします。

ステップ 3 JavaScript のテストの対象となるカスタム タスクを選択します。

ステップ 4 [コンテキストを使用してインタープリタを起動（Launch Interpreter with Context）] アクションをクリックします。
[インタープリタの起動（Launch Interpreter）] ダイアログボックスが表示されます。ここには、カスタム タスクの入力値を収集するための入力フィールドが示されます。入力フィールドは、選択したカスタム タスクに対して定義されたものです。

ステップ 5 フォームに入力値を入力します。

ステップ 6 [送信（Submit）] をクリックします。

ステップ 7 [送信（Submit）] をクリックします。
[Cloupiaスクリプトインタープリタ（Cloupia Script Interpreter）] ダイアログボックスが表示されます。

ステップ 8 インタープリタ ダイアログの下部のテキスト入力フィールドをクリックします。

ステップ 9 JavaScript コードの行を入力して、Enter を押します。
コードが実行され、結果が表示されます。コードに構文エラーがある場合、そのエラーが表示されます。

例：CloupiaScript インタープリタの使用

printObj 関数は、その関数に含まれているすべてのプロパティとメソッドを出力します。`functiontoString()` を呼び出して、関数に関する詳細を確認できます。次に、ReportContext クラスを調べて `ReportContext.setCloudName()` に関する詳細を取得する方法を示します。

```
session started
> importPackage(com.cloupia.model.cIM);
> var ctx = new ReportContext();
> printObj(ctx);
properties =
cloudName:null
class:class com.cloupia.model.cIM.ReportContext
filterId:null
id:null
targetCuicId:null
type:0
ids:[Ljava.lang.String;@4de27bc5
methods =
setIds
jdoReplaceField
jdoReplaceFields
toString
getCloudName
wait
getClass
jdoReplaceFlags
hashCode
jdoNewInstance
jdoReplaceStateManager
jdoIsDetached
notify
jdoGetVersion
jdoProvideField
jdoCopyFields
jdoGetObjectId
jdoGetPersistenceManager
jdoCopyKeyFieldsToObjectId
jdoGetTransactionalObjectId
getType
getFilterId
setType
jdoIsPersistent
equals
setCloudName
jdoNewObjectIdInstance
jdoIsDeleted
getTargetCuicId
setId
setFilterId
jdoProvideFields
jdoMakeDirty
jdoIsNew
requiresCloudName
getIds
notifyAll
jdoIsTransactional
getId
jdoReplaceDetachedState
jdoIsDirty
setTargetCuicId
jdoCopyKeyFieldsFromObjectId

> var func = ctx.setCloudName;
> func
void setCloudName(java.lang.String)
> func.toString();
```

```
function setCloudName() {/*
void setCloudName(java.lang.String)
*/}
```