REST を使用したソース・コード・ファイルへのアクセス

Version 2024.1
2024-06-03

InterSystems Corporation　1 Memorial Drive　Cambridge MA 02142　www.intersystems.com

REST を使用したソース・コード・ファイルへのアクセス
InterSystems IRIS Data Platform　Version 2024.1　2024-06-03

インターシステムズ社の製品に関するサポートやご質問は、以下にお問い合わせください：

InterSystems Worldwide Response Center (WRC)
Tel:　+1-617-621-0700
Tel:　+44 (0) 844 854 2917
Email:　support@InterSystems.com

目次

1

InterSystems IRIS ソース・コード・ファイル REST API の概要

この REST API は、InterSystems IRIS® データ・プラットフォームのソース・コード・ファイルへのアクセスを可能にします。インターシステムズの VS Code – ObjectScript IDE は、この API を使用して、InterSystems IRIS のコード・ファイルにアクセスするために必要なアクションを実行します。また、この API はユーザ独自のコードに使用できます。これらの操作としては、以下が挙げられます。

- InterSystems IRIS インスタンス上で使用できるネームスペースの取得
- ネームスペース内で定義されているクラス定義およびルーチンの検索
- クラスおよびルーチンのテキスト定義の取得
- クラス定義またはルーチンの更新
- 新しいクラス定義またはルーチンの作成
- クラス定義またはルーチンの削除
- InterSystems IRIS クラスまたはルーチンのコンパイル
- テーブルに対する SQL クエリの実行による InterSystems IRIS 環境のプロパティの検出

これらの操作は、InterSystems IRIS のソース・コード・ファイルにアクセスするためのメカニズムを提供します。InterSystems IRIS 開発環境を作成するためには、この API を理解して、InterSystems IRIS のソース・コード・ファイルが InterSystems IRIS 内でどのように使用されるのかを包括的に理解する必要があります。

これは特定目的用の API です。開発環境を作成する場合や、クラス・ブラウザなどの同様のアプリケーションを扱う場合は、この API が役に立つ可能性があります。ただし、これは InterSystems IRIS オブジェクトにアクセスするための汎用 REST API ではありません。

このドキュメントでは、InterSystems IRIS のソース・コード・ファイル REST API のバージョン 1 と 2 について説明します。将来の InterSystems IRIS リリースでは、追加の呼び出しを提供する、この REST API の上位バージョンがサポートされる可能性がありますが、いつでも以前のバージョンを呼び出すことができます。バージョン 1 の API では URL に `/v1/` が、バージョン 2 の API では `/v2/` が含まれています。InterSystems IRIS で提供されているこの API のバージョンを確認するには、GetServer メソッドを呼び出します。

以下では、この API の主な機能について、およびこれらの機能を提供するメソッドについて説明します。

- サーバ環境に関する情報の取得 :
  - GetServer メソッドは、サーバに関する重要な情報を提供します (サーバ上のネームスペースに関する情報を含む)。

  - GetNamespace メソッドは、指定されたネームスペースに関する追加情報を提供します。この情報には、そのネームスペースにマップされているデータベースのリストが含まれます。
  - HeadServer メソッドは、サーバに関するヘッダ情報を提供します。HeadServer を呼び出して、サーバが使用可能かどうかを確認できます。
  - GetJobs メソッドは、InterSystems IRIS で実行中のジョブに関する情報を提供します。
  - GetCSPApps メソッドは、サーバによって定義された Web アプリケーションに関する情報を提供します。これらのアプリケーションによって、InterSystems IRIS へのアクセスが提供されます。

- ソース・コード・ファイルに関する情報の取得:
  - GetDocNames メソッドは、ネームスペース内のソース・コード・ファイルの名前を返します。希望に応じて、対象となるファイルのファイル・カテゴリやファイル・タイプを限定できます。
  - GetModifiedDocNames メソッドは GetDocNames と同じ名前を返しますが、それに加えてデータベース状態のハッシュを返します。該当ファイルのローカル・コピーが保持されている場合は、GetModifiedDocNames を呼び出して、そのドキュメントが前回の取得時以降に変更されたかどうかを確認できます。
  - GetDoc メソッドは、指定されたソース・コード・ファイルの内容を取得します。希望に応じて ETAG および If-None-Match ヘッダを使用することで、ソース・コード・ファイルが前回の取得時以降に変更されている場合にのみ、ソース・コード・ファイルの内容を取得することもできます。
  - GetDocs メソッドは、指定されたファイルの内容を取得します。
  - Index メソッドは、ネームスペース内のクラス定義の一部のキー・プロパティを提供します。お使いのアプリケーションはこの情報を使用して、アクセスするクラス定義を選択できます。
  - HeadDoc メソッドは、ソース・コード・ファイルのヘッダ情報を提供します。

- ソース・コード・ファイルの作成、更新、および削除
  - PutDoc メソッドは、既存のソース・コード・ファイルを更新するか、既存のソース・コード・ファイルがない場合は新しいソース・コード・ファイルを作成します。
  - DeleteDoc メソッドは、指定されたソース・コード・ファイルを削除します。
  - DeleteDocs メソッドは、指定された一連のソース・コード・ファイルを削除します。

- ソース・コード・ファイルのコンパイル
  - Compile メソッドは、ソース・コード・ファイルをコンパイルします。

- InterSystems IRIS テーブルから情報を取得するための SQL クエリの実行
  - Query メソッドは、任意の InterSystems IRIS データベースに対する SQL クエリを実行します。

- ソース・コード・ファイルの検索
  - Search メソッドは、InterSystems IRIS データベース内のソース・コード・ファイルを検索します。

- Ensemble クラスを扱うための特別な呼び出し
  - GetEnsClassType メソッドは、Ensemble オブジェクトのクラス・タイプを返します。
  - GetAdpInputOutputClass メソッドは、プロダクション・アダプタの入力/出力アダプタ・クラスを返します。

2

ソース・コード・ファイル REST API のチュートリアル

このページでは、一連の例を通じて InterSystems IRIS® データ・プラットフォームのソース・コード・ファイル REST API の使用法を紹介する簡単なチュートリアルを提供します。

2.1 API の基礎

この API は、InterSystems IRIS のソース・コード・ファイルへのアクセスを可能にします。

API メソッドを呼び出すには、以下について知っておく必要があります。

- HTTP メソッド – GET、POSfT、PUT、DELETE、または HEAD のいずれかです。
- HTTP ヘッダ – 呼び出しのコンテキスト情報を提供します。この API で使用されるヘッダは次のとおりです。
  - 認証:サーバへのアクセスを可能にします。サーバを最小限のセキュリティでインストールしている場合を除いて、この API にアクセスするためにはユーザ名とパスワードを入力する必要があります。
  - `Content-Type application/json`:インバウンド・ペイロードが JSON で提供されていることを指定します。このヘッダは、すべての POST メソッドと PUT メソッドについて指定する必要があります。
  - `If-None-Match`:ソース・コード・ファイルが前回のアクセス以降に変更されたかどうかを GetDoc または PutDoc 呼び出しによって確認できるようにします。
- URL – URL は以下の要素で構成されています。
  - `https://`
  - baseURL`/` – このチュートリアルでは、InterSystems IRIS がサーバ `devsys` 上でポート 52773 を使用して実行されており、baseURL は `devsys:52773` であると想定しています。
  - `api/atelier/` – これは、%Api.Atelier ディスパッチ・クラスを持つ Web アプリケーションによって定義されます。
  - メソッドとターゲットを識別する URL 要素。この要素には、固定テキストを含めることができると共に、ネームスペース、ドキュメント名、またはタイプを特定するために指定するテキストを含めることができます。

  例えば、GetDocNames メソッドを識別する URL 要素は `v1/`namespace`/docnames/` です。MYNS ネームスペースからドキュメントを取得するこのメソッドの完全な URL は次のようになります。

```
https://devsys:52773/api/atelier/v1/MYNS/docnames
```

この URL 要素では、GetServer メソッドが空の文字列であることが指定されるため、GetServer の完全な URL は次のとおりです。

```
https://devsys:52773/api/atelier/
```

- URL パラメータ – 呼び出しを修正します。この API のメソッドで指定できる URL パラメータについては、リファレンスのセクションで説明しています。
- インバウンド JSON ペイロード – POST メソッドと PUT メソッドのインバウンド・メッセージのフォーマット。
- アウトバウンド JSON ペイロード – HTTP メソッドによって返されるアウトバウンド・メッセージのフォーマット。

注釈　この API のエンドポイントには、用語 `atelier` があります。Eclipse ベースの IDE であり、インターシステムズでは非推奨になっている Atelier で使用する目的で、この API が開発されたからです。インターシステムズの VS Code – ObjectScript IDE では、現在この同じ API を使用しています。

2.2 サーバに関する情報の取得

通常、最初に発行する REST 呼び出しは GetServer メソッドの呼び出しです。このメソッドは、InterSystems API のバージョン番号およびサーバ上で使用可能なネームスペースに関する情報を返します。

```
GET https://devsys:52773/api/atelier/
```

この呼び出しは、以下の JSON メッセージを返します。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": {
      "version": "IRIS for Windows (x86-64) 2018.1.1 (Build 515U) Mon Feb 5 2018 08:24:13 EST",
      "id": "98E1697E-13F9-4D6A-8B73-827873D1D61C",
      "api": 2,
      "features": [
...
      ],
      "namespaces": [
        "%SYS",
        "USER"
      ]
    }
    }
}
```

JSON メッセージを返すすべてのメソッドでは、以下のように同一の汎用フォーマットが使用されます。

- status errors – InterSystems IRIS ソース・コード・ファイル REST API は、通常、エラーを HTTP ステータス・コードとして返します。このフィールドはいくつかの特殊な状況で使用され、この要素には InterSystems IRIS の %Status 値が含まれており、この値には複数のエラーのテキストが含まれていることがあります。
- status summary – ステータス・エラーの要約情報が含まれています。
- console – この操作について InterSystems IRIS によってコンソールに表示されるテキストが含まれています。
- result – そのメソッドの結果が含まれています。

GetServer メソッドは、サーバに関する情報を “result” 要素内に返します。result 要素には “content” という 1 つの値が含まれており、この値には以下が含まれています。

- version – サーバ上で実行されている InterSystems IRIS のインスタンスのバージョン文字列が含まれています。
- id – InterSystems IRIS のインスタンス GUID が含まれています。
- api – このバージョンの InterSystems IRIS で実装されている InterSystems IRIS ソース・コード・ファイル REST API のバージョン番号が示されます。
- features – このインスタンスで有効になっている機能を示します。
- namespaces – InterSystems IRIS サーバ上で定義されたネームスペースがリストされます。

GetNamespace メソッドは、指定されたネームスペースに関する情報を返します。この情報には、そのネームスペースにマップされたデータベース、および各データベースのハッシュが含まれます。このハッシュは、サーバとの通信の効率を高めるのに役立ちます。ただし、GetServer によって返されるネームスペース情報のみを含めて、そのネームスペース内のソース・コード・ファイルに関する情報を取得できます。

2.3 ネームスペース内のソース・コードの取得

ネームスペース内のソース・コード・ファイルに関する情報を取得する手順は次のとおりです。

- 最初に、GetDocNames メソッドを使用してそれらのファイルの名前を取得します。
- 次に、GetDoc メソッドを使用して単一ファイルのコンテンツを取得するか、GetDocs メソッドを使用して複数ファイルのコンテンツを取得します。
- お使いのアプリケーションのネットワーク効率を高めるために、ソース・コード・ファイルの名前とコンテンツのローカル・キャッシュを保持しておいて、GetModifiedDocNames メソッドを使用して、コンテンツが変更されたソース・コード・ファイルの名前のみを取得することも、`If-None-Match` HTTP ヘッダを指定して GetDoc メソッドを使用することもできます。

GetDocNames メソッドは、指定されたネームスペースにマップされたすべてのデータベース内のすべてのソース・コード・ファイルの名前を返します。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": [
      {
        "name": "%Api.DocDB.cls",
        "cat": "CLS",
        "ts": "2016-08-03 20:01:42.000",
        "upd": true,
        "db": "IRISLIB",
        "gen": false
      },
      ...
      {
        "name": "EnsProfile.mac",
        "cat": "RTN",
        "ts": "2003-09-19 13:53:31.000",
        "upd": true,
        "db": "INVENTORYR",
        "gen": false
      },
      ...
      {
        "name": "xyz.mac",
        "cat": "RTN",
        "ts": "2016-08-11 15:05:02.167",
        "upd": false,
        "db": "INVENTORYR",
        "gen": false
      }
```

```
    ]
  }
}
```

次の GetDoc 呼び出しは xyz.mac ファイルのコンテンツを返します。

```
https://devsys:52773/api/atelier/v1/INVENTORY/doc/xyz.mac
```

この呼び出しは以下を返します。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "name": "xyz.mac",
    "db": "INVENTORYR",
    "ts": "2016-09-14 14:10:16.540",
    "upd": false,
    "cat": "RTN",
    "status": "",
    "enc": false,
    "flags": 0,
    "content": [
      "ROUTINE xyz",
      "xyz ;",
      "    w \"hello\""
    ]
  }
}
```

2.4 ネームスペース内の新規ファイルの作成または既存ファイルの更新

ネームスペース内で新規ファイルを作成するか、既存のファイルを更新するには、PutDoc メソッドを使用します。例えば、次の REST 呼び出しは、新しい xyz.mac ソース・コード・ファイルを INVENTORY ネームスペース内に作成するか、xyz.mac ファイルが存在する場合は、このファイルの元の定義を新しい定義に置き換えます。新しいファイルを更新する場合は、HTTP ヘッダ `If-None-Match` を指定してファイルの現在のバージョンを特定するか、`?ignoreConflict=1` URL パラメータを指定してバージョン確認を省略する必要があります。詳細は、リファレンス・セクションの ″PutDoc″ を参照してください。

```
PUT https://devsys:52773/api/atelier/v1/INVENTORY/doc/xyz.mac
```

Content-Type application/json と次の JSON メッセージを指定する必要があります。

```
{
 "enc": false,
 "content": [
   "ROUTINE xyz",
   "xyz ;",
   "    w \"hello\""
   ]
}
```

この呼び出しは、以下の JSON メッセージを返します。このメッセージは、このソース・コード・ファイルが INVENTORYR データベース内に作成されたことを示しています。このデータベースは、INVENTORY ネームスペース内のルーチンの既定データベースです。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "name": "xyz.mac",
    "db": "INVENTORYR",
    "ts": "2016-09-14 14:10:16.540",
    "upd": false,
    "cat": "RTN",
    "status": "",
    "enc": false,
    "flags": 0,
    "content": []
  }
}
```

バイナリ・ファイルを更新または作成する場合は、enc に true という値を指定して、バイナリ値の base64 エンコーディングのブロックの配列としてバイナリ・コンテンツを含めます。

2.5 ファイルのコンパイル

Compile メソッドは、受信 JSON 配列内の名前によって指定されたソース・コード・ファイルをコンパイルします。例えば、xyz.mac をコンパイルするには、以下を POST します。

```
https://devsys:52773/api/atelier/v1/INVENTORY/action/compile
```

この際に、次の JSON メッセージを含めます。

```
["xyz.mac"]
```

このメソッドは以下を返します。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [
    "",
    "Compilation started on 08/14/2016 15:25:20 with qualifiers 'cuk'",
    "xyz.int is up to date. Compile of this item skipped.",
    "Compilation finished successfully in 0.008s."
  ],
  "result": {
    "content": []
  }
}
```

クラスなどの一部のソース・コード・ファイルの場合は、Compile メソッドは返されるコンテンツ内に保管情報を返します。

2.6 ファイルの削除

DeleteDoc メソッドは、URL で指定されたファイルを削除します。DeleteDoc メソッドの URL は GetDoc メソッドと同じですが、唯一の相違点として、Get メソッドの代わりに HTTP Delete メソッドを使用します。xyz.mac を削除するには、次の URL を使用して HTTP DELETE 要求を発行します。

```
https://devsys:52773/api/atelier/v1/INVENTORY/doc/xyz.mac
```

この Delete メソッドは、以下の JSON メッセージを返します。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "name": "xyz.mac",
    "db": "INVENTORYR",
    "ts": "",
    "cat": "RTN",
    "status": "",
    "enc": false,
    "flags": 0,
    "content": []
  }
}
```

ファイルが削除されると、タイムスタンプ `ts` の値は "" (空の文字列) になります。

2.7 SQL クエリの実行

Query メソッドは、任意の InterSystems IRIS データベースに対する SQL クエリを実行します。例えば、アプリケーションで InterSystems IRIS のロールをユーザに一覧表示するには、次の呼び出しを使用してそれらのロールを検出できます。

```
POST https://devsys:52773/api/atelier/v1/%25SYS/action/query
```

この際に、受信 JSON メッセージ内で指定された SQL クエリを使用します。

```
{"query": "SELECT ID,Description FROM Security.Roles"}
```

この呼び出しは、SQL クエリの結果を JSON 形式で result content 要素内に返します。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": [
      {
        "ID": "%all",
        "Description": "The Super-User Role"
      },
      {
        "ID": "%db_%default",
        "Description": "R/W access for this resource"
      },
      ...
      {
        "ID": "%sqltunetable",
       "Description": "Role for use by tunetable to sample tables irrespective of row level security"

      }
```

```
    ]
  }
}
```

Query メソッドを使用して、InterSystems IRIS 内の任意のテーブルに対してクエリを実行できます。例えば、次の呼び出しでは、SAMPLES というネームスペースの Sample.Person というテーブルがクエリされます。クエリは、クエリ条件を満たす最大 2 つのレコード (`?max=2`) を要求します。

```
POST https://devsys:52773/api/atelier/v1/SAMPLES/action/query?max=2
{"query": "SELECT Age,SSN,Home_City,Name FROM Sample.Person WHERE Age = 25"}
```

この呼び出しは以下を返します。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": [
      {
        "Age": 25,
        "SSN": "230-78-7696",
        "Home_City": "Larchmont",
        "Name": "DeLillo,Jose F."
      },
      {
        "Age": 25,
        "SSN": "546-73-7513",
        "Home_City": "Gansevoort",
        "Name": "Klingman,Thelma H."
      }
    ]
  }
}
```

クエリ API は `positional` クエリ・パラメータもサポートしています。これにより、レコードのコンテンツに付随する配列で各レコードの列メタデータを受け取るオプションが提供されます。詳細は、Query() のクラス・リファレンスを参照してください。

3

ソース・コード・ファイル REST API のリファレンス

このページは、InterSystems IRIS® データ・プラットフォームのソース・コード・ファイル REST インタフェースに関するリファレンス情報を提供します。

注釈　この API のエンドポイントには、用語 `atelier` があります。Eclipse ベースの IDE であり、インターシステムズでは非推奨になっている Atelier で使用する目的で、この API が開発されたからです。インターシステムズの VS Code – ObjectScript IDE では、現在この同じ API を使用しています。

3.1 GetServer

このメソッドは、サーバに関する情報を返します。これには、ソース・コード REST API のバージョンや、サーバ上で使用可能なネームスペースに関する情報が含まれます。

例および追加情報は、"InterSystems IRIS サーバに関する情報の取得" を参照してください。

3.1.1 URL

GET `https://`<baseURL>`/api/atelier/`

<baseURL> は、インスタンスのベース URL です。

3.1.2 JSON メッセージ

以下の返されるコンテンツはサーバ記述子です。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": {
      "version": "IRIS for Windows (x86-64) 2018.1.1 (Build 515U) Mon Feb 5 2018 08:24:13 EST",
      "id": "98E1697E-13F9-4D6A-8B73-827873D1D61C",
      "api": 2,
      "features": [
...
      ],
```

```
      "namespaces": [
        "%SYS",
        "USER"
      ]
    }
    }
}
```

3.1.3 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.2 HeadServer

このメソッドは、サーバの HttpHeader を返します。

3.2.1 URL

HEAD `https://<baseURL>/api/atelier/`

<baseURL> は、インスタンスのベース URL です。

3.2.2 JSON メッセージ

返されるコンテンツはありません。

3.2.3 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.3 GetJobs

このメソッドは、InterSystems IRIS インスタンス上の実行中ジョブのリストを返します。

3.3.1 URL

GET `https://<baseURL>/api/atelier/v1/%25SYS/jobs`

<baseURL> は、インスタンスのベース URL です。

注釈　% は URL 特殊文字であるため、リテラル % を指定するには、% の後ろに 25 (パーセント記号の 16 進コード) を付加する必要があります。したがって、リテラル %SYS を指定するには %25SYS を使用する必要があります。

3.3.2 JSON メッセージ

以下の返されるコンテンツは、ジョブ記述子の配列です。


```
{
  "status": {
  "errors": [],
  "summary": ""
  },
  "console": [],
  "result": {
  "content": [
    {
    "pid": 1394,
    "namespace": "%SYS",
    "routine": "%Studio.Debugger.1",
    "state": "RUN",
    "device": "|TCP|1972|1394"
    },
    {
    "pid": 1345,
    "namespace": "%SYS",
    "routine": "RECEIVE",
    "state": "HANG",
    "device": "/dev/null"
    },
    {
    "pid": 1364,
    "namespace": "%SYS",
    "routine": "%SYS.TaskSuper.1",
    "state": "SELECTW",
    "device": "/dev/null"
    },
    {
    "pid": 1396,
    "namespace": "%SYS",
    "routine": "%SYS.cspServer3",
    "state": "READ",
    "device": "|TCP|1972|1396"
    },
    {
    "pid": 1346,
    "namespace": "%SYS",
    "routine": "ECPWork",
    "state": "RUNW",
    "device": "/dev/null"
    },
      {
        "pid": 1417,
        "namespace": "%SYS",
        "routine": "%SYS.BINDSRV",
        "state": "READ",
        "device": "|TCP|1972|1417"
      }
    ]
  }
}
```


3.3.3 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.4 GetMetaData

このメソッドは、指定されたデータベースの METADATA.zip ファイルのバイナリ・コンテンツを返します。Atelier はこのファイルにインデックス情報を保管して、今後のセッションのためにこの情報を保存できるようにします。

3.4.1 URL

GET `https://`<baseURL>`/api/atelier/v1/%25SYS/metadata/`database

<baseURL> は、インスタンスのベース URL です。

注釈　% は URL 特殊文字であるため、リテラル % を指定するには、% の後ろに 25 (パーセント記号の 16 進コード) を付加する必要があります。したがって、リテラル %SYS を指定するには %25SYS を使用する必要があります。

3.4.2 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 404:指定されたソース・コード・ファイルが存在しない場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.5 GetCSPApps

このメソッドは、サーバ上で定義されているかサーバ上の指定したネームスペース向けに定義されている Web アプリケーションのリストを返します。

3.5.1 URL

GET `https://`<baseURL>`/api/atelier/v1/%25SYS/cspapps`

GET `https://`<baseURL>`/api/atelier/v1/%25SYS/cspapps/`namespace

以下はその説明です。

<baseURL>

インスタンスのベース URL です。

namespace

ネームスペースの名前を指定します。namespace を指定しない場合、このメソッドはすべてのネームスペースの Web アプリケーションを返します。

注釈　% は URL 特殊文字であるため、リテラル % を指定するには、% の後ろに 25 (パーセント記号の 16 進コード) を付加する必要があります。したがって、リテラル %SYS を指定するには %25SYS を使用する必要があります。

3.5.2 URL パラメータ

URL パラメータの ?detail=1 を渡すことで、アプリケーションを詳しく記述しているオブジェクトが含まれた配列を返すことができます。

3.5.3 JSON メッセージ

以下の返されるコンテンツは、定義されている Web アプリケーションを列挙している配列です。


```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": [
      "/csp/broker",
      "/csp/documatic",
      "/csp/sys",
      "/csp/sys/exp",
      "/csp/sys/mgr",
      "/csp/sys/op",
      "/csp/sys/sec",
      "/isc/studio/rules",
      "/isc/studio/templates",
      "/isc/studio/usertemplates",
      "/csp/user"
    ]
  }
}
```


以下は、detail=1 を使用した場合の同じ返されるコンテンツです。


```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": [
      {
        "name": "/csp/broker",
        "default": false
      },
      {
        "name": "/csp/documatic",
        "default": false
      },
      {
        "name": "/csp/sys",
        "default": true
      },
      {
        "name": "/csp/sys/exp",
        "default": false
      },
      {
        "name": "/csp/sys/mgr",
        "default": false
      },
      {
        "name": "/csp/sys/op",
        "default": false
      },
      {
        "name": "/csp/sys/sec",
        "default": false
      },
      {
        "name": "/isc/studio/rules",
        "default": false
      },
      {
        "name": "/isc/studio/templates",
        "default": false
      },
      {
        "name": "/isc/studio/usertemplates",
        "default": false
      },
      {
        "name": "/csp/user",
        "default": true
```

```
      }
    ]
  }
}
```

3.5.4 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.6 GetNamespace

このメソッドは、指定されたネームスペースに関する情報を返します。

3.6.1 URL

GET `https://`<baseURL>`/api/atelier/v1/`namespace

<baseURL> は、インスタンスのベース URL です。

3.6.2 JSON メッセージ

以下は、ネームスペース USER に関して返されるコンテンツ情報です。

```
{
    "status": {
        "errors": [],
        "summary": ""
    },
    "console": [],
    "result": {
        "content": {
            "name": "USER",
            "db": [
                {
                    "name": "USER",
                    "crhash": "3A1A0E8B6C8",
                    "default": true,
                    "dbsys": false
                },
                {
                    "name": "IRISLIB",
                    "crhash": "A56AAA8D5418",
                    "default": false,
                    "dbsys": true
                },
                {
                    "name": "IRISLOCALDATA",
                    "crhash": "3A1A0551876",
                    "default": false,
                    "dbsys": false
                },
                {
                    "name": "IRISSYS",
                    "crhash": "3A19FFD2EF0",
                    "default": false,
                    "dbsys": true
                }
            ],
            "features": [
                {
                    "name": "ENSEMBLE",
                    "enabled": false
                }
            ]
```

```
        }
    }
}
```

3.6.3 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.7 GetDocNames

このメソッドは、ソース・コード・ファイル名のリストを返します。オプションの cat と type によって、ソース・コード・ファイルのタイプが限定されます。

例および追加情報は、"ネームスペース内で定義されたソース・コード・ファイルの取得" を参照してください。

3.7.1 URL

GET **https://**<baseURL>**/api/atelier/v1/**namespace**/docnames**

GET **https://**<baseURL>**/api/atelier/v1/**namespace**/docnames/**cat

GET **https://**<baseURL>**/api/atelier/v1/**namespace**/docnames/**cat**/**type

以下はその説明です。

<baseURL>

インスタンスのベース URL です。

cat

カテゴリ・コードを指定します (CLS = クラス、RTN = ルーチン、CSP = CSP ファイル (InterSystems IRIS での使用は非推奨)、OTH = その他)。既定値は * です。

type

ソース・コード・ファイルのタイプを指定します。* というワイルドカードまたはファイル・タイプを指定できます。CLS の場合は、タイプは * である必要があります。RTN の場合は、タイプは mac、int、inc、bas、mvi、または mvb のいずれでもかまいません。CSP ではタイプには、コンマで区切った js、css などのファイル・タイプのリストを指定できます。既定値は * です。

3.7.2 URL パラメータ

- URL パラメータの 'generated=1' は、生成されたソース・コード・ファイルを含める必要がある旨を指定します。
- URL パラメータの 'filter' は、名前に合致させるために使用できる SQL フィルタを提供します。

3.7.3 JSON メッセージ

以下の返されるコンテンツは、ソース・コード・ファイル記述子の配列です。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": [
      {
        "name": "%Api.DocDB.cls",
        "cat": "CLS",
        "ts": "2016-08-03 20:01:42.000",
        "upd": true,
        "db": "IRISLIB",
        "gen": false
      },
      ...
      {
        "name": "EnsProfile.mac",
        "cat": "RTN",
        "ts": "2003-09-19 13:53:31.000",
        "upd": true,
        "db": "INVENTORYR",
        "gen": false
      },
      ...
      {
        "name": "xyz.mac",
        "cat": "RTN",
        "ts": "2016-08-11 15:05:02.167",
        "upd": false,
        "db": "INVENTORYR",
        "gen": false
      }
    ]
  }
}
```

3.7.4 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.8 GetModifiedDocNames

このメソッドは、指定されたハッシュがデータベースで保持された時点以降に変更されたソース・コード・ファイルのリストを返します。このメソッドには、データベースのキーとハッシュのリストを JSON 配列として渡します。これらのハッシュ値に基づいて、そのキーによって定義されたデータベース内で何らかの内容が変更されたかどうかが判定されます。通常は、受信 JSON メッセージとして空の配列を使用してこの API を最初に呼び出します。これにより、各ファイルのデータベース・キーおよびデータベース・ハッシュと共に、ネームスペース内のすべてのソース・コード・ファイルの名前が返されます。その後、dbname と dbhash をポストして、前回の呼び出し以降にサーバ上で変更されたソース・コード・ファイルを特定できます。

確認対象となるソース・コード・ファイルのリストは、次の例のようにポストします。

```
[ { "dbname" : "USER",
    "dbhash" : "KWAGbOdnRblPzANaiv1Oiu0BZLI"
}, ... ]
```

3.8.1 URL

POST `https://`<baseURL>`/api/atelier/v1/`namespace`/modified/`type

以下はその説明です。

<baseURL>

インスタンスのベース URL です。

type

＊ または 3 文字コード (ls、mac、int、inc、bas、または mvi) で、ソース・コードのファイル・タイプを指定します。既定値は ＊ です。

この呼び出しでは、`Content-Type application/json` ヘッダが必要です。

3.8.2 JSON メッセージ

以下の返されるコンテンツは、ソース・コード・ファイル記述子の配列です。

```
[ { "dbname" : "USER",
    "dbhash" : "Qx1zuNaulq3b_1yR9ahZAfjkc-",
    "crhash" : "47763751EC",
    "docs": [{
            "name": "User.NewClass1.cls",
            "ts": "2016-01-04 14:00:04.000",
            "gen": false,
            "depl": false
            }, ... ]
   }, ... ]
```

指定した dbhash 以降にソース・コード・ファイルが削除された場合は、次のように空の文字列に設定されたタイム・スタンプと共に、そのソース・コード・ファイルがリスト内に返されます。

```
"ts": ""
```

マッピングによってデータベースがインクルードされており、マッピングが削除されている場合、dbhash と crhash の両方が "000" 値と共に返され、docs が空の配列として返されます。

3.8.3 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 400:ポストされたコンテンツが空の場合や、type が CLS 以外の場合。
- HTTP 415:コンテンツ・タイプが application/json でない場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.9 PutDoc

このメソッドは、指定されたソース・コード・ファイルを保存します。そのファイルが存在しない場合は、このメソッドはそのファイルを作成し、そのファイルが存在する場合は、このメソッドはその既存ファイルを指定されたファイルに置き換えます。確実に正しいバージョンのファイルを上書きするには、前回の PutDoc または GetDoc の ETAG ヘッダで返されたタイムスタンプ値で If-None-Match ヘッダを指定します。バージョンのチェックなしでファイルを上書きする場合は、URL パラメータ ?ignoreConflict=1 を指定します。このメソッドは、対応するソース・コード・ファイル・オブジェクトを返します。

バイナリ・ファイルを保存する場合は、受信 JSON メッセージの enc 要素を true に設定して、そのファイルのコンテンツを base64 のブロックの配列として組み込んでください。サーバ上のテキストが保存プロセス中に (例えばソース・コントロール・フックなどによって) 変更された場合は、更新されたテキストは、返されるソース・コード・ファイルのコンテンツ配列内に返されます。

ソース・コード・ファイルに関するエラーは、返されるソース・コード・ファイル・オブジェクトの status プロパティに格納されます。

バージョン 2 の PutDoc は、3 つの形式 (既定の UDL 形式、XML 形式、および従来の %RO エクスポート・ユーティリティで使用される形式) のファイル・コンテンツを受け入れます。PutDoc は、自動的にファイル・コンテンツの形式を認識します。

例および追加情報は、"ネームスペース内の新規ファイルの作成または既存ファイルの更新" を参照してください。

3.9.1 URL と入力 JSON メッセージ

PUT **https://**<baseURL>**/api/atelier/v1/**namespace**/doc/**doc-name

PUT **https://**<baseURL>**/api/atelier/v2/**namespace**/doc/**doc-name

<baseURL> は、インスタンスのベース URL です。

以下は、ソース・コード・ファイル xyz.mac に対する PutDoc の入力 JSON メッセージの例を示しています。

```
{
 "enc": false,
 "content": [
   "ROUTINE xyz",
   "xyz ;",
   "    w \"hello\""
   ]
}
```

注釈　CSP ファイル (InterSystems IRIS では非推奨) を作成する場合は、doc-name の値には / (スラッシュ) 文字が含まれます。そのため、PutDoc を定義する URLMap には :docname の代わりにこのパラメータ名用に (.*) が含まれています。詳細は、"REST の URL マップの作成" を参照してください。

3.9.2 URL パラメータ

URL パラメータの ?ignoreConflict=1 を渡すことで、ETAG チェックを省略できます。この例では、前回のアクセス以降にファイルが変更されていたとしても、ソース・コード・ファイルを強制的にサーバに書き込みます

3.9.3 HTTP ヘッダ

- If-None-Match ー 確実に正しいバージョンのファイルを上書きするには、前回の PutDoc または GetDoc の ETAG ヘッダで返されたタイムスタンプ値で If-None-Match ヘッダを指定します。

3.9.4 JSON メッセージ

以下は、ソース・コード・ファイル xyz.mac の PUT の返されるコンテンツです。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "name": "xyz.mac",
    "db": "INVENTORYR",
    "ts": "2016-09-14 14:10:16.540",
    "upd": false,
    "cat": "RTN",
    "status": "",
    "enc": false,
    "flags": 0,
    "content": []
  }
}
```

クラスの保存時、サーバによって正規化される可能性があるために、PutDoc は常にストレージ・セクションを返します。コンテンツが単なるストレージ・セクションである場合、'flags' json フィールドは 1 になります。PutDoc がコンテンツ内でクラス全体を返すか、またはコンテンツが空である場合、'flags' は 0 になります。

3.9.5 HTTP 返りコード

- HTTP 200:更新された場合。
- HTTP 201:作成された場合。
- HTTP 400:指定されたリソース名が無効なソース・コード・ファイル名である場合。
- HTTP 404:指定されたリソースが見つからない場合。
- HTTP 409:タイムスタンプに基づいて、サーバとクライアントのバージョン間に競合が検出された場合。ファイルが存在する場合に、PutDoc はこの値を返します。ただし、If-None-Match ヘッダでサーバ上のファイルの現行のタイムスタンプ値を指定した場合を除きます。競合が発生した場合、返されるメッセージにはサーバ上のソース・コード・ファイルのコンテンツが含まれます。
- HTTP 415:コンテンツ・タイプとして application/json が渡されなかった場合。
- HTTP 425:ソース・コード・ファイルがロックされているために書き込み不能である場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.10 GetDoc

このメソッドは、指定されたソース・コード・ファイルおよびネームスペースのテキストを返します。

返されるコンテンツには、ソース・コード・ファイル・オブジェクトが含まれています。

ソース・コード・ファイルに関するエラーは、このソース・コード・ファイル・オブジェクトの status プロパティに格納されます。ソース・コントロール・フックがそのネームスペースに対して有効になっている場合は、フックによって生成されたコンソール出力はすべて取得されて、'コンソール' 配列内の行の配列として返されます。

結果に含まれているのは、要求されたファイルの名前、そのファイルが保管されているデータベース、そのファイルのタイム・スタンプ、そのファイルのカテゴリ略称 (CLS = クラス、RTN = ルーチン、CSP = CSP ファイル [InterSystems IRIS での使用は非推奨]、OTH = その他)、および配列内に返されるソース・コード・ファイル・コンテンツです。

- テキスト・ファイルの場合は、これは文字列の配列となり、'enc' json フィールドは false に設定されます。
- バイナリ・ファイルの場合は、これは base64 でエンコードされたチャンクの配列となり、'enc' フィールドは true に設定されます。

バージョン 2 の GetDoc は、UDL 形式 (既定)、XML 形式、または従来の %RO ユーティリティで使用される形式でファイルのコンテンツを返すことができます。

例および追加情報は、"ネームスペース内で定義されたソース・コード・ファイルの取得" を参照してください。

3.10.1 URL

GET `https://`<baseURL>`/api/atelier/v1/`namespace`/doc/`doc-name

GET `https://`<baseURL>`/api/atelier/v2/`namespace`/doc/`doc-name

<baseURL> は、インスタンスのベース URL です。

注釈　CSP ファイル (InterSystems IRIS での使用は非推奨) を取得する場合は、doc-name の値には / (スラッシュ) 文字が含まれます。そのため、GetDoc を定義する URLMap には :docname の代わりにこのパラメータ名用に (.*) が含まれています。詳細は、"REST の URL マップの作成" を参照してください。

3.10.2 URL パラメータ

- URL パラメータの ?binary=1 を渡すことで、ソース・コード・ファイルをバイナリとして強制的にエンコードできます。
- URL パラメータの ?storageOnly=1 を渡すことで、クラスの保管部分のみを返すことができます。
- バージョン 2 では、URL パラメータ ?format= parameter を渡すことで、UDL 形式 (既定)、XML 形式、または従来の %RO ユーティリティで使用される形式でファイルのコンテンツを返すように指定できます。
  - ?format=udl
  - ?format=xml
  - ?format=%RO

  ?binary=1 を指定すると、GetDoc は format パラメータを無視します。

3.10.3 HTTP ヘッダ

- If-None-Match ー このファイルに対する前回の GetDoc または PutDoc 呼び出しから HTTP ETAG ヘッダで返された値を指定します。このファイルが前回の呼び出し以降に変更されていない場合は、GetDoc は HTTP 304 ステータスを返します。

3.10.4 JSON メッセージ

以下は、%Api.DocDB.cls を要求することによって返される結果の最初の部分の例です。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "name": "%Api.DocDB.cls",
    "db": "IRISLIB",
    "ts": "2016-09-13 22:31:24.000",
    "upd": true,
    "cat": "CLS",
    "status": "",
    "enc": false,
    "flags": 0,
    "content": [
       /// Routing class for the DocDB REST services",
       "Class %Api.DocDB Extends %DocDB.REST",
       "{",
...
```

以下は、?binary=1 を使用した場合の同じ要求の結果です。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "name": "%Api.DocDB.cls",
    "db": "IRISLIB",
    "ts": "2016-01-04 14:00:04.000",
    "cat": "CLS",
    "status": "",
    "enc": true,
    "content": [
      "Ly8vIFRoaXMgY2xhc3MgaXMgdGhlIHN1cGVyY2xhc3MgZm9yIGFsbCBl  ... PSAzIF0KewoKfQo="
    ]
  }
}
```

3.10.5 HTTP 返りコード

- HTTP 200：OK の場合。
- HTTP 304：指定されたソース・コード・ファイルが変更されていない場合 (https://en.wikipedia.org/wiki/HTTP_ETag を参照してください)。
- HTTP 400：指定されたリソースが有効なソース・コード・ファイル名でない場合。
- HTTP 404：指定されたソース・コード・ファイルが存在しない場合。
- HTTP 500：予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

[ソース・コード・ファイルが存在しません] などの「ソフト」エラーが発生した場合は、結果の「ステータス」フィールドに追加情報が含まれます。他のソフト・エラーの例としては、[ファイルがロックされています] などが挙げられます。例えば、以下が HTTP 404 返りコードと共に返される可能性があります。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "name": "xyz1.mac",
    "db": "",
    "ts": "",
    "cat": "RTN",
    "enc": false,
    "content": "",
    "status": "ERROR #16005: Document 'xyz1.mac' does NOT exist"
  }
}
```

3.11 DeleteDoc

このメソッドは、指定されたネームスペース内の指定されたソース・コード・ファイルを削除します。このメソッドは、対応するソース・コード・ファイル・オブジェクトを返します。

ソース・コード・ファイルに関するエラーは、このソース・コード・ファイル・オブジェクトの status プロパティに格納されます。

例および追加情報は、"ファイルの削除" を参照してください。

3.11.1 URL

DELETE **https://**<baseURL>**/api/atelier/v1/**namespace**/doc/**doc-name

<baseURL> は、インスタンスのベース URL です。

注釈　CSP ファイル [InterSystems IRIS での使用は非推奨] を削除する場合は、doc-name の値には / (スラッシュ) 文字が含まれます。そのため、DeleteDoc を定義する URLMap には :docname の代わりにこのパラメータ名用に (.*) が含まれています。詳細は、"REST の URL マップの作成" を参照してください。

3.11.2 JSON メッセージ

以下は、ソース・コード・ファイル xyz.mac の DELETE の返されるコンテンツです。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "name": "xyz.mac",
    "db": "INVENTORYR",
    "ts": "",
    "cat": "RTN",
    "status": "",
    "enc": false,
    "flags": 0,
    "content": []
  }
}
```

3.11.3 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 400:指定されたリソースが有効なソース・コード・ファイル名でない場合。
- HTTP 404:指定されたソース・コード・ファイルが存在しない場合。
- HTTP 423:指定されたリソースがロックされている場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.12 HeadDoc

このメソッドは、指定されたソース・コード・ファイルとネームスペースの HttpHeader を返します。このヘッダに含まれているタイムスタンプを使用して、サーバとクライアントのバージョンの不一致を検知できます。

3.12.1 URL

HEAD `https://`<baseURL>`/api/atelier/v1/`namespace`/doc/`doc-name

<baseURL> は、インスタンスのベース URL です。

注釈　CSP ファイル [InterSystems IRIS での使用は非推奨] の HTTP ヘッダを取得する場合は、doc-name の値には / (スラッシュ) 文字が含まれます。そのため、HeadDoc を定義する URLMap には :docname の代わりにこのパラメータ名用に (.*) が含まれています。詳細は、"REST の URL マップの作成" を参照してください。

3.12.2 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 400:指定されたリソース名が無効なソース・コード・ファイル名である場合。
- HTTP 404:指定されたリソースが見つからない場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.13 GetDocs

このメソッドは、指定されたネームスペース内の指定された全ソース・コード・ファイルのテキストを返します。

3.13.1 URL

POST `https://`<baseURL>`/api/atelier/v1/`namespace`/docs`

<baseURL> は、インスタンスのベース URL です。

取得するソース・コード・ファイルのリストを HTTP 要求の本体内に渡します。この要求本体は、取得するソース・コード・ファイルの名前の JSON 配列です。例 : `[ "%Api.DocDB.cls", ...]`

この呼び出しでは、Content-Type application/json ヘッダが必要です。

3.13.2 JSON メッセージ

返されるコンテンツは、ソース・コード・ファイル・オブジェクトの配列です。ソース・コード・ファイル・オブジェクトの構造の例については、"GetDoc" メソッドを参照してください。

ソース・コード・ファイルに関するエラーは、各ソース・コード・ファイル・オブジェクトの status プロパティに格納されます。このメソッドは storageOnly フラグをサポートしていません。このメソッドは ETAG チェックを実行しません (したがって、どのような状況でも HTTP 304 を返すことはありません)。

3.13.3 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 415:渡されたコンテンツ・タイプが application/json でない場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.14 DeleteDocs

このメソッドは、指定された一連のソース・コード・ファイルを削除します。このメソッドは、対応するソース・コード・ファイル・オブジェクト配列を返します。

3.14.1 URL

DELETE https://<baseURL>/api/atelier/v1/namespace/docs

<baseURL> は、インスタンスのベース URL です。

削除するファイルのリストは、JSON 配列として HTTP 要求の本体内に渡されます。例 : ["%Api.DocDB.cls", ...]

この呼び出しでは、Content-Type application/json ヘッダが必要です。

3.14.2 JSON メッセージ

以下は、ソース・コード・ファイル xyz.mac と存在しないクラス notexist.cls の DELETE の返されるコンテンツです。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [
  ],
  "result": [
    {
      "name": "xyz.mac",
      "db": "INVENTORYR",
      "status": ""
    },
    {
      "name": "notexist.cls",
      "db": "",
      "status": "ERROR #5001: Document Does Not Exist: User.notexist.cls
[zExistsDoc+3^%Atelier.v1.Utils.General.1:%SYS]"
    }
  ]
}
```

各ソース・コード・ファイルに関するエラーは、返される各ソース・コード・ファイル・オブジェクトの status プロパティに格納されます。ステータスが空の文字列の場合は、そのソース・コード・ファイルは正常に削除されました。そうでない場合は、そのソース・コード・ファイルは削除されませんでした。

削除されたソース・コード・ファイルについては、db プロパティに、そのドキュメントがどのデータベースから削除されたのかが示されます。

3.14.3 HTTP 返りコード

- HTTP 200：OK の場合。
- HTTP 400：ポストされたデータに JSON 配列が含まれていない場合。
- HTTP 415：渡されたコンテンツ・タイプが application/json でない場合。
- HTTP 500：予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.15 Compile

このメソッドは、ソース・コード・ファイルをコンパイルします。このメソッドは、複数のソース・コード・ファイルを同時にコンパイルできます。このメソッドは、対応するソース・コード・ファイル・オブジェクトの配列を返します。

コンパイルするファイルのリストは、JSON 配列として HTTP 要求の本体内に渡されます。例：`[ "%Api.DocDB.cls", ...]`

例および追加情報は、"ファイルのコンパイル" を参照してください。

3.15.1 URL

POST `https://`<baseURL>`/api/atelier/v1/`namespace`/action/compile`

<baseURL> は、インスタンスのベース URL です。

この呼び出しでは、`Content-Type application/json` ヘッダが必要です。

3.15.2 URL パラメータ

- URL パラメータの 'flags' (デフォルトは "cuk") を渡すことができます。このパラメータはコンパイラに渡されます。
- コンパイルされたソース・コード・ファイルのソースが返されないようにするには、URL パラメータの 'source' を 0 という値に設定して渡します。

3.15.3 JSON メッセージ

以下は、Atelier.NewClass1 をコンパイルする際に返されるコンテンツです。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [
    "Compilation started on 01/12/2016 17:44:00 with qualifiers 'cuk'",
    "Compiling class Atelier.NewClass1",
    "Compiling table Atelier.NewClass1",
    "Compiling routine Atelier.NewClass1.1",
    "Compilation finished successfully in 0.067s.",
    ""
  ],
  "result": {
    "content": [
      {
        "name": "Atelier.NewClass1.cls",
        "status": "",
        "content": [
          "Storage Default",
          "{",
          "<Data name=\"NewClass1DefaultData\">",
          "<Value name=\"1\">",
          "<Value>%%CLASSNAME</Value>",
          "</Value>",
          "</Data>",
          "<DataLocation>^Atelier.NewClass1D</DataLocation>",
          "<DefaultData>NewClass1DefaultData</DefaultData>",
          "<IdLocation>^Atelier.NewClass1D</IdLocation>",
          "<IndexLocation>^Atelier.NewClass1I</IndexLocation>",
          "<StreamLocation>^Atelier.NewClass1S</StreamLocation>",
          "<Type>%Storage.Persistent</Type>",
          "}",
          ""
        ],
        "db": "IRISSYS",
        "ts": "2016-01-12 17:44:00.053",
        "enc": false,
        "flags": 1
      }
    ]
  }
}
```

ソース・コード・ファイルに関するエラーは、各ソース・コード・ファイル・オブジェクトの status プロパティに格納されます。

永続クラスをコンパイルするとストレージ定義が変更される場合、ストレージ定義はソース・コード・ファイル・オブジェクトのコンテンツとして返されます。それ以外の場合は、結果コンテンツは空になります。

3.15.4 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 400:リソース名が無効なソース・コード・ファイル名である場合。
- HTTP 404:指定されたリソースが見つからない場合。
- HTTP 423:ソース・コード・ファイルがロックされている場合。

- HTTP 500: 予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.16 Index

このメソッドは、指定されたソース・コード・ファイルに関する要約情報を返します。お使いのアプリケーションはこの情報を使用して、そのソース・コード・ファイルに対するインデックスを作成できます。このメソッドは、インデックス・ソース・コード・ファイル・オブジェクトの配列を返します。

インデックスを作成するソース・コード・ファイルのリストは、HTTP 要求の本体内に渡されます。この要求本体は、ソース・コード・ファイルの名前の JSON 配列です。例 : `[ "%Api.DocDB.cls", ...]`

3.16.1 URL

POST `https://`<baseURL>`/api/atelier/v1/`namespace`/action/index`

<baseURL> は、インスタンスのベース URL です。

この呼び出しでは、`Content-Type application/json` ヘッダが必要です。

3.16.2 JSON メッセージ

ソース・コード・ファイルに関するエラーは、各ソース・コード・ファイル・オブジェクトの status プロパティに格納されています。返される配列には、サーバ上のソース・コード・ファイルの構造とドキュメントに関する情報が含まれています。これは、そのソース・コード・ファイルが属しているカテゴリによって異なります。以下は、クラス (カテゴリ CLS) の例です。(現在はクラスのインデックス作成のみがサポートされています。)

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": [
      {
        "name": "%Activate.GenericObject.cls",
        "db": "IRISLIB",
        "ts": "2016-01-04 14:00:04.000",
        "gen": false,
        "others": [
          "%Activate.GenericObject.1.INT"
        ],
        "cat": "CLS",
        "content": {
          "desc": "This class provides functionality to create an ActiveX object, invoke its methods
and Get/Set its properties by name.",
          "depl": false,
          "depr": false,
          "final": false,
          "hidden": false,
          "super": [
            "%Activate.IDispatch"
          ],
          "methods": [
            {
              "name": "CreateObject",
              "desc": "This method is used to create a generic object given only its progid. If the
object cannot be found an exception is thrown.
                       The return value should be tested against $$$NULLOREF  in the usual manner to
ensure that the object has been successfully created",
              "depr": false,
              "final": true,
              "internal": false,
              "private": false,
              "scope": "class",
              "returntype": "%Library.RegisteredObject",
```

```
              "args": [
                {
                  "name": "Progid",
                  "type": "%Library.String"
                }
              ]
            },
            {
              "name": "GetObject",
              "desc": "This method is used to create a generic object from a moniker. If the object
cannot be found an exception is thrown.
                       The return value should be tested against $$$NULLOREF  in the usual manner to
ensure that the object has been successfully created.",
              "depr": false,
              "final": true,
              "internal": false,
              "private": false,
              "scope": "class",
              "returntype": "%Library.RegisteredObject",
              "args": [
                {
                  "name": "Moniker",
                  "type": "%Library.String"
                }
              ]
            }
          ],
          "parameters": [],
          "properties": []
        },
        "status": ""
      }
    ]
  }
}
```


3.16.3 HTTP 返りコード

- HTTP 200：OK の場合。
- HTTP 415：渡されたコンテンツ・タイプが application/json でない場合。
- HTTP 500：予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.17 Query

このメソッドは、InterSystems IRIS テーブルに対する SQL クエリを実行して、結果を返します。要求の本体は、そのクエリを指定する JSON オブジェクトです。このメソッドは、クエリ条件に一致するオブジェクトの配列を返します。それぞれの返されるオブジェクトには、そのクエリによって返される単一行に関する情報が含まれています。サポートされるパラメータについては、Query() のクラス・リファレンスを参照してください。

3.17.1 URL

POST `https://`<baseURL>`/api/atelier/v1/`namespace`/action/query`

<baseURL> は、インスタンスのベース URL です。

SQL クエリは、URL 要求の本体内に指定します。クエリでは、指定されたネームスペース内のデータベースを指定する必要があります。

この呼び出しでは、`Content-Type application/json` ヘッダが必要です。

3.17.2 JSON メッセージ

返されるコンテンツは、オブジェクトの配列です。エラーは、各ソース・コード・ファイル・オブジェクトの status プロパティに格納されます。

```
{
  "status": {
    "errors": [],
    "summary": ""
  },
  "console": [],
  "result": {
    "content": [
      {
        "ID": "%all",
        "Description": "The Super-User Role"
      },
      {
        "ID": "%db_%default",
        "Description": "R/W access for this resource"
      },
      {
        "ID": "%db_irislocaldata",
        "Description": "R/W access for this resource"
      },
      ...
      {
        "ID": "%sqltunetable",
        "Description": "Role for use by tunetable to sample tables irrespective of row level security"

      }
    ]
  }
}
```

3.17.3 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 415:渡されたコンテンツ・タイプが application/json でない場合。
- HTTP 500:予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.18 Search

このメソッドは、指定されたコンテンツで InterSystems IRIS データベース内のファイルを検索します。Search メソッドは API のバージョン 2 で使用できます。このメソッドは、ユーザに向けて表示することを目的とした形式で検索結果を返します。

3.18.1 URL

POST `https://`<baseURL>`/api/atelier/v2/`namespace`/action/search`

<baseURL> は、インスタンスのベース URL です。

3.18.2 URL パラメータ

- 必須の URL パラメータ ?query=expression では、指定したファイル内を検索するための正規表現またはテキスト文字列を指定します。

- 必須の URL パラメータ ?files=file-list では、指定した expression を検索するファイルのコンマ区切りリストまたはファイル・マスク (例えば、al*.mac) を指定します。
- オプションの URL パラメータ ?regex=1 は、URL パラメータ query に正規表現が含まれていることを指定します (既定)。?regexp=0 は、query にテキスト文字列が含まれていて、正規表現として解釈してはいけないことを指定します。
- オプションの URL パラメータ ?sys=1 は、検索にシステム・ファイルを含めることを指定します。既定値は ?sys=0 で、システム・ファイルが除外されます。
- オプションの URL パラメータ ?gen=1 は、生成されたファイルを検索に含めることを指定します。既定値は ?gen=0 で、生成されたファイルが除外されます。
- オプションの URL パラメータ ?max=integer は、返す結果の最大数を指定します。既定値は ?max=200 です。

3.18.3 JSON メッセージ

以下の search REST 呼び出しは、前後にスペースが付いた “Email” という単語について、すべての .cls ファイルと .mac ファイルを検索します (正規表現では、¥s はスペース文字と一致します)。

```
GET
localhost:52773/api/atelier/v2/SAMPLES/action/search?query=.*\sEmail\s.*&files=*.cls,*.mac
```

この呼び出しは、以下のメッセージを返します。返されるメッセージは、SAMPLES ネームスペースのコンテンツに応じて異なる場合があります。

```
{
    "status": {
        "errors": [],
        "summary": ""
    },
    "console": [
        "",
        "Searching for '.*\\sEmail\\s.*' in '*.cls,*.mac'",
        "Wasabi.Data.Employee.cls(Email): Property Email ",
        "Wasabi.Person.API.Employee.cls(Email): Property Email ",
        "ZAUTHENTICATE.mac(175): Properties(\"EmailAddress\") - Email address",
        "Found 3 occurrence/s in 3 file/s."
    ],
    "result": [
        {
            "doc": "Wasabi.Data.Employee.cls",
            "matches": [
                {
                    "member": "Email",
                    "text": "Property Email "
                }
            ]
        },
        {
            "doc": "Wasabi.Person.API.Employee.cls",
            "matches": [
                {
                    "member": "Email",
                    "text": "Property Email "
                }
            ]
        },
        {
            "doc": "ZAUTHENTICATE.mac",
            "matches": [
                {
                    "line": "175",
                    "text": "Properties(\"EmailAddress\") - Email address"
                }
            ]
        }
    ]
}
```

3.18.4 HTTP 返りコード

- HTTP 200:要求が有効な場合。
- HTTP 400:必須の URL パラメータが不足している場合。

3.19 GetEnsClassType

このメソッドは、プロダクションの作成に使用することを目的としたクラスの名前のリストを返します。取得するクラスのタイプ (ビジネス・サービス・クラスなど) を指定できます。

3.19.1 URL

GET `https://`<baseURL>`/api/atelier/v1/`namespace`/ens/classes/`type

以下はその説明です。

<baseURL>

インスタンスのベース URL です。

type

ここでは整数であり、以下のようにその整数に対応するクラスを返します。

Adapters 1

InboundAdapters 2

OutboundAdapters 3

Messages 4

Requests 5

Responses 6

BusinessServices 7

BusinessProcesses 8

BusinessOperations 9

DataTransformation 10

Production 11

BusinessHost 12

Dashboard 13

Rule 14

3.19.2 JSON メッセージ

以下の返されるコンテンツは、クラス名の配列です。

```
{
  status: {
    errors: []
    summary: ""
  }
  console: []
  result: {
    content: [
        "Ens.Enterprise.MsgBank.BankTCPAdapter",
        "Ens.Enterprise.MsgBank.ClientTCPAdapter",
        "Ens.InboundAdapter",
        "Ens.OutboundAdapter"
    ]
  }
}
```

3.19.3 HTTP 返りコード

- HTTP 200 : OK の場合。
- HTTP 500 : 予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。

3.20 GetAdpInputOutputClass

このメソッドは、指定されたプロダクション・アダプタの入力/出力タイプを返します。

3.20.1 URL

GET `https://`<baseURL>`/api/atelier/v1/`namespace`/ens/adapter/`name

<baseURL> は、インスタンスのベース URL です。

3.20.2 JSON メッセージ

以下は、返されるコンテンツの例です。

```
{
  status: {
    errors: []
    summary: ""
  }
  console: []
  result: {
    content: {
      input: "%Stream.Object"
      output: "%String"
    }
  }
}
```

3.20.3 HTTP 返りコード

- HTTP 200:OK の場合。
- HTTP 404:指定されたアダプタが存在しない場合。

- HTTP 500: 予期しないエラーが発生した場合 (詳細情報はステータス・エラー配列に格納されます)。