



# IRISファーストステップ・ガイド

(IRIS Version 2024.1 ベース)

V1.01

2024年12月

インターシステムズジャパン株式会社

<b>1.</b>	<b>はじめに</b>	<b>3</b>
<b>2.</b>	<b>ユーティリティ</b>	<b>5</b>
<b>3.</b>	<b>最初のIRISプログラム</b>	<b>16</b>
<b>4.</b>	<b>IRIS オブジェクト</b>	<b>20</b>
<b>5.</b>	<b>IRIS SQL</b>	<b>34</b>

## 1. はじめに

ようこそIRISの世界へ！

本ガイドは、IRISを初めて操作する方を対象に、IRISでのプログラミングの基本を解説します。

本ガイドでは、ユーザの皆さんがIRISでの開発を始められるのに必要な事項をStep-by-Step形式で説明していきます。

評価版のIRISを実際にインストールして操作しながらお読みいただければ、理解が深まると思います。

評価版のIRISは開発者コミュニティサイトのリソース集の所のInterSystems IRISダウンロードから入手できます。

入手に先立ち開発者コミュニティのアカウント登録が必要です。

評価方法として以下のページでも解説しています。

<https://faq.intersystems.co.jp/csp/faq/result.csp?DocNo=453>

なお、本ガイドには、インストールの解説は含まれていません。

インストール方法解説については、弊社WebサイトにあるIRISドキュメントをご参照ください。

[https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=PAGE\\_deployment\\_install](https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=PAGE_deployment_install)

本ガイドをやり終えれば、IRISについて開発をスタートする必要最小限の知識が得られます。

さらに情報が必要な方は、オンラインドキュメントやその他の技術情報をご参照ください。

それでは、IRISをインストールすると標準機能として提供されるユーティリティについての説明から、始めていきましょう。

実際にIRISをインストールしていただき、手を動かしながら、IRISに触れてみてください！

本ガイドには、インストール時に設定する初期セキュリティ設定で「最小」を選択した状態での内容を記載しています。

インストール時初期セキュリティ設定の種類については、最小／標準／ロックダウンが選択できます。

初期セキュリティ設定詳細は以下URLをご参照ください。

[https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=ROARS\\_gsecuring\\_plan\\_KEY=GCI\\_security#GCI\\_security\\_init\\_settings](https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=ROARS_gsecuring_plan_KEY=GCI_security#GCI_security_init_settings)

セキュリティ全体の詳細説明については以下URLをご参照ください。

[https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=PAGE\\_security](https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=PAGE_security)

## 2. ユーティリティ

ここでは、IRISのユーティリティについて説明します。

IRISをインストールすると、IRISを操作するために必要なユーティリティが、標準でインストールされます。

ユーティリティ起動方法は、IRISインストール後、Windowsのタスクバーに現れるIRISランチャーをクリックし、現れるメニューから起動します。

これらユーティリティは、Windows上でのみ、ご利用いただけます。

Windows以外のOSへインストールしたときは、IRISクライアント機能をWindows上にインストールしていただき、指定のIRISサーバへ接続し、これらユーティリティをお使いいただけます。

なお、管理ポータルは Webベースで動作するユーティリティ化するため、Windows以外でも利用可能です。

IRISクライアントインストール方法、IRISサーバへの接続設定については、以下ドキュメントをご参照ください。

IRISクライアントインストールについて：

[https://docs.intersystems.com/iris20241/csp/docbookj/Doc.View.cls?KEY=GIWIN\\_client](https://docs.intersystems.com/iris20241/csp/docbookj/Doc.View.cls?KEY=GIWIN_client)

IRIS サーバへの接続設定について：

[https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=GSA\\_connect\\_remote#GSA\\_connect\\_remote\\_define](https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=GSA_connect_remote#GSA_connect_remote_define)

では、ユーティリティを起動してみましょう！

- ① まず IRISランチャーをクリックまたは、右クリックします。

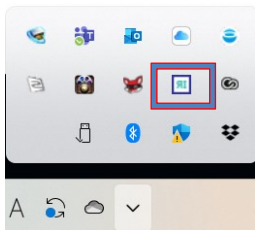
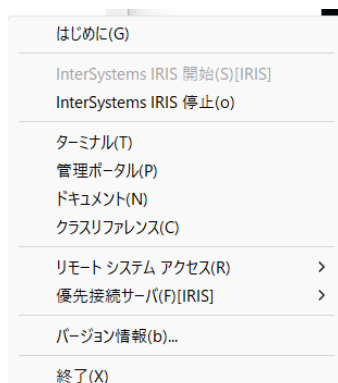


図1 IRIS ランチャーのメニュー

- ② 現れたメニューから、ユーティリティを選択します。



## 2.1. ドキュメント

IRISランチャーからIRISのドキュメントを表示させることができます。

本ガイド内に、ドキュメントへのリンクがいくつかありますが、この説明以降のドキュメントについては、IRISランチャーから起動するドキュメントへのリンクでご紹介しています。

また、IRISのドキュメントは、InterSystemsの独自Web技術であるCache Server Page(CSP)を使用しています。

## 2.2. Visual Studio Code

Visual Studio Codeの使用方法については以下のリンクを参照ください。

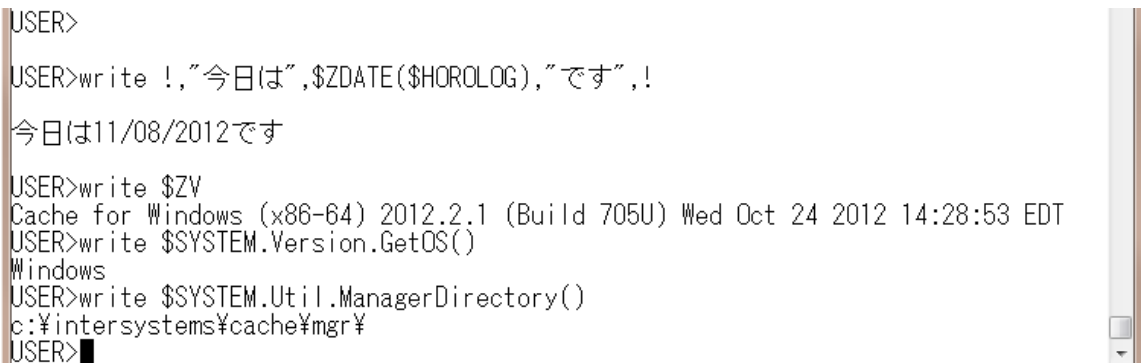
[Visual Studio Codeの使い方](#)

## 2.3. ターミナル

IRISの端末エミュレータとなります。

ターミナルでは、ObjectScript のコマンド、システムユーティリティなどを実行できます。

アプリケーションを作成する際、作成したルーチンの動作確認などにも使用できます。



```
USER>
USER>write !,"今日は",$ZDATE($HOROLOG),"です",!
今日は11/08/2012です
USER>write $ZV
Cache for Windows (x86-64) 2012.2.1 (Build 705U) Wed Oct 24 2012 14:28:53 EDT
USER>write $SYSTEM.Version.GetOS()
Windows
USER>write $SYSTEM.Util.ManagerDirectory()
c:\intersystems\cache\mgr\
USER>
```

## 2.4. 管理ポータル

IRISの統合管理環境である管理ポータルでは、大きく分けて以下3つの機能を提供しています。

管理ポータル画面について詳細は以下ドキュメントをご参照ください。

### [管理ポータル](#)

起動には、IRISランチャーのメニューを選択されるか以下URLを起動して下さい。

(ポート番号は環境により異なる場合があります)

<http://localhost:52773/csp/sys/UtilHome.csp>



- システム管理（システム管理者タスク）

サーバ   ネームスペース [USER](#) ユーザ [Unknown](#)

# ようこそ UnknownUser

表示: □□□

<div style="margin-bottom: 10px;"> <b>ホーム</b> </div> <div style="margin-bottom: 10px;"> <b>Analytics</b> </div> <div style="margin-bottom: 10px;"> <b>Interoperability</b> </div> <div style="margin-bottom: 10px;">  システムオペレーション         </div> <div style="margin-bottom: 10px;">  システムエクスプローラ         </div> <div> <span style="border: 2px solid #8b0000; padding: 2px 5px;">システム管理</span> </div>	<div style="margin-bottom: 10px;">           構成 <span style="float: right;">&gt;</span> </div> <div style="margin-bottom: 10px;">           セキュリティ <span style="float: right;">&gt;</span> </div> <div style="margin-bottom: 10px;">           ライセンス <span style="float: right;">&gt;</span> </div> <div>           暗号化 <span style="float: right;">&gt;</span> </div>
---	--

図2 管理ポータル：システム管理

システムの構成情報（ネームスペース・データベースの作成、ECP の設定、起動環境調整用項目など）、セキュリティ管理、ライセンス管理、データベースの暗号化 を用意しています。

詳細については以下ドキュメントをご参照ください。

構成について

[https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=GSA\\_config\\_system](https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=GSA_config_system)

セキュリティについて

[https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=PAGE\\_security](https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=PAGE_security)

ライセンスについて

[https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=GSA\\_license](https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=GSA_license)

データベースの暗号化について

[https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=AENCRYPT/DocBook.UI.Page.cls?KEY=GCAS\\_encrypt](https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=AENCRYPT/DocBook.UI.Page.cls?KEY=GCAS_encrypt)

- システムエクスプローラー（データ管理）

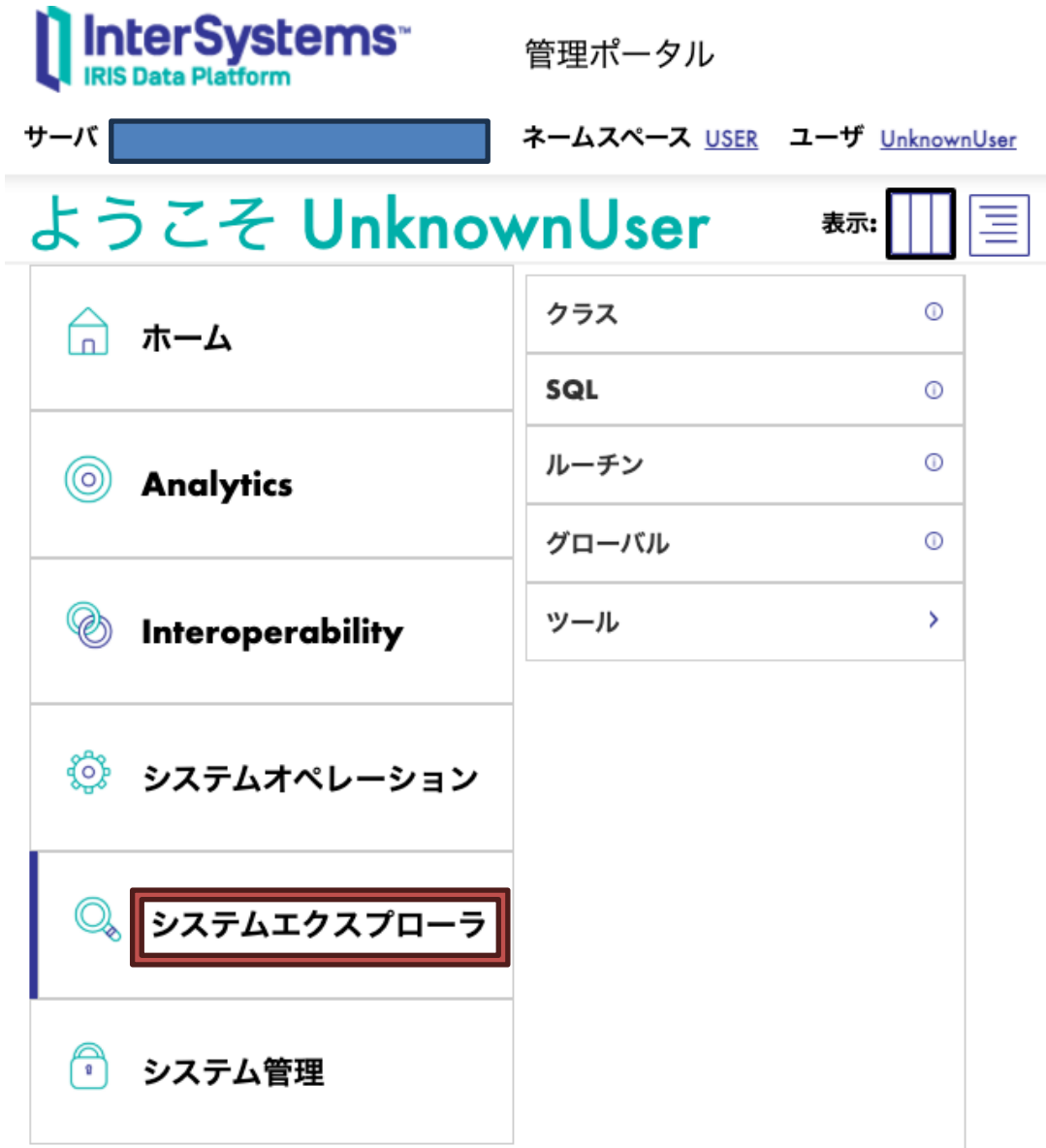


図3 管理ポータル：システムエクスプローラー

システムエクスプローラーには、データ管理用に、クラス定義、ルーチン、グローバル変数の一覧やインポート／エクスポート機能、SQL文操作メニューやテーブルなど用意しています。

SQLの項目では、IRISクラス定義により定義されたIRISのデータをテーブル形式で参照することができます。  
また SQL文の実行、ASCII ファイルからのデータインポートやエクスポート機能など取り揃えています。

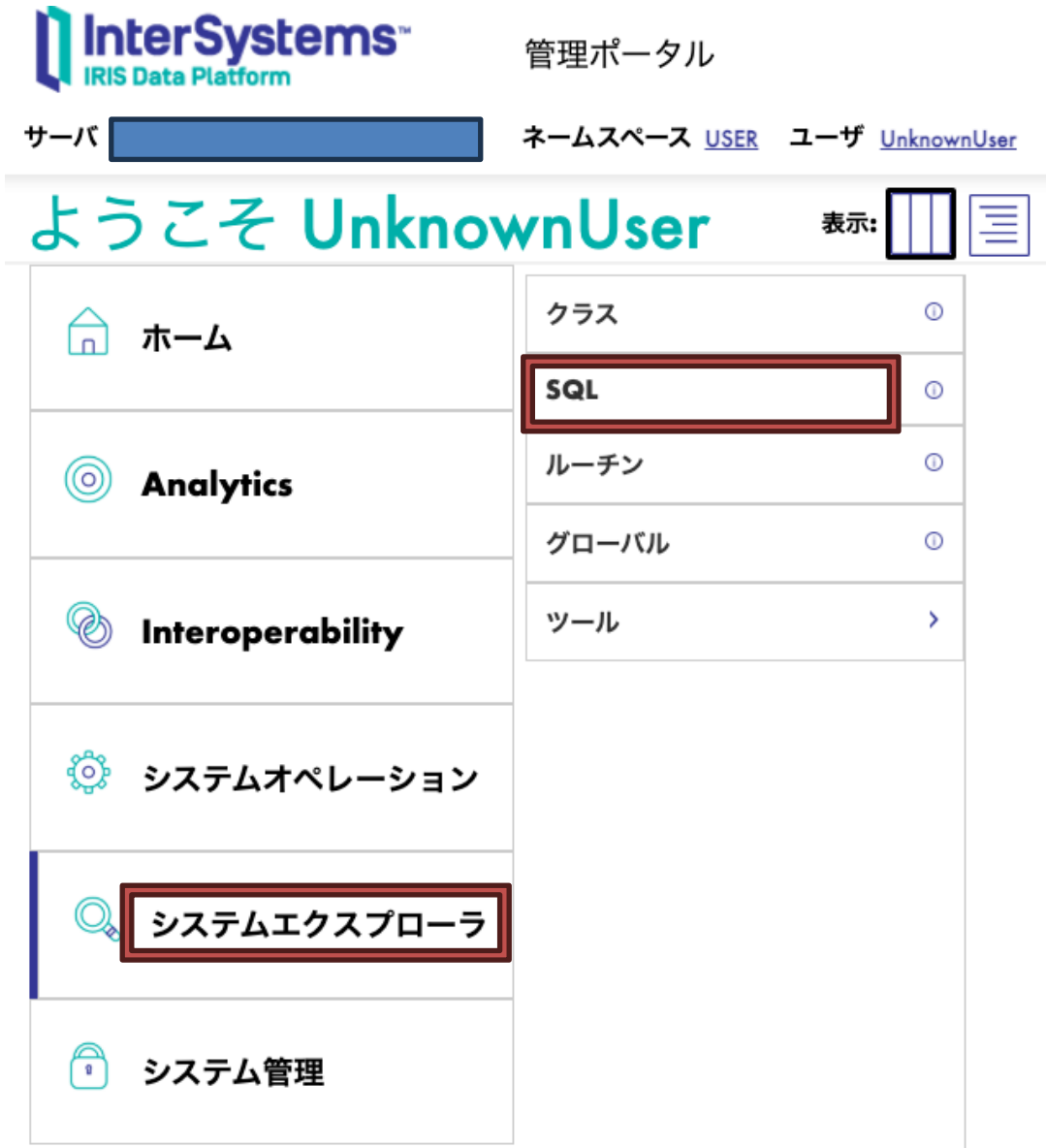


図4 管理ポータル：システムエクスプローラ：SQL

その他機能では、実行したSQLクエリに対してのクエリプランの表示機能もあります。

- クエリプランとは、特定のクエリが実行されたときにそれがデータベース内からどのようにデータを取得するか（プラン）の表示とそれに要する相対コストを概算するものです。
- クエリプランを使用し、あるテーブルに対する特定のクエリを効率よく実行するために、どのようなインデックスを設定すればよいかなど、判断基準となる数値を取得すること

- ・ システムオペレーション (システム運用タスク)

**InterSystems™** IRIS Data Platform **管理ポータル**

サーバ   ネームスペース [USER](#) ユーザ [UnknownUser](#)

ようこそ **UnknownUser** 表示:  

 <b>ホーム</b>	システムダッシュボード 
 <b>Analytics</b>	バックアップ 
 <b>Interoperability</b>	データベース 
 <b>システムオペレーション</b>	プロセス 
 <b>システムエクスプローラ</b>	SQL アクティビティ 
 <b>システム管理</b>	ロック 
	ジャーナル 
	トランザクション 
	シャドウサーバ 
	ミラーモニタ 
	タスクマネージャ 
	<b>LDAP 構成</b> 
	システムログ 
	システム使用 
	ライセンス使用量 
	相互運用性の使用 
	<b>CSPセッション</b> 
	バックグラウンドタスク 
	診断レポート 

追加オプションや詳細を表示するにはここをクリック

図6 管理ポータル：システムオペレーション

システムオペレーションでは、システムの運用管理面で利用するタスク (バックアップ、ジャーナル管理、ロック管理など) を用意しています。

各項目について詳細はドキュメントの「IRIS監視ガイド」、「IRIS高可用性ガイド」、「IRISシステム管理ガイド」をご参照ください。

[https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=GOPS\\_intro/DocBook.UI.Page.cls?KEY=SETSysAdmin](https://docs.intersystems.com/iris20241/csp/docbookj/DocBook.UI.Page.cls?KEY=GOPS_intro/DocBook.UI.Page.cls?KEY=SETSysAdmin)

### 3. 最初のIRISプログラム

ここからは、IRISのプログラミングについて説明していきます。

なお、特に断りのない限りネームスペース“USER”を使用しているものとして説明しています。

IRISはデータベースであると同時に、強力な開発用スクリプト言語を装備しています。

このスクリプト言語をObjectScriptと呼んでいます。

ObjectScriptはオブジェクト指向プログラミングをサポートしており、また IRISデータベースへのSQLアクセス、ネイティブ構造にダイレクトアクセスを行うことができます。

また、文字列処理などのユーティリティ関数を豊富に用意しています。

#### 3.1. Hello world

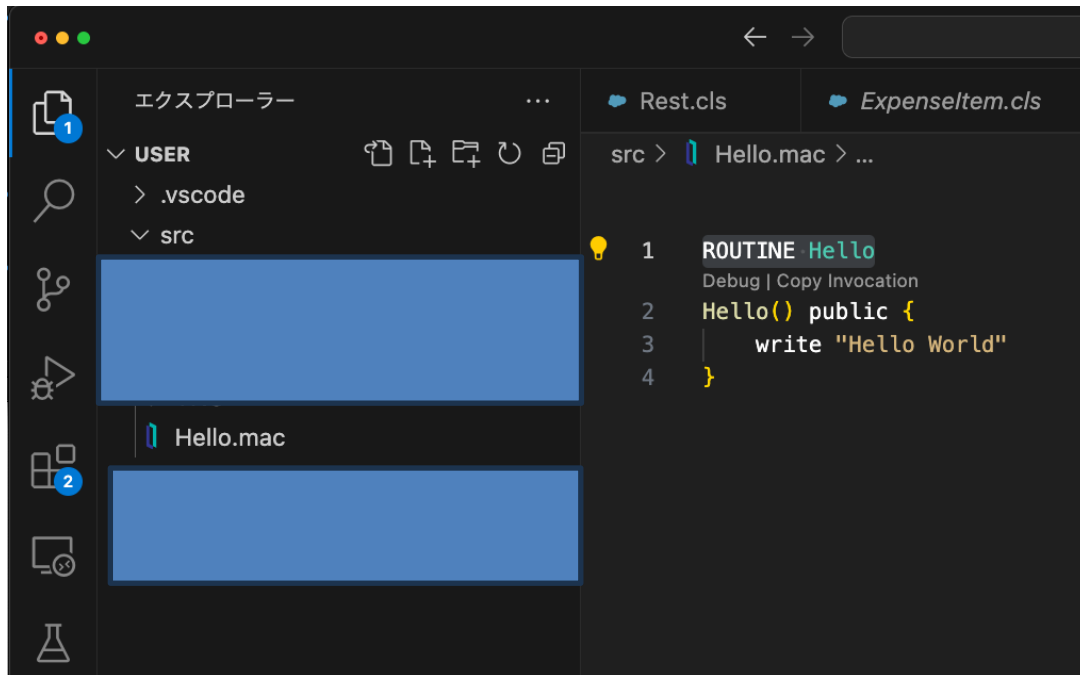
では、最初のIRISのプログラムということで、典型的な例としてHello Worldプログラムを書いてみましょう。

これは、実行すると単に Hello World!と表示するだけのプログラムです。

Visual Studio Codeで、新しくルーチンを作成します。作成するには、Visual Studio Codeのメニューから、「ファイル」→「新規作成」を選択します。



エディタを開いて左側のエクスプローラペインで右クリックし新しいファイルをクリックし、名前をHello.macにします。



次のようなコードを記述してください。

```
ROUTINE Hello
Hello() public {
    write "Hello, world!"
}
```

**図12 Hello world の作成**

ここで“Hello”は、ルーチンのラベル（もしくはエントリ）と呼ばれ、Public宣言により外部から呼び出し可能なことを示しています。

また、Writeは、ObjectScriptのコマンドの一つで端末に対して出力を行うコマンドです。

なおコマンドを書き始める場合には先頭にスペースまたはタブを書く必要があります。

またIRISでは、Writeなどのコマンドは大文字小文字の区別をしません、変数名やラベル名では大文字小文字が区別されますので、注意してください。

さて、このルーチンを実行するわけですが、その前にルーチンを保存・コンパイルする必要があります。

これを行うには、Visual Studio Codeで「ファイル」メニューの「保存」を選択します。

```
12/11/2024 14:24:02 に修飾子 'cuk' でコンパイルを開始しました。  
ルーチンのコンパイル中 : Hello.mac  
コンパイルが正常に終了しました (所要時間: 0.006秒)。
```

上記のように、コンパイル結果が出力されます。

では、いよいよ実行です。

IRISランチャーからターミナルを起動します。

ターミナルが起動すると、プロンプトにネームスペースが表示されます (通常は USER)。そのプロンプトから、

### **Do Hello^Hello()**

と入力し、リターンキーを押下します。

これは、ルーチン HelloのHelloラベル (エントリ) を実行しなさいという IRISのコマンドの形式です。

実行すると、次のような結果が表示されるはずです。

**Hello, world!**

### 3.2. パラメータを渡す

もちろん、ルーチンのエントリに対して引数を定義し、実行時に値を渡すことも出来ます。

Hello()エントリの定義を次のように変更してみましょう。

```
Hello(name) Public {  
    Write "Hello, world, "_name_"!"  
}
```

**図 13 Hello world引数の追加**

ここで、"\_" が文字列の結合演算子として使用されていることに注意してください。そしてコンパイル後、実行します。

例えば、

**Do Hello^Hello("太郎さん")**

と入力すれば、

**Hello, world, 太郎さん!**

と出力されます。

## 4. IRISオブジェクト

これまでで、最初のプログラムHelloを見てきましたが、そこではオブジェクト指向の考え方は使用していませんでした。しかし最初に説明しましたように、IRISに装備された言語ObjectScriptは、完全なオブジェクト指向をサポートしています。

ここからは、オブジェクト指向の考え方をを使ったプログラムの例を見ていきます。

### 4.1. クラスを作成

オブジェクト指向では、クラスがプログラムの単位となります。

詳細はオブジェクト指向の入門書などを参照してください。

では、Visual Studio Codeでクラスを作成してみましょう。

ここでは、「人」を表すPerson クラスを定義してみます。

クラスを定義する際には、そのクラスが持つ属性を定義する必要があります。

IRISではこの属性のことをプロパティと呼んでいます。

では、「人」が持つプロパティは何でしょうか。

もちろんこれは、開発するシステムによって色々なものが考えられますが、ここでは、次のようなプロパティを考えてみます。

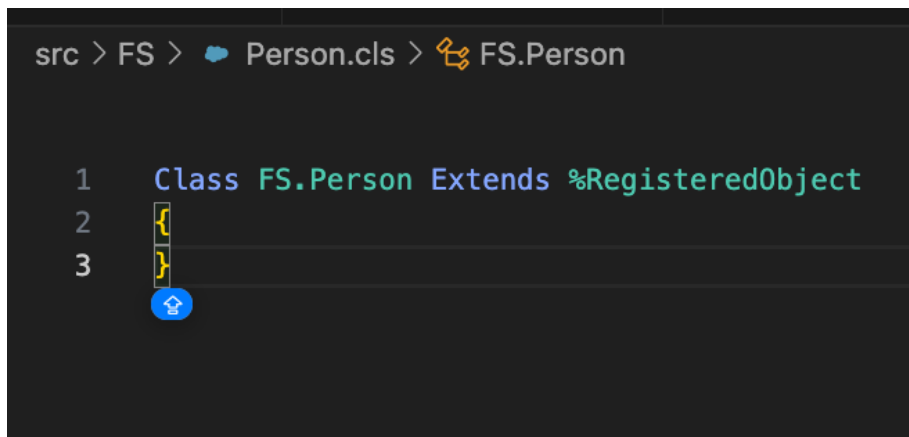
《永続クラス》 Person
Name As %String
Address As %String
Gender As %String
DOB As %Date

Nameは人の名前、Address は住所、Gender は性別、DOB は誕生日 として定義します。

Visual Studio Codeでクラスを定義するには、エクスプローラペインのsrcの所で右クリックし、新しいフォルダーを選び、その名前をFSとします。

次にそのFSのところで右クリックし、新しいファイルをクリックし、ファイル名をPerson.clsにします。

以下のようなコードが生成されると思います。



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows the file structure: src > FS > Person.cls. The main editor area shows the content of Person.cls, which is a class definition: `Class FS.Person Extends %RegisteredObject`. The code is on line 1. Lines 2 and 3 are empty, with line 2 containing a closing curly brace. A blue icon with a plus sign is visible at the bottom left of the editor area.

```
src > FS > Person.cls > FS.Person

1  Class FS.Person Extends %RegisteredObject
2
3
```

それでは、プロパティの定義に移りましょう。

以下のようなコードを追加します。

```
Class FS.Person Extends %RegisteredObject
{
    Property Name As %String;
    Property Address As %String;
    Property Gender As %String;
    Property DOB As %Date;
```

ここで、誕生日 (DOB) プロパティの型が日付を表す %Date型であることに注意してください。

Gender (性別) は、値として "M" (男性) もしくは "F" (女性) の 2 通りを持つプロパティであるとして。

このような場合、IRIS では、値の選択肢をあらかじめ定義しておくことが可能です。

```
Property Gender As %String(VALUELIST = "M,F");
```

```
Class FS.Person Extends %RegisteredObject
{
    Property Name As %String;
    Property Address As %String;
    Property Gender As %String(VALUELIST = ",M,F");
    Property DOB As %Date;
}
```

なお、VALUELISTにセットするとき、最初の文字をデリミタとみなされることに注意してください。

,M,Fとしてもいいですし、例えば #M#Fとしても同じ結果になります。

ですが、最初のデリミタ付け忘れて、M,Fなどとすると、Mをデリミタとみなしてしまいますので注意してください。

そしてデータベースに保存するために、Extendsの後ろを%Persistentに変更します。

```
Class FS.Person Extends %Persistent
{
    Property Name As %String;
    Property Address As %String;
    Property Gender As %String(VALUELIST = ",M,F");
    Property DOB As %Date;
}
```

#### 4.2. コンパイル

それではクラスをコンパイルします[ファイル]→[保存]。

コンパイルが成功すれば、IRIS はこのFS.Personクラスのインスタンスをデータベースに格納するコードを生成します。

たったこれだけで、オブジェクトを処理するデータベースが構築できるのです！

#### 4.3. インスタンスの生成・保存

では早速作成したFS.Personクラスを使用して、そのインスタンスを生成してみましょう。

ObjectScriptは即時実行可能な言語ですので、このようなちょっとした操作も簡単に実行できます。

では、IRISターミナルを開いて次のようなコマンドを実行してください。

```
Set p = ##class(FS.Person).%New()
```

このコマンドでは、FS.Personクラスのインスタンスを生成し、メモリ上の変数pに格納しています。

この時点では、インスタンスはまだデータベースに保存されていないことに注意してください。

また、%New()は新しいインスタンスを生成するクラスメソッドで、クラスメソッドを呼び出すには、

**##class(classname).method()**というシンタックスを使用します。

このインスタンスに対して、各プロパティの値を設定します。

```
Set p.Name = "インターシステムズ 太郎"
```

```
Set p.Address = "東京都新宿区西新宿"
```

```
Set p.Gender = "M"
```

```
Set p.DOB = ##class(%Date).DisplayToLogical("5/1/1978")
```

DOBプロパティ設定時に、日付を**##class(%Date).DisplayToLogical()**によって、文字列からIRISの内部日付フォーマットに変換していることにも注意してください。



最後に、

### **Write p.%Save()**

としてメモリ上のインスタンスをデータベースに保存します。

1 が出力されれば保存が成功したことを示します。

以上でインスタンスが 1 つデータベースに保存されました。

非常に簡単ですね。

#### 4.4. インスタンスの読み込み

本当にインスタンスがデータベースに保存されたか、別のターミナルを開いてそこからインスタンスを読み込むことにより確かめてみましょう。

別のIRISターミナルから次のコマンドを実行します。

```
Set p = ##class(FS.Person).%OpenId(1)
```

ここで%OpenId()は、指定されたOID (Object ID) を持つインスタンスをオープンするメソッドです。

OIDは、インスタンス保存時にIRISにより自動的に付与され、通常は正の整数です。

ですから、クラスの最初のインスタンスは通常OID=1となるので、このコマンドで先程保存されたインスタンスを読み込むことができる訳です。

もちろん、OIDを指定せずにインスタンスを検索する方法もあります。

これについては後のセクションで説明します。

オープンしたインスタンスのプロパティの値を確かめてみます。

**Write p.Name**

インターシステムズ 太郎

**Write p.Address**

東京都新宿区西新宿

先程保存した値が出力されることを確認してください。

#### 4.5. メソッドの定義

オブジェクト指向では、プロパティと共に、クラスを構成するもう一つの要素はメソッドです。

メソッドは、クラスのインスタンスに対して行われる操作を記述するプログラムのことです。

例えば、注文オブジェクトに対して合計を求めたり、明細を追加したりするなどのメソッドが考えられます。

```
Method printGender()
{
    If ..Gender = "M" {
        Write "私は男です"
    }
    Else {
        Write "私は女です"
    }
}
```

図19 メソッドの追加 (printGender())

ここでは、簡単な例として、男性か女性かをターミナルに出力するメソッドを考えてみます。

Personクラス定義の中に、このメソッドを追加します。

メソッドを定義するには、ウィザードを使用することもできますが、ここではコードウインドウを直接編集することにします。

メソッドは、Method宣言によって定義を始めます。

メソッド内で自インスタンスのプロパティや他のメソッドを参照する場合は、**..property** や **..method()** のように **".."** をつけて、自インスタンスであることを示します。

またこのメソッドは、ObjectScriptの if ... else 文の例になっていることにも注意してください。

では、このメソッドを先程作ったインスタンスに対して実行してみます。

実行の前にコンパイルしてください。

**注意**

コンパイルする前に、IRISターミナルでオープンしているインスタンスを全てクローズする必要があります。

そのためには、`Set p=""` と変数を `""` でクリアするか、Kill コマンドを実行し変数を削除します。

再び、IRISターミナルで先程の%OpenId()を使用してインスタンスをオープンします。  
そのインスタンスpに対して、

### **Do p.printGender()**

を実行すると、Genderプロパティの値に応じて端末に出力されるはずです。  
Doコマンドは、このように戻り値がないメソッドの実行にも使用します。

## 4.6. 計算プロパティ

FS.Personクラスに年齢を表すAgeプロパティがないのを不思議に思われたかもしれません。  
が、考えてみると年齢は 1年ごとに増えていきますからデータベースに年齢をもつのは効率的な方法とはいえません。  
とはいえ、年齢は利用者からすると、「人」が持っていて当然の属性であると思うでしょう。

IRISはそのような場合に最適な仕組みを持っています。  
それが「計算プロパティ」です。  
計算プロパティは、実際にデータベースには格納されませんが、参照時に計算してあたかもプロパティが存在するかのように見せる仕組みです。

では、実際に Age プロパティを定義してみましょう。  
Visual Studio Codeのコードウィンドウの任意の行に、次のような定義を加えます。

```
Property Age As %Integer [ Calculated, Readonly];
```

通常のプロパティと同様に定義しますが、最後に “Calculated” という修飾をつけて、計算プロパティであることを宣言しています。  
このように宣言されたプロパティは、実際にデータベースに格納されません。  
また、年齢は参照のみ可能ですので、“Readonly” も付けています。

では、参照時にどのような値を返せばよいのでしょうか。年齢の場合、現在日付と誕生日を比較して計算すれば

良いでしょう。

計算プロパティでは、このような計算ロジックを、ある特定の名前のメソッドで実装します。

その名称は以下のとおりです。

*PropertyGet()*

作成中クラスでは、AgeGet() となります。  
では、AgeGet()の実装を見てみます。

```
Method AgeGet() As %Integer
{
    Quit ($zdate($Horolog,8)-
        $zdate(..DOB),8))¥10000
}
```

ここでいくつかのことについて説明しなければなりません。

まず、+\$Horologという表現式ですが、これは IRISのシステム変数の一つで、現在日付を1841年1月1日  
以来の日数を返します。(省略形では、\$Hと記述できます。)

また、DOBプロパティもこの形式で保存されています。

YYYYMMDD形式で計算した値を10000で割って年齢を求めています('¥' は商を表す演算子です)。

この値を Quitの引数にしています。

Quitは、Cや Javaの returnと同様、メソッドを終了し、戻り値があればそれを渡します。

以上でAgeの定義ができましたので、いつものようにコンパイルして、OID=1のインスタンスをオープンし、次のコマンドを  
実行してみます。

## Write p.Age

ターミナルに年齢が表示されたと思います。



次にこのようにオブジェクトアクセスして生成したデータをSQLで参照する方法について見てみたいと思います。

そのための準備として以下のような処理を行います。（gitの事前インストールが必要です）

操作はc:¥gitで行っているという想定です。

git clone <https://github.com/intersystems-jp/IRISFirstStep.git>

ターミナルを開いて以下のコマンドを順番に実行します。

```
>set file = "c:¥git¥irisfirststep¥src¥FS¥Person.cls"  
>do $system.OBJ.Load(file,"ck")  
>set file = "c:¥git¥irisfirststep¥src¥FS¥Address.cls"  
>do $system.OBJ.Load(file,"ck")  
>do ##class(FS.Person).%KillExtent()  
>do ##class(FS.Person).Populate(1000)
```

## 5. IRIS SQL

これまでは、IRISでクラスを定義し、そのインスタンスを生成・アクセスすることを説明しました。

IRISがオブジェクト指向をサポートし、しかも非常に簡単に使えるデータベースであることがご理解いただけたと思います。

IRISのユニークな機能の一つは、クラスのインスタンスとして作成されたデータが自動的にテーブルとしてもアクセスできることです。

例えば、これまで定義した FS.Personが、同じ名前のテーブルとしてすでに IRIS に定義されています。

ここでは、それを確認してみましょう。

## 5.1. SQLメニュー

IRISの、SQLメニューから、定義されているデータをリレーショナルの観点から操作することができます。

FSスキーマのPersonテーブルのデータをSQLメニューから確認します。(ネームスペースがUSERになっていることを確認してください)



図23 FS.Person テーブルのデータ参照

この例では 1つのスキーマしか存在していませんが、複数のスキーマが存在すれば、それらがすべてリストされます。

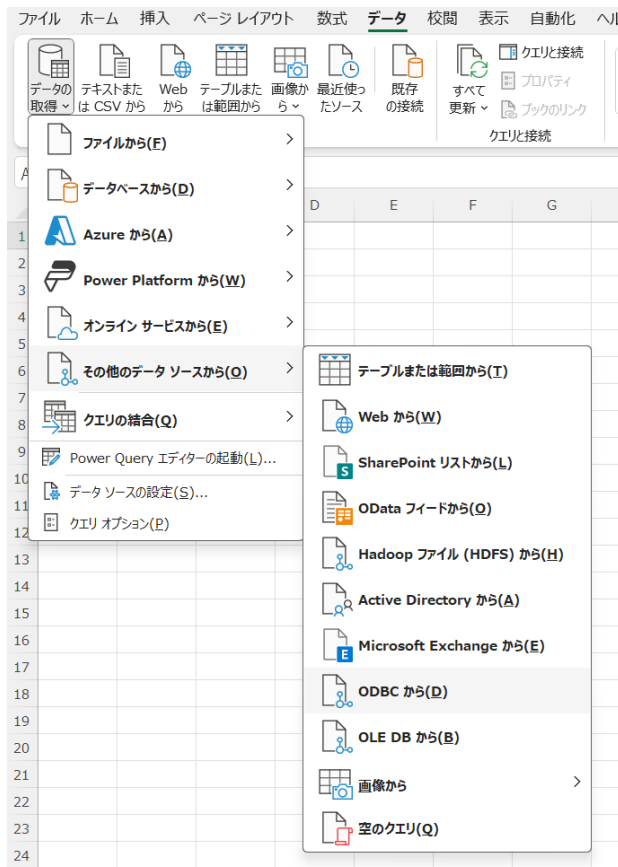
## 5.2. MS-Excelからのアクセス (ODBC)

IRISはODBCからのアクセスをサポートしています。

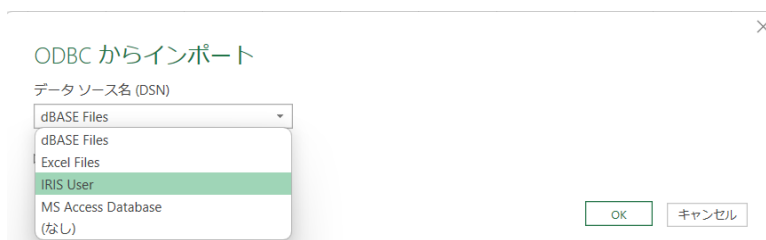
ここでは、ODBCの代表的クライアントツールであるMicrosoft社のExcel(MS-Query)を使用して、ODBC経由でIRISのデータベースにアクセスしてみます。

MS-Excelを起動して、データベースに接続します。

そして、「データの取得」→「その他のデータソースから」→「ODBCから」を選択します。



ODBCからインポートのデータソース名からIRIS Userを選択します。

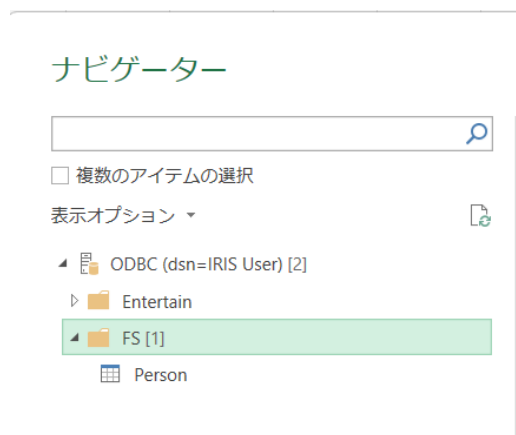


最初にユーザー名とパスワードの入力を求められると思いますが、そこで以下を指定します

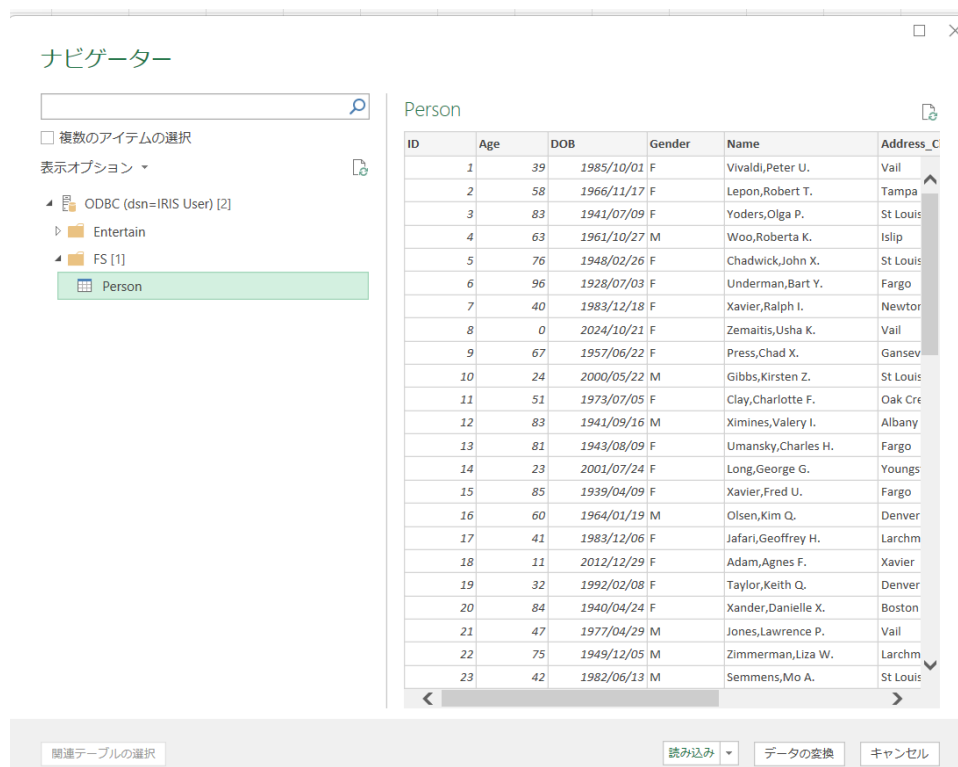
ユーザー名: \_system

パスワード: SYS

次のナビゲーター画面でFSをクリックし、Personをクリックします。



右側にテーブルの内容が表示されます。



取り込みボタンをクリックするとエクセルシートにデータが取り込まれます。

以上、IRISを初めて体験する方を対象に、IRISの持つ基本的機能についてご紹介してきました。

ここまでお読みいただいた方は、すでに IRISを使ったアプリケーションを構築するのに必要な基礎知識を習得されたこととなります。

もちろん、IRISにはここにご紹介し切れなかった機能がたくさんあります。

Javaや.NETとの連携、XML、Webアプリケーションの構築、Webサービス、複雑なオブジェクト連携などです。

これらの機能については、マニュアルや、今後弊社から出される資料をご参照ください。

またFAQサイトも合わせてご参照ください。(<http://faq.intersystems.co.jp/>)

さらに、より詳しいことがお聞きになりたい場合は、インターシステムズジャパン (株) までお問い合わせください。

インターシステムズジャパン (株)

<http://www.intersystems.co.jp/>

TEL: 03-5321-6200 (代表)

Email: [jpinfo@intersystems.com](mailto:jpinfo@intersystems.com)