文章
Hao Ma · 一月 10, 2021 阅读大约需 11 分钟

使用规范优先的方式开发REST API

在本文中，我想谈一谈规范优先的 REST API 开发方式。

传统的代码优先 REST API 开发是这样的：

- 编写代码
- 使其支持 REST
- 形成文档（成为 REST API）

规范优先遵循同样的步骤，不过是反过来的。 我们先制定规范（同时兼做文档），然后根据它生成一个样板 REST 应用，最后编写一些业务逻辑。

这是有好处的，因为：

- 对于想要使用你的 REST API 的外部或前端开发者，你总是有相关且有用的文档
- 使用 OAS (Swagger) 创建的规范可以导入各种工具，从而进行编辑、客户端生成、API 管理、单元测试和自动化，或者许多其他任务的简化
- 改进了 API 架构。 在代码优先的方式中，API 是逐个方法开发的，因此开发者很容易失去对整体 API 架构的跟踪，但在规范优先的方式中，开发者被强制从 API 使用者的角度与 API 进行交互，这通常有助于设计出更简洁的 API 架构
- 更快的开发速度 - 由于所有样板代码都是自动生成的，你无需编写代码，只需开发业务逻辑。
- 更快的反馈循环 - 使用者可以立即查看 API，并且只需修改规范即可轻松提供建议
  让我们以规范优先的方式开发 API 吧！

计划

1. 使用 swagger 制定规范
    - Docker
    - 本地
    - 在线
2. 将规范加载到 IRIS 中
    - API 管理 REST API
    - ^REST
    - 类
3. 我们的规范会怎样？
4. 实现
5. 进一步开发
6. 注意事项
    - 特殊参数
    - CORS
7. 将规范加载到 IAM 中

制定规范

勿庸置疑，第一步是编写规范。 InterSystems IRIS 支持 Open API 规范 (OAS)：

> **OpenAPI 规范** （以前称为 Swagger 规范）是 REST API 的 API 描述格式。 OpenAPI 文件允许描述整个

API，包括：

- 可用端点 (/users) 和每个端点上的操作（GET /users、POST /users）
- 每次操作的操作参数输入和输出
- 身份验证方法
- 联系信息、许可证、使用条款和其他信息。

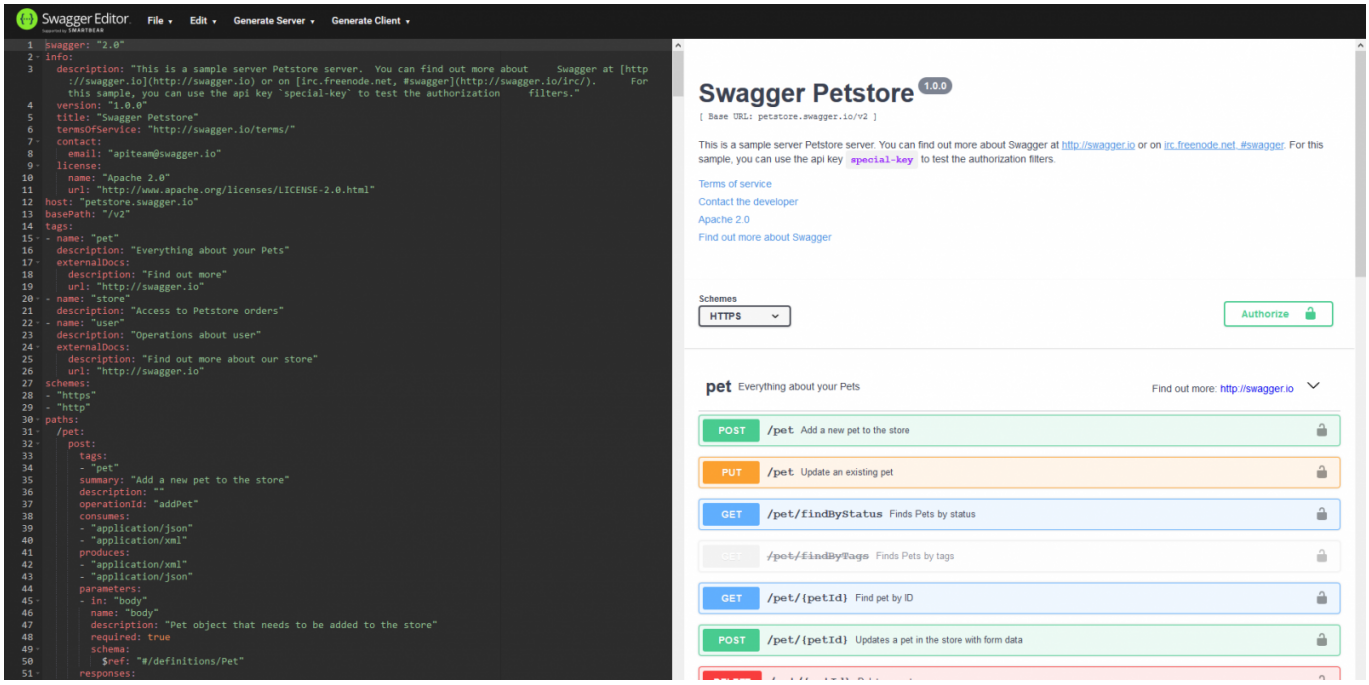
API 规范可以使用 YAML 或 JSON 编写。 格式易于学习，并且对人和机器都可读。 完整的 OpenAPI 规范可在 GitHub 上找到：OpenAPI 3.0 规范

- 来自 Swagger 文档。

我们将使用 Swagger 编写 API。 使用 Swagger 有几种方法：

- 在线
- Docker：docker run -d -p 8080:8080 swaggerapi/swagger-editor
- 本地安装

安装/运行 Swagger 后，你应该在 Web 浏览器中看到以下窗口：



在左侧编辑 API 规范，在右侧可以立即看到渲染的 API 文档/测试工具。

我们将第一个 API 规范加载到其中（使用 YAML）。 这是一个简单的 API，包含一个 GET 请求 - 返回指定范围内的随机数。

Math API 规范

以下是其包含的内容。

有关 API 和使用的 OAS 版本的基本信息。

```
swagger: "2.0"
info:
  description: "Math"
  version: "1.0.0"
  title: "Math REST API"
```

服务器主机、协议（http、https）和 Web 应用程序名称：

```
host: "localhost:52773"
basePath: "/math"
schemes:
  - http
```

接下来指定路径（完整的 URL 是 http://localhost:52773/math/random/:min/:max）和 HTTP 请求方法（get、post、put、delete）：

```
paths:
  /random/{min}/{max}:
    get:
```

之后，指定有关请求的信息：

```
      x-ISC_CORS: true
      summary: "Get random integer"
      description: "Get random integer between min and max"
      operationId: "getRandom"
      produces:

      - "application/json"
      parameters:

      - name: "min"
        in: "path"
        description: "Minimal Integer"
        required: true
        type: "integer"
        format: "int32"

      - name: "max"
        in: "path"
        description: "Maximal Integer"
        required: true
        type: "integer"
        format: "int32"
      responses:
        200:
          description: "OK"
```

在此部分中，我们定义请求：

- 为 CORS 启用此路径（稍后将详细介绍）

- 提供 *summary* 和 *description*
- *operationId* 允许规范内引用，它也是我们的实现类中生成的方法名
- *produces* - 响应格式（例如文本、xml、json）
- *parameters* 指定输入参数（在 URL 或正文中），在我们的示例中，我们指定 2 个参数 - 随机数生成器的范围
- *responses* 列出服务器的可能响应

如你所见，这种格式并不是特别有挑战性，虽然还有很多可用功能。这里是规范。

最后，我们将定义导出为 JSON。 转到 “ 文件→ 转换 ” 并另存为 JSON。 规范应如下所示：

Math API 规范

将规范加载到 IRIS 中

现在我们有了规范，我们可以在 InterSystems IRIS 中为此 REST API 生成样板代码。

要进入此阶段，我们需要三个东西：

- REST 应用程序名称：我们生成的代码的包（假定为 math</ 0>）
JSON 格式的 OAS 规范：我们刚刚在上一步中创建
Web 应用程序名称：用于访问我们的 REST API 的基本路径（我们的示例中为 <code>/math）

有三种方法使用我们的规范来生成代码，它们本质上是相同的，只是提供了多种访问相同功能的方式

1. 调用 ^%REST 例程（在交互式终端会话中 Do ^%REST）， 参见文档。
2. 调用 %REST 类（Set sc = ##class(%REST.API).CreateApplication(applicationName, spec)，非交互式），参见文档。
3. 使用 API 管理 REST API，参见文档。

我认为文档足以描述所需的步骤，因此选择一个即可。 我再补充两点说明：

- 在第 1 种和第 2 种方法中，可以向动态对象传递文件名或 URL
- 在第 2 种和第 3 种方法中，**必须** 进行一个额外的调用才能创建 Web 应用程序：set sc = ##class(%SYS.REST).DeployApplication(restApp, webApp, authenticationType)，所以在我们的示例中为 set sc = ##class(%SYS.REST).DeployApplication("math", "/math")，从 %sySecurity 包含文件获取 authenticationType 参数的值，相关条目为 $$$Authe*，因此对于未经身份验证的访问，传递 $$$AutheUnauthenticated。 如果省略，该参数默认为密码身份验证。

我们的规范会怎样？

如果你已成功创建应用，新的 math 包应该包含三个类：

- *Spec* - 按原样存储规范。
- *Disp* - 在调用 REST 服务时直接调用。 它封装 REST 处理并调用实现方法。
- *Impl* - 保存 REST 服务的实际内部实现。 你只应该编辑此类。

文档包含有关这些类的更多信息。

实现

最初，我们的实现类 math.impl 只包含一个方法，对应于我们的 /random/{min}/{max} 操作：

```
/// Get random integer between min and max<br/>
/// The method arguments hold values for:<br/>
///     min, Minimal Integer<br/>
///     max, Maximal Integer<br/>
ClassMethod getRandom(min As %Integer, max As %Integer) As %DynamicObject
{
    //(Place business logic here)
    //Do ..%SetStatusCode(<HTTP_status_code>)
    //Do ..%SetHeader(<name>,<value>)
    //Quit (Place response here) ; response may be a string, stream or dynamic object
}
```

让我们从简单的实现开始：

```
ClassMethod getRandom(min As %Integer, max As %Integer) As %DynamicObject
{
    quit {"value":($random(max-min)+min)}
}
```

最后，我们可以通过在浏览器中打开此页面来调用我们的 REST API：http://localhost:52773/math/random/1/100

输出应该是：

```
{
    "value": 45
}
```

此外，在 Swagger 编辑器中按 Try it out（试用）按钮并填写请求参数也会发送同样的请求：

恭喜！ 我们使用规范优先的方式创建的第一个 REST API 现在已经生效！

进一步开发

当然，我们的 API 不是静态的，我们需要添加新路径等等。 在规范优先的开发中，首先要修改规范，然后更新 REST 应用程序（调用与创建应用程序相同），最后编写代码。
请注意，规范更新是安全的：你的代码不会受到影响，即使从规范中删除路径，在实现类中也不会删除方法。

注意事项

更多说明！

特殊参数

InterSystems 向 swagger 规范添加了特殊参数，如下所示：

名称	数据类型	默认值	位置	描述
x-ISCDispatchParent	类名	%CSP.REST	信息	调度类的超类。
x-ISCCORS	布尔	false	操作	一个标志，指示对此端点/方法组合的

x-ISCRequiredResource	数组	操作	CORS 请求应该获得支持。 以逗号分隔的已定义资源及其访问模式（资源:模式）的列表，这些资源和模式是访问 REST 服务的此端点所必需的。 示例：["%Development:USE"]
x-ISCServiceMethod	字符串	操作	后端调用的用于维护此操作的类方法的名称；默认值为 operationId，通常就很合适。

CORS

有三种方法启用 CORS 支持。

1. 在逐条路由的基础上，将 x-ISCCORS 指定为 true。 我们的 Math REST API 中就是这样做的。
2. 在每个 API 的基础上，添加

   Parameter HandleCorsRequest = 1;

然后重新编译该类。 规范更新后它也会保留。

3. （推荐）在每个 API 的基础上，实现自定义调度器超类（应该扩展 %CSP.REST）并编写 CORS 处理逻辑。 要使用此超类，请将 x-ISCDispatchParent 添加到规范中。

将规范加载到 IAM 中

最后，我们将规范添加到 IAM中，以便将其发布给其他开发者。

如果您尚未开始使用 IAM，请参见此文章。 它还涵盖了通过 IAM 提供 REST API，所以我在这里不做介绍。
您可能需要修改规范的 host 和 basepath 参数，使它们指向 IAM，而不是 InterSystems IRIS 实例。

打开 IAM 管理员门户，转到相关工作区的 Specs（规范）选项卡。

点击 Add Spec（添加规范）按钮并输入新 API 的名称（我们的示例中为 math）。 在 IAM 中创建新规范后，点击 Edit（编辑）并粘贴规范代码（JSON 或 YAML - IAM 都支持）：

不要忘记点击 Update File（更新文件）。

现在我们的 API 已发布给开发者。 打开开发者门户，然后点击右上角的 Documentation（文档）。 除了三个默认 API，还应该看到我们的新 Math REST API：

打开它：

现在，开发者可以看到我们的新 API 的文档，并在同一个地方试用它！

结论

InterSystems IRIS 简化了 REST API 的开发过程，规范优先的方式使 REST API 生命周期管理更快更简单。
通过这种方式，你可以使用各种工具来完成各种相关任务，例如客户端生成、单元测试、API 管理等等。

链接

- OpenAPI 3.0 规范
- 创建 REST 服务
- 从 IAM 开始
- IAM 文档

#API #InterSystems API管理器（IAM） #REST API #InterSystems IRIS

源
URL:
https://cn.community.intersystems.com/post/%E4%BD%BF%E7%94%A8%E8%A7%84%E8%8C%83%E4%BC%98%E5%85%88%E7%9A%84%E6%96%B9%E5%BC%8F%E5%BC%80%E5%8F%91rest-api