



Solution 1: Create a Contact REST API

1. Define the API schema.

- a) Create a new package.
 - Right-click on **integration** package and click **New ➔ Package**.
 - Enter **apis.si.ta** as the new package name.
- b) Create a new API schema file.
 - Right-click on **apis.si.ta** package and click **New ➔ File**.
 - Enter **contact-1.0.swagger.yaml** as the new file name.
- c) Add schema header information and new paths.

```
swagger: '2.0'
info:
  version: '1.0'
  title: "Contact API"
  description: "Contact API"
basePath: /si/ta/contact/v1
x-gw-schema-import:
  contact: si.ta.contact-1.0
x-gw-apihandlers:
- si.ta.restapi.ContactAPIHandler
produces:
- application/json
consumes:
- application/json
paths:
  /contacts/{contactId}:
    get:
      summary: "Retrieves a contact details"
      operationId: getContactDetailInformation
      parameters:
        - $ref: "#/parameters/contactId"
      responses:
        '200':
          description: "Contact details returned"
          schema:
            $ref: "contact#/definitions/ContactDetails"
parameters:
  contactId:
    name: contactId
    in: path
    type: string
    required: true
```

2. Define the API handler class.

- a) Create a new package.
 - Right-click on **gsrc** package and click **New ➔ Package**.

- Enter **si.ta.restapi** as the new package name.
- b) Create a new Gosu class.
- Right-click on **restapi** package and click **New ➔ Gosu Class**.
 - Enter **ContactAPIHandler** as the new Gosu class name.
- c) Configure the API method.

```
package si.ta.restapi

uses gw.api.database.Query
uses gw.api.database.Relop
uses gw.api.json.JsonConfigAccess
uses gw.api.json.mapping.JsonMappingOptions
uses gw.api.json.mapping.TransformResult

class ContactAPIHandler {

    function getContactDetailInformation(contactId : String) : TransformResult {
        // Query for contact
        var user = findContactById(contactId)
        // Create JsonMapper object
        var jsonMapper = JsonConfigAccess.getMapper("si.ta.contact-1.0", "ContactDetails")
        var mappingOpts = new JsonMappingOptions().withFilter("si.ta.contact_api-1.0")
        return jsonMapper.transformObject(user, mappingOpts)
    }

    private function findContactById(id : String) : ABContact {
        // Query for contact
        var queryObj = Query.make(ABContact)
        queryObj.compare(ABContact#PublicID, Relop.Equals, id)
        var targetObj = queryObj.select().AtMostOneRow
        return targetObj
    }
}
```

3. Publish the API.

- a) Update the **published-apis.yaml** file with the fully qualified name of the new API.



```
published-apis.yaml
apis:
  - name: gw.pl.framework.api_list-1.0
  - name: trn.ta.contact-1.0
  - name: si.ta.contact-1.0
defaultTemplate:
  - name: gw.pl.framework.dev_template-1.0
```

4. Deploy code changes.

- a) From the Studio menu, **Restart the server**.

5. Perform verification steps.