swagger: "2.0"

info:

description: "This is a sample server car Accident Center."

version: "1.0.0"

title: "Swagger Accident Car Center"

termsOfService: "http://example.com/terms/"

contact:

email: "tomasz.krajewski@opitz-consulting.com"

license:

name: "Apache 2.0"

url: "http://www.apache.org/licenses/LICENSE-2.0.html"

host: "example.com"

basePath: "/v2"

tags:

- name: "car accident"

description: "Reporting data while a car accident"

externalDocs:

description: "Find out more"

url: "http://swagger.io"

- name: "accident server"

description: "Access to saving data into the server"

- name: "car"

description: "Operations about car"

schemes:

- "http"

paths:

/accident:

post:

tags:

- "accident"

summary: "Add a new accident to emergency center"

description: ""

operationId: "addAccident"

consumes:

- "application/json"

- "application/xml"

produces:

- "application/xml"

- "application/json"

parameters:

- in: "body"

name: "body"

description: "Accident object that needs to be added to the emergency center"

required: true

schema:

$ref: "#/definitions/Accident"

responses:

405:

description: "Invalid input"

security:

- accidentstore\_auth:

- "write:accidents"

- "read:accidents"

put:

tags:

- "accident"

summary: "Update an existing accident"

description: ""

operationId: "updateAccident"

consumes:

- "application/json"

- "application/xml"

produces:

- "application/xml"

- "application/json"

parameters:

- in: "body"

name: "body"

description: "Accident object that needs be added to the emergency center"

required: true

schema:

$ref: "#/definitions/Accident"

responses:

400:

description: "Invalid ID supplied"

404:

description: "Accident not found"

405:

description: "Validation exception"

security:

- accidentstore\_auth:

- "write:accidents"

- "read:accidents"

/accident/findByStatus:

get:

tags:

- "accident"

summary: "Finds Accidents by status"

description: "Multiple status values can be provided with comma separated strings"

operationId: "findAccidentsByStatus"

produces:

- "application/xml"

- "application/json"

parameters:

- name: "status"

in: "query"

description: "Status values that need to be considered for filter"

required: true

type: "array"

items:

type: "string"

enum:

- "registred"

- "service is pending"

- "resolved"

default: "registred"

collectionFormat: "multi"

responses:

200:

description: "successful operation"

schema:

type: "array"

items:

$ref: "#/definitions/Accident"

400:

description: "Invalid status value"

security:

- accidentstore\_auth:

- "write:accidents"

- "read:accidents"

/accident/findByInjury:

get:

tags:

- "accident"

summary: "Finds Accidents by injury"

description: "Multiple status values can be provided with comma separated strings"

operationId: "findAccidentsByInjury"

produces:

- "application/xml"

- "application/json"

parameters:

- name: "injury"

in: "query"

description: "injury values that need to be considered for filter"

required: true

type: "array"

items:

type: "string"

enum:

- "Undetermined"

- "Good"

- "Fair"

- "Serious"

- "Critical"

- "Dead"

default: "Undetermined"

collectionFormat: "multi"

responses:

200:

description: "successful operation"

schema:

type: "array"

items:

$ref: "#/definitions/Accident"

400:

description: "Invalid status value"

security:

- accidentstore\_auth:

- "write:accidents"

- "read:accidents"

/accident/findByTags:

get:

tags:

- "accident"

summary: "Finds Accidents by tags"

description: "light, death, no\_victims, victims, fire\_brigade, police, no\_police"

operationId: "findAccidentsByTags"

produces:

- "application/xml"

- "application/json"

parameters:

- name: "tags"

in: "query"

description: "Tags to filter by"

required: true

type: "array"

items:

type: "string"

collectionFormat: "multi"

responses:

200:

description: "successful operation"

schema:

type: "array"

items:

$ref: "#/definitions/Accident"

400:

description: "Invalid tag value"

security:

- accidentstore\_auth:

- "write:accidents"

- "read:accidents"

deprecated: true

/accident/{accidentId}:

get:

tags:

- "accident"

summary: "Find accident by ID"

description: "Returns a single accident"

operationId: "getAccidentById"

produces:

- "application/xml"

- "application/json"

parameters:

- name: "accidentId"

in: "path"

description: "ID of accident to return"

required: true

type: "integer"

format: "int64"

responses:

200:

description: "successful operation"

schema:

$ref: "#/definitions/Accident"

400:

description: "Invalid ID supplied"

404:

description: "Accident not found"

security:

- api\_key: []

post:

tags:

- "accident"

summary: "Updates an accident in the emergency center with form data"

description: ""

operationId: "updateAccidentWithForm"

consumes:

- "application/x-www-form-urlencoded"

produces:

- "application/xml"

- "application/json"

parameters:

- name: "accidentId"

in: "path"

description: "ID of accident that needs to be updated"

required: true

type: "integer"

format: "int64"

- name: "CarModel"

in: "formData"

description: "Updated car model of the accident"

required: true

type: "string"

- name: "carNumberPlates"

in: "formData"

description: "Updated car number plates"

required: true

type: "string"

- name: "accidentTime"

in: "formData"

description: "Updated time of accident"

required: true

type: "string"

- name: "status"

in: "formData"

description: "Updated status of the accident"

required: true

type: "injury"

- name: "status"

in: "formData"

description: "Updated injuries of the accident"

required: true

type: "string"

- name: "owner"

in: "formData"

description: "Updated status of the owner of the car"

required: true

type: "string"

- name: "location"

in: "formData"

description: "Updated status of GPS car location"

required: true

type: "string"

- name: "averageSpeed"

in: "formData"

description: "Updated status avarage speed from last 5 minutes before accident"

required: true

type: "number"

- name: "passangers"

in: "formData"

description: "Updated status of passangers"

required: true

type: "number"

responses:

405:

description: "Invalid input"

security:

- accidentstore\_auth:

- "write:accidents"

- "read:accidents"

delete:

tags:

- "accident"

summary: "Deletes a accident"

description: ""

operationId: "deleteAccident"

produces:

- "application/xml"

- "application/json"

parameters:

- name: "api\_key"

in: "header"

required: false

type: "string"

- name: "accidentId"

in: "path"

description: "Accident id to delete"

required: true

type: "integer"

format: "int64"

responses:

400:

description: "Invalid ID supplied"

404:

description: "Accident not found"

security:

- accidentstore\_auth:

- "write:accidents"

- "read:accidents"

/accidents/all:

get:

tags:

- "accidents"

summary: "Returns all accidents by status"

description: "Returns all accidents by status"

operationId: "getAllAccidents"

produces:

- "application/json"

parameters: []

responses:

200:

description: "successful operation"

schema:

type: "object"

additionalProperties:

type: "integer"

format: "int32"

security:

- api\_key: []

securityDefinitions:

accidentstore\_auth:

type: "oauth2"

authorizationUrl: "http://accidentstore.swagger.io/oauth/dialog"

flow: "implicit"

scopes:

write:accidents: "modify accidents in your account"

read:accidents: "read your accidents"

api\_key:

type: "apiKey"

name: "api\_key"

in: "header"

definitions:

Tag:

type: "object"

properties:

id:

type: "integer"

format: "int64"

name:

type: "string"

xml:

name: "Tag"

Accident:

type: "object"

required:

- "CarModel"

properties:

accidentId:

type: "integer"

format: "int64"

CarModel:

type: "string"

example: "Toyota Yaris II 2014"

carNumberPlates:

type: "string"

example: "D65261"

accidentTime:

type: "string"

format: "date-time"

example: "1985-04-12T23:20:50.52Z"

owner:

type: "string"

example: "Wojtek Konowal"

location:

type: "array"

example: "[47.4925, 19.0513]"

averageSpeed:

type: "number"

format: "double"

passangers:

type: "number"

format: "int32"

tags:

type: "array"

xml:

name: "tag"

wrapped: true

items:

$ref: "#/definitions/Tag"

status:

type: "string"

description: "accident status for emergency center"

enum:

- "registred"

- "service is pending"

- "resolved"

xml:

name: "Accident"