Swagger Petstore

This is a sample server Petstore server. You can find out more about Swagger at http://swagger.io or on irc.freenode.net, #swagger. For this sample, you can use the api key "special-key" to test the authorization filters

Table of contents

1 Paths

1.1 POST /pet

Add a new pet to the store

Request body

application/json application/xml

Pet object that needs to be added to the store

Request-model:

```
{
    "$ref": "#/definitions/Pet"
}
```

Responses

application/json application/xml

HTTP Status Code	Reason
405	Invalid input

1.2 PUT /pet

Update an existing pet

Request body

application/json application/xml

Pet object that needs to be added to the store

Request-model:

```
{
    "$ref": "#/definitions/Pet"
}
```

Responses

application/json application/xml

HTTP Status Code	Reason
400	Invalid ID supplied
404	Pet not found
405	Validation exception

1.3 GET /pet/findByStatus

Finds Pets by status

Multiple status values can be provided with comma seperated strings

Parameters

Name	Description	Parameter Type	Data Type
status	Status values that need to be considered for filter	query	string[], Format: multi

Responses

application/json application/xml

HTTP Status Code	Reason

HTTP Status Code	Reason

200	successful operation
	Response Model:
	<pre>{
	  "items": {
	  "$ref": "#/definitions/Pet"
	  },
	  "type": "array"
	}</pre>

400	Invalid status value

1.4 GET /pet/findByTags

Finds Pets by tags

Muliple tags can be provided with comma seperated strings. Use tag1, tag2, tag3 for testing.

Parameters

Name	Description	Parameter Type	Data Type
tags	Tags to filter by	query	string[], Format: multi

Responses

application/json application/xml

HTTP Status Code	Reason

200	successful operation
	Response Model:
	<pre>{
	  "items": {
	  "$ref": "#/definitions/Pet"
	  },
	  "type": "array"
	}</pre>

400	Invalid tag value

1.5 GET /pet/{petId}

Find pet by ID

Returns a pet when ID < 10. ID > 10 or nonintegers will simulate API error conditions

Parameters

Name	Description	Parameter Type	Data Type
petId	ID of pet that needs to be fetched	path	integer

Responses

application/json application/xml

HTTP Status Code	Reason
200	successful operation **Response Model:** `{ "$ref": "#/definitions/Pet" }`
400	Invalid ID supplied
404	Pet not found

1.6 POST /pet/{petId}

Updates a pet in the store with form data

Request body

application/x-www-form-urlencoded

Parameters

Name	Description	Parameter Type	Data Type
petId	ID of pet that needs to be updated	path	string
name	Updated name of the pet	formData	string
status	Updated status of the pet	formData	string

Responses

application/json application/xml

HTTP Status Code	Reason
405	Invalid input

1.7 DELETE /pet/{petId}

Deletes a pet

Parameters

Name	Description	Parameter Type	Data Type
api_key		header	string
petId	Pet id to delete	path	integer

Responses

application/json application/xml

HTTP Status Code	Reason
400	Invalid pet value

1.8 POST /pet/{petId}/uploadImage

uploads an image

Request body

multipart/form-data

Parameters

Name	Description	Parameter Type	Data Type
petId	ID of pet to update	path	integer
additionalMetadata	Additional data to pass to server	formData	string
file	file to upload	formData	file

Responses

application/json

HTTP Status Code	Reason
200	successful operation **Response Model:** `{ "$ref": "#/definitions/ApiResponse" }`

1.9 GET /store/inventory

Returns pet inventories by status

Returns a map of status codes to quantities

Responses

application/json application/xml

HTTP Status Code	Reason

200	successful operation
	Response Model:
	{
	"additionalProperties": {
	"format": "int32",
	"type": "integer"
	},
	"type": "object"
	}

1.10 POST /store/order

Place an order for a pet

Responses

application/json application/xml

HTTP Status Code	Reason

200	successful operation
	Response Model:
	{
	"$ref": "#/definitions/Order"
	}

400	Invalid Order

1.11 GET /store/order/{orderId}

Find purchase order by ID

For valid response try integer IDs with value <= 5 or > 10. Other values will generated exceptions

Parameters

Name	Description	Parameter Type	Data Type
orderId	ID of pet that needs to be fetched	path	string

Responses

application/json application/xml

HTTP Status Code	Reason
200	successful operation **Response Model:** `{ "$ref": "#/definitions/Order" }`
400	Invalid ID supplied
404	Order not found

1.12 DELETE /store/order/{orderId}

Delete purchase order by ID

For valid response try integer IDs with value < 1000. Anything above 1000 or nonintegers will generate API errors

Parameters

Name	Description	Parameter Type	Data Type
orderId	ID of the order that needs to be deleted	path	string

Responses

application/json application/xml

HTTP Status Code	Reason
400	Invalid ID supplied
404	Order not found

1.13 POST /user

Create user

This can only be done by the logged in user.

Responses

application/json application/xml

HTTP Status Code	Reason
default	successful operation

1.14 POST /user/createWithArray

Creates list of users with given input array

Responses

application/json application/xml

HTTP Status Code	Reason
default	successful operation

1.15 POST /user/createWithList

Creates list of users with given input array

Responses

application/json application/xml

HTTP Status Code	Reason
default	successful operation

1.16 GET /user/login

Logs user into the system

Parameters

Name	Description	Parameter Type	Data Type
username	The user name for login	query	string
password	The password for login in clear text	query	string

Responses

application/json application/xml

HTTP Status Code	Reason
200	successful operation **Response Model:** `{ "type": "string" }`
400	Invalid username/password supplied

1.17 GET /user/logout

Logs out current logged in user session

Responses

application/json application/xml

HTTP Status Code	Reason
default	successful operation

1.18 GET /user/{username}

Get user by user name

Parameters

Name	Description	Parameter Type	Data Type
username	The name that needs to be fetched. Use user1 for testing.	path	string

Responses

application/json application/xml

HTTP Status Code	Reason
200	successful operation **Response Model:** `{ "$ref": "#/definitions/User" }`

HTTP Status Code	Reason
400	Invalid username supplied
404	User not found

1.19 PUT /user/{username}

Updated user

This can only be done by the logged in user.

Parameters

Name	Description	Parameter Type	Data Type
username	name that need to be deleted	path	string

Responses

application/json application/xml

HTTP Status Code	Reason
400	Invalid user supplied
404	User not found

1.20 DELETE /user/{username}

Delete user

This can only be done by the logged in user.

Parameters

Name	Description	Parameter Type	Data Type
username	The name that needs to be deleted	path	string

Responses

application/json application/xml

HTTP Status Code	Reason
400	Invalid username supplied
404	User not found

2 Definitions

User


```
{
    "properties": {
        "email": {
            "type": "string"
        },
        "firstName": {
            "type": "string"
        },
        "id": {
            "format": "int64",
            "type": "integer"
        },
        "lastName": {
            "type": "string"
        },
        "password": {
            "type": "string"
        },
        "phone": {
            "type": "string"
        },
        "userStatus": {
            "description": "User Status",
            "format": "int32",
            "type": "integer"
        },
        "username": {
            "type": "string"
        }
    },
    "xml": {
        "name": "User"
    }
}
```


ApiResponse

```
{
    "properties": {
        "code": {
            "format": "int32",
            "type": "integer"
        },
        "message": {
            "type": "string"
        },
        "type": {
            "type": "string"
        }
    },
    "xml": {
        "name": "##default"
    }
}
```


Category


```
{
    "properties": {
        "id": {
            "format": "int64",
            "type": "integer"
        },
        "name": {
            "type": "string"
        }
    },
    "xml": {
        "name": "Category"
    }
}
```


Pet

```
{
    "properties": {
        "category": {
            "$ref": "#/definitions/Category"
        },
        "id": {
            "format": "int64",
            "type": "integer"
        },
        "name": {
            "example": "doggie",
            "type": "string"
        },
        "photoUrls": {
            "items": {
                "type": "string"
            },
            "type": "array",
            "xml": {
                "name": "photoUrl",
                "wrapped": true
            }
        },
        "status": {
            "description": "pet status in the store",
            "enum": [
                "available",
                "pending",
                "sold"
            ],
            "type": "string"
        },
        "tags": {
            "items": {
                "$ref": "#/definitions/Tag"
            },
            "type": "array",
            "xml": {
                "name": "tag",
                "wrapped": true
            }
        }
    },
    "required": [
        "name",
        "photoUrls"
    ],
    "xml": {
        "name": "Pet"
    }
}
```


Tag

```
{
    "properties": {
        "id": {
            "format": "int64",
            "type": "integer"
        },
        "name": {
            "type": "string"
        }
    },
    "xml": {
        "name": "Tag"
    }
}
```


Order


```
{
    "properties": {
        "complete": {
            "type": "boolean"
        },
        "id": {
            "format": "int64",
            "type": "integer"
        },
        "petId": {
            "format": "int64",
            "type": "integer"
        },
        "quantity": {
            "format": "int32",
            "type": "integer"
        },
        "shipDate": {
            "format": "date-time",
            "type": "string"
        },
        "status": {
            "description": "Order Status",
            "enum": [
                "placed",
                "approved",
                "delivered"
            ],
            "type": "string"
        }
    },
    "xml": {
        "name": "Order"
    }
}
```



```
{
```

```
    "basePath": "/v2",
    "definitions": {
        "ApiResponse": {
            "properties": {
                "code": {
                    "format": "int32",
                    "type": "integer"
                },
                "message": {
                    "type": "string"
                },
                "type": {
                    "type": "string"
                }
            },
            "xml": {
                "name": "##default"
            }
        },
        "Category": {
            "properties": {
                "id": {
                    "format": "int64",
                    "type": "integer"
                },
                "name": {
                    "type": "string"
                }
            },
            "xml": {
                "name": "Category"
            }
        },
        "Order": {
            "properties": {
                "complete": {
                    "type": "boolean"
                },
                "id": {
                    "format": "int64",
                    "type": "integer"
                },
                "petId": {
                    "format": "int64",
                    "type": "integer"
                },
                "quantity": {
                    "format": "int32",
                    "type": "integer"
                },
                "shipDate": {
                    "format": "date-time",
                    "type": "string"
                },
                "status": {
                    "description": "Order Status",
                    "enum": [
```

```
                    "placed",
                    "approved",
                    "delivered"
                ],
                "type": "string"
            }
        },
        "xml": {
            "name": "Order"
        }
    },
    "Pet": {
        "properties": {
            "category": {
                "$ref": "#/definitions/Category"
            },
            "id": {
                "format": "int64",
                "type": "integer"
            },
            "name": {
                "example": "doggie",
                "type": "string"
            },
            "photoUrls": {
                "items": {
                    "type": "string"
                },
                "type": "array",
                "xml": {
                    "name": "photoUrl",
                    "wrapped": true
                }
            },
            "status": {
                "description": "pet status in the store",
                "enum": [
                    "available",
                    "pending",
                    "sold"
                ],
                "type": "string"
            },
            "tags": {
                "items": {
                    "$ref": "#/definitions/Tag"
                },
                "type": "array",
                "xml": {
                    "name": "tag",
                    "wrapped": true
                }
            }
        },
        "required": [
            "name",
            "photoUrls"
```

```
            ],
            "xml": {
                "name": "Pet"
            }
        },
        "Tag": {
            "properties": {
                "id": {
                    "format": "int64",
                    "type": "integer"
                },
                "name": {
                    "type": "string"
                }
            },
            "xml": {
                "name": "Tag"
            }
        },
        "User": {
            "properties": {
                "email": {
                    "type": "string"
                },
                "firstName": {
                    "type": "string"
                },
                "id": {
                    "format": "int64",
                    "type": "integer"
                },
                "lastName": {
                    "type": "string"
                },
                "password": {
                    "type": "string"
                },
                "phone": {
                    "type": "string"
                },
                "userStatus": {
                    "description": "User Status",
                    "format": "int32",
                    "type": "integer"
                },
                "username": {
                    "type": "string"
                }
            },
            "xml": {
                "name": "User"
            }
        }
    },
    "host": "petstore.swagger.io",
    "info": {
        "contact": {
```

```
        "contact": {
            "email": "apiteam@wordnik.com"
        },
        "description": "This is a sample server Petstore server.  You can find out more abo
ut Swagger at <a href=\"http://swagger.io\">http://swagger.io</a> or on irc.freenode.net, #
swagger.  For this sample, you can use the api key \"special-key\" to test the authorizati
on filters",
        "license": {
            "name": "Apache 2.0",
            "url": "http://www.apache.org/licenses/LICENSE-2.0.html"
        },
        "termsOfService": "http://helloreverb.com/terms/",
        "title": "Swagger Petstore",
        "version": "1.0.0"
    },
    "paths": {
        "/pet": {
            "post": {
                "_request_body": {
                    "description": "Pet object that needs to be added to the store",
                    "in": "body",
                    "name": "body",
                    "required": false,
                    "schema": {
                        "$ref": "#/definitions/Pet"
                    }
                },
                "consumes": [
                    "application/json",
                    "application/xml"
                ],
                "description": "",
                "operationId": "addPet",
                "parameters": [
                ],
                "produces": [
                    "application/json",
                    "application/xml"
                ],
                "responses": {
                    "405": {
                        "description": "Invalid input"
                    }
                },
                "security": [
                    {
                        "petstore_auth": [
                            "write:pets",
                            "read:pets"
                        ]
                    }
                ],
                "summary": "Add a new pet to the store",
                "tags": [
                    "pet"
                ]
            },
```

```
            "put": {
                "_request_body": {
                    "description": "Pet object that needs to be added to the store",
                    "in": "body",
                    "name": "body",
                    "required": false,
                    "schema": {
                        "$ref": "#/definitions/Pet"
                    }
                },
                "consumes": [
                    "application/json",
                    "application/xml"
                ],
                "description": "",
                "operationId": "updatePet",
                "parameters": [
                ],
                "produces": [
                    "application/json",
                    "application/xml"
                ],
                "responses": {
                    "400": {
                        "description": "Invalid ID supplied"
                    },
                    "404": {
                        "description": "Pet not found"
                    },
                    "405": {
                        "description": "Validation exception"
                    }
                },
                "security": [
                    {
                        "petstore_auth": [
                            "write:pets",
                            "read:pets"
                        ]
                    }
                ],
                "summary": "Update an existing pet",
                "tags": [
                    "pet"
                ]
            }
        },
        "/pet/findByStatus": {
            "get": {
                "description": "Multiple status values can be provided with comma seperated
strings",
                "operationId": "findPetsByStatus",
                "parameters": [
                    {
                        "collectionFormat": "multi",
                        "default": "available",
                        "description": "Status values that need to be considered for filter
```

```
",
                        "in": "query",
                        "items": {
                            "type": "string"
                        },
                        "name": "status",
                        "required": false,
                        "type": "array"
                    }
                ],
                "produces": [
                    "application/json",
                    "application/xml"
                ],
                "responses": {
                    "200": {
                        "description": "successful operation",
                        "schema": {
                            "items": {
                                "$ref": "#/definitions/Pet"
                            },
                            "type": "array"
                        }
                    },
                    "400": {
                        "description": "Invalid status value"
                    }
                },
                "security": [
                    {
                        "petstore_auth": [
                            "write:pets",
                            "read:pets"
                        ]
                    }
                ],
                "summary": "Finds Pets by status",
                "tags": [
                    "pet"
                ]
            }
        },
        "/pet/findByTags": {
            "get": {
                "description": "Muliple tags can be provided with comma seperated strings.
Use tag1, tag2, tag3 for testing.",
                "operationId": "findPetsByTags",
                "parameters": [
                    {
                        "collectionFormat": "multi",
                        "description": "Tags to filter by",
                        "in": "query",
                        "items": {
                            "type": "string"
                        },
                        "name": "tags",
                        "required": false,
```

```
                "type": "array"
            }
        ],
        "produces": [
            "application/json",
            "application/xml"
        ],
        "responses": {
            "200": {
                "description": "successful operation",
                "schema": {
                    "items": {
                        "$ref": "#/definitions/Pet"
                    },
                    "type": "array"
                }
            },
            "400": {
                "description": "Invalid tag value"
            }
        },
        "security": [
            {
                "petstore_auth": [
                    "write:pets",
                    "read:pets"
                ]
            }
        ],
        "summary": "Finds Pets by tags",
        "tags": [
            "pet"
        ]
    }
},
"/pet/{petId}": {
    "delete": {
        "description": "",
        "operationId": "deletePet",
        "parameters": [
            {
                "description": "",
                "in": "header",
                "name": "api_key",
                "required": false,
                "type": "string"
            },
            {
                "description": "Pet id to delete",
                "format": "int64",
                "in": "path",
                "name": "petId",
                "required": true,
                "type": "integer"
            }
        ],
        "produces": [
```

```
                "produces": [
                    "application/json",
                    "application/xml"
                ],
                "responses": {
                    "400": {
                        "description": "Invalid pet value"
                    }
                },
                "security": [
                    {
                        "petstore_auth": [
                            "write:pets",
                            "read:pets"
                        ]
                    }
                ],
                "summary": "Deletes a pet",
                "tags": [
                    "pet"
                ]
            },
            "get": {
                "description": "Returns a pet when ID < 10.  ID > 10 or nonintegers will si
mulate API error conditions",
                "operationId": "getPetById",
                "parameters": [
                    {
                        "description": "ID of pet that needs to be fetched",
                        "format": "int64",
                        "in": "path",
                        "name": "petId",
                        "required": true,
                        "type": "integer"
                    }
                ],
                "produces": [
                    "application/json",
                    "application/xml"
                ],
                "responses": {
                    "200": {
                        "description": "successful operation",
                        "schema": {
                            "$ref": "#/definitions/Pet"
                        }
                    },
                    "400": {
                        "description": "Invalid ID supplied"
                    },
                    "404": {
                        "description": "Pet not found"
                    }
                },
                "security": [
                    {
                        "api_key": [
```

```
                    ]
                },
                {
                    "petstore_auth": [
                        "write:pets",
                        "read:pets"
                    ]
                }
            ],
            "summary": "Find pet by ID",
            "tags": [
                "pet"
            ]
        },
        "post": {
            "consumes": [
                "application/x-www-form-urlencoded"
            ],
            "description": "",
            "operationId": "updatePetWithForm",
            "parameters": [
                {
                    "description": "ID of pet that needs to be updated",
                    "in": "path",
                    "name": "petId",
                    "required": true,
                    "type": "string"
                },
                {
                    "description": "Updated name of the pet",
                    "in": "formData",
                    "name": "name",
                    "required": false,
                    "type": "string"
                },
                {
                    "description": "Updated status of the pet",
                    "in": "formData",
                    "name": "status",
                    "required": false,
                    "type": "string"
                }
            ],
            "produces": [
                "application/json",
                "application/xml"
            ],
            "responses": {
                "405": {
                    "description": "Invalid input"
                }
            },
            "security": [
                {
                    "petstore_auth": [
                        "write:pets",
                        "read:pets"
```

```
                    ]
                }
            ],
            "summary": "Updates a pet in the store with form data",
            "tags": [
                "pet"
            ]
        }
    },
    "/pet/{petId}/uploadImage": {
        "post": {
            "consumes": [
                "multipart/form-data"
            ],
            "description": "",
            "operationId": "uploadFile",
            "parameters": [
                {
                    "description": "ID of pet to update",
                    "format": "int64",
                    "in": "path",
                    "name": "petId",
                    "required": true,
                    "type": "integer"
                },
                {
                    "description": "Additional data to pass to server",
                    "in": "formData",
                    "name": "additionalMetadata",
                    "required": false,
                    "type": "string"
                },
                {
                    "description": "file to upload",
                    "in": "formData",
                    "name": "file",
                    "required": false,
                    "type": "file"
                }
            ],
            "produces": [
                "application/json"
            ],
            "responses": {
                "200": {
                    "description": "successful operation",
                    "schema": {
                        "$ref": "#/definitions/ApiResponse"
                    }
                }
            },
            "security": [
                {
                    "petstore_auth": [
                        "write:pets",
                        "read:pets"
                    ]
```

```
                ]
            }
        ],
        "summary": "uploads an image",
        "tags": [
            "pet"
        ]
    }
},
"/store/inventory": {
    "get": {
        "description": "Returns a map of status codes to quantities",
        "operationId": "getInventory",
        "produces": [
            "application/json",
            "application/xml"
        ],
        "responses": {
            "200": {
                "description": "successful operation",
                "schema": {
                    "additionalProperties": {
                        "format": "int32",
                        "type": "integer"
                    },
                    "type": "object"
                }
            }
        },
        "security": [
            {
                "api_key": [
                ]
            }
        ],
        "summary": "Returns pet inventories by status",
        "tags": [
            "store"
        ]
    }
},
"/store/order": {
    "post": {
        "_request_body": {
            "description": "order placed for purchasing the pet",
            "in": "body",
            "name": "body",
            "required": false,
            "schema": {
                "$ref": "#/definitions/Order"
            }
        },
        "description": "",
        "operationId": "placeOrder",
        "parameters": [
        ],
        "produces": [
```

```
                    "application/json",
                    "application/xml"
                ],
                "responses": {
                    "200": {
                        "description": "successful operation",
                        "schema": {
                            "$ref": "#/definitions/Order"
                        }
                    },
                    "400": {
                        "description": "Invalid Order"
                    }
                },
                "summary": "Place an order for a pet",
                "tags": [
                    "store"
                ]
            }
        },
        "/store/order/{orderId}": {
            "delete": {
                "description": "For valid response try integer IDs with value < 1000. Anyth
ing above 1000 or nonintegers will generate API errors",
                "operationId": "deleteOrder",
                "parameters": [
                    {
                        "description": "ID of the order that needs to be deleted",
                        "in": "path",
                        "name": "orderId",
                        "required": true,
                        "type": "string"
                    }
                ],
                "produces": [
                    "application/json",
                    "application/xml"
                ],
                "responses": {
                    "400": {
                        "description": "Invalid ID supplied"
                    },
                    "404": {
                        "description": "Order not found"
                    }
                },
                "summary": "Delete purchase order by ID",
                "tags": [
                    "store"
                ]
            },
            "get": {
                "description": "For valid response try integer IDs with value <= 5 or > 10.
Other values will generated exceptions",
                "operationId": "getOrderById",
                "parameters": [
                    {
```

```
                    "description": "ID of pet that needs to be fetched",
                    "in": "path",
                    "name": "orderId",
                    "required": true,
                    "type": "string"
                }
            ],
            "produces": [
                "application/json",
                "application/xml"
            ],
            "responses": {
                "200": {
                    "description": "successful operation",
                    "schema": {
                        "$ref": "#/definitions/Order"
                    }
                },
                "400": {
                    "description": "Invalid ID supplied"
                },
                "404": {
                    "description": "Order not found"
                }
            },
            "summary": "Find purchase order by ID",
            "tags": [
                "store"
            ]
        }
    },
    "/user": {
        "post": {
            "_request_body": {
                "description": "Created user object",
                "in": "body",
                "name": "body",
                "required": false,
                "schema": {
                    "$ref": "#/definitions/User"
                }
            },
            "description": "This can only be done by the logged in user.",
            "operationId": "createUser",
            "parameters": [
            ],
            "produces": [
                "application/json",
                "application/xml"
            ],
            "responses": {
                "default": {
                    "description": "successful operation"
                }
            },
            "summary": "Create user",
            "tags": [
```

```
                "user"
            ]
        }
    },
    "/user/createWithArray": {
        "post": {
            "_request_body": {
                "description": "List of user object",
                "in": "body",
                "name": "body",
                "required": false,
                "schema": {
                    "items": {
                        "$ref": "#/definitions/User"
                    },
                    "type": "array"
                }
            },
            "description": "",
            "operationId": "createUsersWithArrayInput",
            "parameters": [
            ],
            "produces": [
                "application/json",
                "application/xml"
            ],
            "responses": {
                "default": {
                    "description": "successful operation"
                }
            },
            "summary": "Creates list of users with given input array",
            "tags": [
                "user"
            ]
        }
    },
    "/user/createWithList": {
        "post": {
            "_request_body": {
                "description": "List of user object",
                "in": "body",
                "name": "body",
                "required": false,
                "schema": {
                    "items": {
                        "$ref": "#/definitions/User"
                    },
                    "type": "array"
                }
            },
            "description": "",
            "operationId": "createUsersWithListInput",
            "parameters": [
            ],
            "produces": [
                "application/json"
```

```
                "application/json",
                "application/xml"
            ],
            "responses": {
                "default": {
                    "description": "successful operation"
                }
            },
            "summary": "Creates list of users with given input array",
            "tags": [
                "user"
            ]
        }
    },
    "/user/login": {
        "get": {
            "description": "",
            "operationId": "loginUser",
            "parameters": [
                {
                    "description": "The user name for login",
                    "in": "query",
                    "name": "username",
                    "required": false,
                    "type": "string"
                },
                {
                    "description": "The password for login in clear text",
                    "in": "query",
                    "name": "password",
                    "required": false,
                    "type": "string"
                }
            ],
            "produces": [
                "application/json",
                "application/xml"
            ],
            "responses": {
                "200": {
                    "description": "successful operation",
                    "schema": {
                        "type": "string"
                    }
                },
                "400": {
                    "description": "Invalid username/password supplied"
                }
            },
            "summary": "Logs user into the system",
            "tags": [
                "user"
            ]
        }
    },
    "/user/logout": {
        "get": {
```

```
                "description": "",
                "operationId": "logoutUser",
                "produces": [
                    "application/json",
                    "application/xml"
                ],
                "responses": {
                    "default": {
                        "description": "successful operation"
                    }
                },
                "summary": "Logs out current logged in user session",
                "tags": [
                    "user"
                ]
            }
        },
        "/user/{username}": {
            "delete": {
                "description": "This can only be done by the logged in user.",
                "operationId": "deleteUser",
                "parameters": [
                    {
                        "description": "The name that needs to be deleted",
                        "in": "path",
                        "name": "username",
                        "required": true,
                        "type": "string"
                    }
                ],
                "produces": [
                    "application/json",
                    "application/xml"
                ],
                "responses": {
                    "400": {
                        "description": "Invalid username supplied"
                    },
                    "404": {
                        "description": "User not found"
                    }
                },
                "summary": "Delete user",
                "tags": [
                    "user"
                ]
            },
            "get": {
                "description": "",
                "operationId": "getUserByName",
                "parameters": [
                    {
                        "description": "The name that needs to be fetched. Use user1 for te
sting. ",
                        "in": "path",
                        "name": "username",
                        "required": true,
```

```
                    "type": "string"
                }
            ],
            "produces": [
                "application/json",
                "application/xml"
            ],
            "responses": {
                "200": {
                    "description": "successful operation",
                    "schema": {
                        "$ref": "#/definitions/User"
                    }
                },
                "400": {
                    "description": "Invalid username supplied"
                },
                "404": {
                    "description": "User not found"
                }
            },
            "summary": "Get user by user name",
            "tags": [
                "user"
            ]
        },
        "put": {
            "_request_body": {
                "description": "Updated user object",
                "in": "body",
                "name": "body",
                "required": false,
                "schema": {
                    "$ref": "#/definitions/User"
                }
            },
            "description": "This can only be done by the logged in user.",
            "operationId": "updateUser",
            "parameters": [
                {
                    "description": "name that need to be deleted",
                    "in": "path",
                    "name": "username",
                    "required": true,
                    "type": "string"
                }
            ],
            "produces": [
                "application/json",
                "application/xml"
            ],
            "responses": {
                "400": {
                    "description": "Invalid user supplied"
                },
                "404": {
                    "description": "User not found"
```

```
                    "description": "User not found"
                }
            },
            "summary": "Updated user",
            "tags": [
                "user"
            ]
        }
    }
},
"schemes": [
    "http"
],
"securityDefinitions": {
    "api_key": {
        "in": "header",
        "name": "api_key",
        "type": "apiKey"
    },
    "petstore_auth": {
        "authorizationUrl": "http://petstore.swagger.io/api/oauth/dialog",
        "flow": "implicit",
        "scopes": {
            "read:pets": "read your pets",
            "write:pets": "modify pets in your account"
        },
        "type": "oauth2"
    }
},
"swagger": "2.0",
"tags": [
    {
        "description": "Operations about user",
        "name": "user"
    },
    {
        "description": "Access to Petstore orders",
        "externalDocs": {
            "description": "Find out more",
            "url": "http://swagger.io"
        },
        "name": "store"
    },
    {
        "description": "Everything about your Pets",
        "externalDocs": {
            "description": "Find out more",
            "url": "http://swagger.io"
        },
        "name": "pet"
    }
]
}
```