

Swagger Petstore

Host: petstore.swagger.io, **Base URL:** /v2, **Version:** 1.0.0

Author: apiteam@swagger.io

Terms Of Service: <http://swagger.io/terms/>

License: [Apache 2.0](#)

This is a sample server Petstore server. You can find out more about Swagger at [http:// swagger.io](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.

Schemes: http

Summary

Pet

Everything about your Pets

Find out more: <http://swagger.io>

Operation	Description
POST /pet	Add a new pet to the store
PUT /pet	Update an existing pet
GET /pet/findByStatus	Finds Pets by status
GET /pet/findByTags	Finds Pets by tags
GET /pet/{petId}	Find pet by ID
POST /pet/{petId}	Updates a pet in the store with form data
DELETE /pet/{petId}	Deletes a pet
POST /pet/{petId}/uploadImage	uploads an image

Store

Access to Petstore orders

Operation	Description
GET /store/inventory	Returns pet inventories by status
POST /store/order	Place an order for a pet
GET /store/order/{orderId}	Find purchase order by ID
DELETE /store/order/{orderId}	Delete purchase order by ID

User

Operations about user

Find out more about our store: <http://swagger.io>

Operation	Description
POST /user	Create user
POST /user/createWithArray	Creates list of users with given input array
POST /user/createWithList	Creates list of users with given input array
GET /user/login	Logs user into the system
GET /user/logout	Logs out current logged in user session
GET /user/{username}	Get user by user name
PUT /user/{username}	Updated user
DELETE /user/{username}	Delete user

Paths

POST /pet

Tags: pet
Add a new pet to the store

REQUEST BODY

Pet object that needs to be added to the store
Request content-types: application/json, application/xml

DEFINITION

PROPERTIES

id: integer (int64)

category: undefined

PROPERTIES

id: integer (int64)

name: string

name: string

photoUrls: string[]

ITEMS: string[]

tags: object[]

ITEMS: object[]

PROPERTIES

id: integer (int64)

name: string

status: string, enum("available", "pending", "sold")
pet status in the store

RESPONSE

Response content-types: application/xml, application/json
405 Method Not Allowed
Invalid input

PUT /pet

Tags: pet
Update an existing pet

REQUEST BODY

Pet object that needs to be added to the store
Request content-types: application/json, application/xml

DEFINITION

PROPERTIES

id: integer (int64)

category: undefined

PROPERTIES

id: integer (int64)

name: string

name: string

photoUrls: string[]

ITEMS: string[]

tags: object[]

ITEMS: object[]

PROPERTIES

id: *integer (int64)*

name: *string*

status: *string, enum("available", "pending", "sold")*
pet status in the store

RESPONSE

Response content-types: application/xml, application/json

400 Bad Request

Invalid ID supplied

404 Not Found

Pet not found

405 Method Not Allowed

Validation exception

GET /pet/findByStatus

Tags: pet

Multiple status values can be provided with comma separated strings

REQUEST PARAMETERS

Name	Description	Type	Data Type
status	Status values that need to be considered for filter	query	string[] [Required]

RESPONSE

Response content-types: application/xml, application/json

200 OK

successful operation

DEFINITION

ITEMS: []

PROPERTIES

id: *integer (int64)*

category: *undefined*

PROPERTIES

id: *integer (int64)*

name: *string*

name: *string*

photoUrls: *string[]*

ITEMS: *string[]*

tags: *object[]*

ITEMS: *object[]*

PROPERTIES

id: *integer (int64)*

name: *string*

status: *string, enum("available", "pending", "sold")*
pet status in the store

400 Bad Request

Invalid status value

GET /pet/findByTags

Tags: pet

Multiple tags can be provided with comma separated strings. Use tag1, tag2, tag3 for testing.

REQUEST PARAMETERS

Name	Description	Type	Data Type
tags	Tags to filter by	query	string[] [Required]

RESPONSE

Response content-types: application/xml, application/json

200 OK

successful operation

DEFINITION

```

ITEMS: []
PROPERTIES
  id: integer (int64)
  category: undefined
    PROPERTIES
      id: integer (int64)
      name: string
    name: string
    photoUrls: string[]
    ITEMS: string[]
    tags: object[]
    ITEMS: object[]
    PROPERTIES
      id: integer (int64)
      name: string
    status: string, enum("available", "pending", "sold")
      pet status in the store

```

400 Bad Request

Invalid tag value

GET /pet/{petId}

Tags: pet

Returns a single pet

REQUEST PARAMETERS

Name	Description	Type	Data Type
petId	ID of pet to return	path	integer [Required] (int64)

RESPONSE

Response content-types: application/xml, application/json

200 OK

successful operation

See section: [#/definitions/Pet](#)

400 Bad Request

Invalid ID supplied

404 Not Found

Pet not found

POST /pet/{petId}

Tags: pet

Updates a pet in the store with form data

REQUEST PARAMETERS

Name	Description	Type	Data Type
petId	ID of pet that needs to be updated	path	integer [Required] (int64)

Name	Description	Type	Data Type
name	Updated name of the pet	formData	string
status	Updated status of the pet	formData	string

RESPONSE

Response content-types: application/xml, application/json

405 Method Not Allowed
Invalid input

DELETE /pet/{petId}

Tags: pet
Deletes a pet

REQUEST PARAMETERS

Name	Description	Type	Data Type
api_key		header	string
petId	Pet id to delete	path	integer [Required] (int64)

RESPONSE

Response content-types: application/xml, application/json

400 Bad Request
Invalid ID supplied

404 Not Found
Pet not found

POST /pet/{petId}/uploadImage

Tags: pet
uploads an image

REQUEST PARAMETERS

Name	Description	Type	Data Type
petId	ID of pet to update	path	integer [Required] (int64)
additionalMetadata	Additional data to pass to server	formData	string
file	file to upload	formData	file

RESPONSE

Response content-types: application/json

200 OK
successful operation
See section: #/definitions/ApiResponse

GET /store/inventory

Tags: store
Returns a map of status codes to quantities

RESPONSE

Response content-types: application/json

200 OK
successful operation

DEFINITION

PROPERTIES

POST /store/order

Tags: store
Place an order for a pet

REQUEST BODY

order placed for purchasing the pet

DEFINITION

PROPERTIES

id: integer (int64)

petId: integer (int64)

quantity: integer (int32)

shipDate: string (date-time)

status: string, enum("placed", "approved", "delivered")
Order Status

complete: boolean (default:"false")

RESPONSE

Response content-types: application/xml, application/json

200 OK
successful operation

See section: #/definitions/Order

400 Bad Request
Invalid Order

GET /store/order/{orderId}

Tags: store
For valid response try integer IDs with value >= 1 and <= 10. Other values will generated exceptions

REQUEST PARAMETERS

Name	Description	Type	Data Type
orderId	ID of pet that needs to be fetched	path	integer [Required] (int64) (1 ≤ x ≤ 10)

RESPONSE

Response content-types: application/xml, application/json

200 OK
successful operation

See section: #/definitions/Order

400 Bad Request
Invalid ID supplied

404 Not Found
Order not found

DELETE /store/order/{orderId}

Tags: store
For valid response try integer IDs with positive integer value. Negative or non-integer values will generate API errors

REQUEST PARAMETERS

Name	Description	Type	Data Type
orderId	ID of the order that needs to be deleted	path	integer [Required] (int64) (1 ≤ x < +∞)

RESPONSE

Response content-types: application/xml, application/json

400 Bad Request
Invalid ID supplied

404 Not Found
Order not found

POST /user

Tags: user

This can only be done by the logged in user.

REQUEST BODY

Created user object

DEFINITION

PROPERTIES

id: *integer (int64)*
username: *string*
firstName: *string*
lastName: *string*
email: *string*
password: *string*
phone: *string*
userStatus: *integer (int32)*
User Status

RESPONSE

Response content-types: application/xml, application/json

default

successful operation

POST /user/createWithArray

Tags: user

Creates list of users with given input array

REQUEST BODY

List of user object

DEFINITION

ITEMS: []

PROPERTIES

id: *integer (int64)*
username: *string*
firstName: *string*
lastName: *string*
email: *string*
password: *string*
phone: *string*
userStatus: *integer (int32)*
User Status

RESPONSE

Response content-types: application/xml, application/json

default

successful operation

POST /user/createWithList

Tags: user

Creates list of users with given input array

REQUEST BODY

List of user object

DEFINITION

ITEMS: []

PROPERTIES

id: integer (int64)

username: string

firstName: string

lastName: string

email: string

password: string

phone: string

userStatus: integer (int32)

User Status

RESPONSE

Response content-types: application/xml, application/json

default
successful operation

GET /user/login

Tags: user
Logs user into the system

REQUEST PARAMETERS

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

RESPONSE

Response content-types: application/xml, application/json

200 OK
successful operation

DEFINITION

400 Bad Request
Invalid username/password supplied

GET /user/logout

Tags: user
Logs out current logged in user session

RESPONSE

Response content-types: application/xml, application/json

default
successful operation

GET /user/{username}

Tags: user
Get user by user name

REQUEST PARAMETERS

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

RESPONSE

Response content-types: application/xml, application/json

200 OK

successful operation

See section: [#/definitions/User](#)

400 Bad Request

Invalid username supplied

404 Not Found

User not found

PUT /user/{username}

Tags: user

This can only be done by the logged in user.

REQUEST PARAMETERS

Name	Description	Type	Data Type
username	name that need to be updated	path	string [Required]

REQUEST BODY

Updated user object

DEFINITION

<p>PROPERTIES</p> <ul style="list-style-type: none"> id: <i>integer (int64)</i> username: <i>string</i> firstName: <i>string</i> lastName: <i>string</i> email: <i>string</i> password: <i>string</i> phone: <i>string</i> userStatus: <i>integer (int32)</i> User Status

RESPONSE

Response content-types: application/xml, application/json

400 Bad Request

Invalid user supplied

404 Not Found

User not found

DELETE /user/{username}

Tags: user

This can only be done by the logged in user.

REQUEST PARAMETERS

Name	Description	Type	Data Type
username	The name that needs to be deleted	path	string [Required]

RESPONSE

Response content-types: application/xml, application/json

400 Bad Request

Invalid username supplied

404 Not Found

User not found

Definitions

Order

DEFINITION

PROPERTIES

id: *integer (int64)*

petId: *integer (int64)*

quantity: *integer (int32)*

shipDate: *string (date-time)*

status: *string, enum("placed", "approved", "delivered")*
Order Status

complete: *boolean (default:"false")*

Category

DEFINITION

PROPERTIES

id: *integer (int64)*

name: *string*

User

DEFINITION

PROPERTIES

id: *integer (int64)*

username: *string*

firstName: *string*

lastName: *string*

email: *string*

password: *string*

phone: *string*

userStatus: *integer (int32)*
User Status

Tag

DEFINITION

PROPERTIES

id: *integer (int64)*

name: *string*

Pet

DEFINITION

PROPERTIES

id: *integer (int64)*

category: *undefined*

PROPERTIES

id: *integer (int64)*

name: *string*

name: *string*

```
photoUrls: string[]
  ITEMS: string[]
tags: object[]
  ITEMS: object[]
  PROPERTIES
    id: integer (int64)
    name: string
status: string, enum("available", "pending", "sold")
  pet status in the store
```

ApiResponse

DEFINITION

```
PROPERTIES
  code: integer (int32)
  type: string
  message: string
```