OpenAPI Petstore

Table of Contents

Abstract

This is a sample server Petstore server. For this sample, you can use the

api key `special-key` *to test the authorization filters.*

1. Access

- **APIKey** KeyParamName: *api_key*, KeyInQuery: *false*, KeyInHeader: *true*
- **OAuth** AuthorizationUrl: *http://petstore.swagger.io/api/oauth/dialog*, TokenUrl: __

2. Endpoints

2.1. Pet

2.1.1. addPet

`POST /pet`

Add a new pet to the store

Description

Parameters

Body Parameter

Name	Description	Required	Default	Pattern
Pet	Pet object that needs to be added to the store *Pet* a Pet	X		

Return Type

Pet a Pet

Content Type

- application/xml
- application/json

Responses

Table 1. http response codes

Code	Message	Datatype
200	successful operation	*Pet* a Pet
405	Invalid input	<<>>

Samples

2.1.2. deletePet

`DELETE /pet/{petId}`

Deletes a pet

Description

Parameters

Path Parameters

Name	Description	Required	Default	Pattern
petId	Pet id to delete	X	null	

Header Parameters

Name	Description	Required	Default	Pattern
api_key		-	null	

Return Type

-

Responses

Table 2. http response codes

Code	Message	Datatype
400	Invalid pet value	<<>>

Samples

2.1.3. findPetsByStatus

`GET /pet/findByStatus`

Finds Pets by status

Description

Multiple status values can be provided with comma separated strings

Parameters

Query Parameters

Name	Description	Required	Default	Pattern
status	Status values that need to be considered for filter [String]	X	null	

Return Type

array[*Pet* a Pet]

Content Type

- application/xml
- application/json

Responses

Table 3. http response codes

Code	Message	Datatype
200	successful operation	List[*Pet* a Pet]
400	Invalid status value	<<>>

Samples

2.1.4. findPetsByTags

`GET /pet/findByTags`

Finds Pets by tags

Description

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

Parameters

Query Parameters

Name	Description	Required	Default	Pattern
tags	Tags to filter by [String]	X	null	

Return Type

array[*Pet* a Pet]

Content Type

- application/xml
- application/json

Responses

Table 4. http response codes

Code	Message	Datatype
200	successful operation	List[*Pet* a Pet]
400	Invalid tag value	<<>>

Samples

2.1.5. getPetById

`GET /pet/{petId}`

Find pet by ID

Description

Returns a single pet

Parameters

Path Parameters

Name	Description	Required	Default	Pattern
petId	ID of pet to return	X	null	

Return Type

Pet a Pet

Content Type

- application/xml
- application/json

Responses

Table 5. http response codes

Code	Message	Datatype
200	successful operation	*Pet* a Pet
400	Invalid ID supplied	<<>>
404	Pet not found	<<>>

Samples

2.1.6. updatePet

`PUT /pet`

Update an existing pet

Description

Parameters

Body Parameter

Name	Description	Required	Default	Pattern
Pet	Pet object that needs to be added to the store *Pet* a Pet	X		

Return Type

Pet a Pet

Content Type

- application/xml
- application/json

Responses

Table 6. http response codes

Code	Message	Datatype
200	successful operation	*Pet* a Pet
400	Invalid ID supplied	<<>>
404	Pet not found	<<>>
405	Validation exception	<<>>

Samples

2.1.7. updatePetWithForm

`POST /pet/{petId}`

Updates a pet in the store with form data

Description

Parameters

Path Parameters

Name	Description	Required	Default	Pattern
petId	ID of pet that needs to be updated	X	null	

Form Parameters

Name	Description	Required	Default	Pattern
name	Updated name of the pet [string]	-	null	
status	Updated status of the pet [string]	-	null	

Return Type

-

Responses

Table 7. http response codes

Code	Message	Datatype
405	Invalid input	<<>>

Samples

2.1.8. uploadFile

`POST /pet/{petId}/uploadImage`

uploads an image

Description

Parameters

Path Parameters

Name	Description	Required	Default	Pattern
petId	ID of pet to update	X	null	

Form Parameters

Name	Description	Required	Default	Pattern
additionalMetadata	Additional data to pass to server [string]	-	null	
file	file to upload [file]	-	null	

Return Type

ApiResponse An uploaded response

Content Type

- application/json

Responses

Table 8. http response codes

Code	Message	Datatype
200	successful operation	*ApiResponse* An uploaded response

Samples

2.2. Store

2.2.1. deleteOrder

`DELETE /store/order/{orderId}`

Delete purchase order by ID

Description

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

Parameters

Path Parameters

Name	Description	Required	Default	Pattern
orderId	ID of the order that needs to be deleted	X	null	

Return Type

-

Responses

Table 9. http response codes

Code	Message	Datatype
400	Invalid ID supplied	<<>>

Code	Message	Datatype
404	Order not found	<<>>

Samples

2.2.2. getInventory

`GET /store/inventory`

Returns pet inventories by status

Description

Returns a map of status codes to quantities

Parameters

Return Type

[Map]

Content Type

- application/json

Responses

Table 10. http response codes

Code	Message	Datatype
200	successful operation	Map[[integer]]

Samples

2.2.3. getOrderById

`GET /store/order/{orderId}`

Find purchase order by ID

Description

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

Parameters

Path Parameters

Name	Description	Required	Default	Pattern
orderId	ID of pet that needs to be fetched	X	null	

Return Type

Order Pet Order

Content Type

- application/xml
- application/json

Responses

Table 11. http response codes

Code	Message	Datatype
200	successful operation	*Order* Pet Order
400	Invalid ID supplied	<<>>
404	Order not found	<<>>

Samples

2.2.4. placeOrder

`POST /store/order`

Place an order for a pet

Description

Parameters

Body Parameter

Name	Description	Required	Default	Pattern
Order	order placed for purchasing the pet *Order* Pet Order	X		

Return Type

Order Pet Order

Content Type

- application/xml
- application/json

Responses

Table 12. http response codes

Code	Message	Datatype
200	successful operation	*Order* Pet Order
400	Invalid Order	<<>>

Samples

2.3. User

2.3.1. createUser

`POST /user`

Create user

Description

This can only be done by the logged in user.

Parameters

Body Parameter

Name	Description	Required	Default	Pattern
User	Created user object *User* a User	X		

Return Type

-

Responses

Table 13. http response codes

Code	Message	Datatype
0	successful operation	<<>>

Samples

2.3.2. createUsersWithArrayInput

`POST /user/createWithArray`

Creates list of users with given input array

Description

Parameters

Body Parameter

Name	Description	Required	Default	Pattern
User	List of user object *User* a User	X		

Return Type

-

Responses

Table 14. http response codes

Code	Message	Datatype
0	successful operation	<<>>

Samples

2.3.3. createUsersWithListInput

`POST /user/createWithList`

Creates list of users with given input array

Description

Parameters

Body Parameter

Name	Description	Required	Default	Pattern
User	List of user object *User* a User	X		

Return Type

-

Responses

Table 15. http response codes

Code	Message	Datatype
0	successful operation	<<>>

Samples

2.3.4. deleteUser

`DELETE /user/{username}`

Delete user

Description

This can only be done by the logged in user.

Parameters

Path Parameters

Name	Description	Required	Default	Pattern
username	The name that needs to be deleted	X	null	

Return Type

-

Responses

Table 16. http response codes

Code	Message	Datatype
400	Invalid username supplied	<<>>
404	User not found	<<>>

Samples

2.3.5. getUserByName

`GET /user/{username}`

Get user by user name

Description

Parameters

Path Parameters

Name	Description	Required	Default	Pattern
username	The name that needs to be fetched. Use user1 for testing.	X	null	

Return Type

User a User

Content Type

- application/xml
- application/json

Responses

Table 17. http response codes

Code	Message	Datatype
200	successful operation	*User* a User
400	Invalid username supplied	<<>>
404	User not found	<<>>

Samples

2.3.6. loginUser

`GET /user/login`

Logs user into the system

Description

Parameters

Query Parameters

Name	Description	Required	Default	Pattern
username	The user name for login	X	null	/^[a-zA-Z0-9\\.\\-_]*[a-zA-Z0-9]$/
password	The password for login in clear text	X	null	

Return Type

[String]

Content Type

- application/xml
- application/json

Responses

Table 18. http response codes

Code	Message	Datatype
200	successful operation	[String]
400	Invalid username/password supplied	<<>>

Samples

2.3.7. logoutUser

`GET /user/logout`

Logs out current logged in user session

Description

Parameters

Return Type

-

Responses

Table 19. http response codes

Code	Message	Datatype
0	successful operation	<<>>

Samples

2.3.8. updateUser

`PUT /user/{username}`

Updated user

Description

This can only be done by the logged in user.

Parameters

Path Parameters

Name	Description	Required	Default	Pattern
username	name that need to be deleted	X	null	

Body Parameter

Name	Description	Required	Default	Pattern
User	Updated user object *User* a User	X		

Return Type

-

Responses

Table 20. http response codes

Code	Message	Datatype
400	Invalid user supplied	<<>>
404	User not found	<<>>

Samples

3. Models

3.1. *ApiResponse* An uploaded response

Describes the result of uploading an image resource

Field Name	Required	Type	Description	Format
code		Integer		int32
type		String		
message		String		

3.2. *Category* Pet category

A category for a pet

Field Name	Required	Type	Description	Format
id		Long		int64
name		String		

3.3. *Order* Pet Order

An order for a pets from the pet store

Field Name	Required	Type	Description	Format
id		Long		int64
petId		Long		int64
quantity		Integer		int32
shipDate		Date		date-time
status		String	Order Status	*Enum:* placed, approved, delivered,
complete		Boolean		

3.4. *Pet* a Pet

A pet for sale in the pet store

Field Name	Required	Type	Description	Format
id		Long		int64
category		Category		
name	X	String		
photoUrls	X	List of [string]		
tags		List of *Tag* Pet Tag		
status		String	pet status in the store	*Enum:* available, pending, sold,

3.5. *Tag* Pet Tag

A tag for a pet

Field Name	Required	Type	Description	Format
id		Long		int64
name		String		

3.6. *User* a User

A User who is purchasing from the pet store

Field Name	Required	Type	Description	Format
id		Long		int64
username		String		
firstName		String		
lastName		String		
email		String		
password		String		
phone		String		
userStatus		Integer	User Status	int32